Discrete Geometric Dynamics and Artistic Control of Curves and Surfaces

Miklós Bergou

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2010

ABSTRACT

Discrete Geometric Dynamics and Artistic Control of Curves and Surfaces

Miklós Bergou

Physical simulations are capable of highly realistic and often very accurate reproductions of the true systems that they are modeling. However, the computational cost of a simulation can often limit its usefulness in contexts where not just the accuracy of the results for a single set of parameters but also the promptness of displaying the result to the user is of interest. This type of scenario can often arise not only in real-time applications, such as in virtual environments, training simulators, or games for example, but also in interactive design contexts in which the user is trying to guide the system toward some desired goal.

We explore two avenues toward achieving efficient, controllable simulations. We first consider how to ensure that a physical simulation matches a user-prescribed input for the problem of simulating thin, flexible objects such as cloth. Within an iterative design framework, this lets the user run lower resolution preview simulations with a much faster turn-around time than a full simulation and guarantees that the final, high-resolution output matches the previews. In addition, we demonstrate that this system can be applied not just to physically simulated motion, but also to hand-animated motion. This allows the artist to focus on animating the expressive elements of the motion without having to worry about manually creating details such as characteristic wrinkles and folds of cloth.

In the second avenue, we apply ideas from Discrete Differential Geometry (DDG) to the problem of deriving principled, efficient computational models for simulating the dynamics of physical systems. The main mandate of DDG is to identify the key geometric structures underlying a system and to preserve these structures when building a discrete model. We build on this foundation within the context of physical simulation to arrive at discrete geo-

metric models for dynamics, and we show the benefit of constructing the discrete models in this geometric manner. We first show that leveraging the assumption of isometric deformations leads naturally to an expression for the bending energy of thin plates that is *quadratic* in the position of the surface. The associated linear gradient and constant Hessian provide an efficient model for simulating cloth, which we compare to the full, nonlinear model. We next apply this line of thinking to simulating long, thin objects such as elastic rods or viscous threads. By focusing on the geometry of the model and drawing on the parallels to the smooth theory, we arrive at an efficient, unified model for simulating these types of objects, which we validate against a number of analytic and experimental results.

“Discrete differential geometry initially arose from the observation that when a notion from smooth geometry (such as that of a minimal surface) is discretized ‘properly’, the discrete objects are not merely approximations of the smooth ones, but have special properties of their own, which make them form a coherent entity by themselves. One might suggest many different reasonable discretizations with the same smooth limit. Among these, which one is the best? From the theoretical point of view, the best discretization is the one which preserves the fundamental properties of the smooth theory. Often such a discretization clarifies the structures of the smooth theory and possesses important connections to other fields of mathematics, for instance to projective geometry, integrable systems, algebraic geometry, or complex analysis. The discrete theory is in a sense the more fundamental one: the smooth theory can always be recovered as a limit, while it is a nontrivial problem to find which discretization has the desired properties.”

—Excerpt from [Bobenko et al., 2008]

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would first like to thank my advisor, Eitan Grinspun, for the countless ways in which he influenced my development as a researcher and for providing the impetus for my joining his work on the topics presented in this thesis. None of my work would have been possible without his creativity, guidance, and help.

I am also forever indebted to all of my collaborators, who each deserve a large portion of the credit for this work coming to fruition: Basile Audoly, David Harmon, Saurabh Mathur, Stephen Robinson, Etienne Vouga, Max Wardetzky, and Denis Zorin. I would like to thank my "adoptive advisors," Basile Audoly and Max Wardetzky. You have taught me more over the past few years than I can possibly remember. I would also like to thank my graduate student accomplices, David Harmon, Saurabh Mathur, Stephen Robinson, and Etienne Vouga, for working endless nights trying to finish our projects before the deadline.

I would like to thank Kevin Egan, Charles Han, Jungseock Joo, and Breannan Smith for their help, advice, and friendship.

Finally, I would like to thank my parents, János and Valéria, and my brother and sister, Attila and Katalin, for their unending love and support.

Chapter 1

Introduction

Current physical simulations are able to achieve a high degree of accuracy and visual realism due to both the increased availability of computational resources and the multitude of researchers from a variety of fields, such as engineering, physics, and computer graphics, which has lead to numerous advances in simulation techniques. For example, the graphical treatment of viscoelastica alone encompasses hundreds of works covered in recent surveys [Witkin and Baraff, 2001; Bridson et al., 2006; Nealen et al., 2006]. More accurate measurements and determination of the characteristics describing a physical system have led to models that can more faithfully reproduce the behavior of the system. At the same time, the development of accurate, general-purpose numerical methods has eased the burden of validating these models and has made it possible to use simulation as a tool for exploring and predicting the behavior of a physical system.

Despite these advances, the computational cost associated with high-resolution simulations can be prohibitively expensive for many applications. Requirements on the desired accuracy of the result typically translate into requirements on the necessary resolution of the model, leading to large, computationally expensive systems. Compounded by the fact that increasingly complex models are being developed for simulating a wider range of phenomena, simulations can often take hours, days, or even more to compute the equivalent of a few seconds.

In addition, controlling the outcome of a simulation is often orders of magnitude slower than simply running the simulation due to the large number of iterations that are required, the added complexity of the system when methods for direct control are added, or some combination of both.

1.1 Contributions and overview

Structure preserving, quadratic bending energy formulation Whereas previously, the bending energy and associated force of a thin plate have always been considered to be nonlinear in the position of the surface, in §2, we show that for the special case of isometric deformations, the bending energy and force can be reformulated as quadratic and linear, respectively. Unlike traditional approaches to linearizing the forces, our model preserves key properties of the nonlinear model, including invariance to rigid motions (including rotations), which is an important prerequisite for conservation of linear and angular momentum, and uniform scaling of the surface, which is an important property that affects the shape of the wrinkles and folds on the surface. By building on our observation, we develop a family of discrete bending energies that by construction are quadratic in vertex positions, which leads to an efficient computational model for simulating cloth. We additionally show how this formulation can be applied to geometry processing algorithms, resulting in the first reported interactive rates for surface smoothing and hole filling applications based on Willmore flow.

Unified, implicit framework for elastic rods and viscous threads In §3, we describe the kinematics of adapted framed curves from the perspective of DDG, developing the discrete along with the smooth theory, with a focus on the impact that the choice of reference frame in the curve-angle representation for adapted framed curves has on the computation of the kinematic quantities. To the best of our knowledge, we are the first to advocate the time-parallel reference frame, which we use to build an implicit framework for simulating elastic rods in §4. Further, we extend this work in §5 to create the first reported model of thin, viscous threads based on the Rayleigh analogy to elastic rods. We validate both

our elastic rod and viscous thread models against a number of analytical and experimental results, and we show for the first time the numerical reproduction of "sewing-machine" patterns that emerge when a thin thread of viscous fluid falls onto a moving conveyor belt (see Fig. 5.1).

Iterative design Our work on applying DDG to physical simulation can be seen as an attempt to achieve fast, qualitatively correct simulations even for coarse discretizations by preserving the key geometric structures of the physical system. Within the context of iterative design, coarse discretizations or "previews" are often used in order to cut down on the computational cost of each iteration and achieve more interactive simulations. However, it is often unreasonable to expect that a coarse simulation accurately predicts the trajectory of a high-resolution simulation. We view this in the broader context of the problem we refer to as *tracking*, which we define below.

The tracking problem defined Given a user-specified input (the guide trajectory), the tracking problem asks for the output from the physical simulation (the tracked trajectory) to "match" the input. The guide trajectory could be from a coarse simulation, as in the iterative design cycle, but it could also be a hand-animated trajectory that the artist wants the system to follow. For a method to qualify as a solution to the tracking problem, we require that it (i) be *tractable* for high-resolution output, (ii) enhance the input with *dynamic detail* that is typical of simulations run at that resolution, (iii) be able to *handle contact* and other complexities of the dynamics, and (iv) *provide controls* for specifying how the input motion should be matched. Previous methods aimed at tracking a user-specified input motion do not satisfy all of these requirements for a solution. Methods based on keyframe-type constraints are tractable only for systems with a small number of degrees of freedom, provide control only at certain instants in time with no control in-between keyframes, and cannot handle complex collisions because they require the entire trajectory to be smooth. Methods based on control forces provide no guarantee of matching because the output could drift arbitrarily far from the input depending on strength of other forces in the system. Geometric methods, such as subdivision or adding procedurally generated

wrinkles, provide no additional dynamics; in fact, for hand-animated motion, they are missing all dynamical effects.

Directable thin shells Our framework for directable thin shells described in §6 is the first to guarantee in a single run of the simulation that the result matches a given surface animation while providing artistic control over the scale of the physically simulated, dynamic detail that is introduced. The method ensures that the simulation satisfies user-defined tracking constraints but can also handle situations involving complex contact. We demonstrate this system in the context of an iterative design cycle, in which a user runs multiple low-resolution preview simulations until some desired goal is achieved before committing resources to the final, high-resolution simulation. We also show the applicability of our framework to artist-driven motion, in which the input is not from a low-resolution simulation, but is instead hand-animated by an artist. To the best of our knowledge, this type of spatial and temporal collocation of artistic animation and thin shell physics has not been previously explored.

Chapter 2

Quadratic curvature energies

Curvature-based energies are important for a variety of applications, ranging from physical simulation of cloth and garments based on the mechanics of thin plates and shells to geometry processing operations such as smoothing, denoising, and hole filling. However, computation of these energies—and in particular their derivatives—is often a costly component of these applications.

We show that under the assumption of *isometric* surface deformations, these curvature-based energies are *quadratic* in the position of the surface. The resulting isometric bending model is shown to significantly speed up common cloth solvers and, when applied to geometry processing operations, to provide runtimes that are close to interactive rates. As compared to the existing nonlinear methods, we demonstrate a two- to three-fold net speedup of cloth simulations and interactive geometry processing applications.

2.1 Introduction

Efficient computation of curvature-based energies is important for practical implementations of physical simulation and geometric modeling applications. Any coordinate-invariant curvature energy may be expressed in terms of the *principal curvatures*, κ_1 and κ_2, of the surface, and desired symmetries often lead to expressions in terms of the elementary

symmetric functions: mean ($H = \kappa_1 + \kappa_2$), Gaussian ($K = \kappa_1\kappa_2$), and total ($\kappa_1^2 + \kappa_2^2$) curvature. Energies assembled from these elementary expressions are widely used in geometric modeling [Schneider and Kobbelt, 2001; Clarenz et al., 2004; Hildebrandt and Polthier, 2004; Bobenko and Schröder, 2005; Deckelnick et al., 2005; Grinspun et al., 2006], as well as physical simulation of 2D deformable objects, such as cloth and thin shells (see the survey of Thomaszewski and Wacker [2006] and the theory of Ciarlet [2000]).

2.1.1 Bending energy formulation

We consider the bending energy functional of a surface S given by

$$E_b = \frac{1}{2}\int_S H^2 \mathrm{d}A \ , \tag{2.1}$$

In the continuous case, this energy is invariant under rigid motions and uniform scaling of the surface (in fact, it is invariant under all Möbius transformations of ambient 3-space [White, 2000]).

We explore the interrelation between isometry, differential operators, and curvature energy. The importance of isometry for simplification of energy was previously acknowledged in the context of surface fairing and modeling by, e.g., Desbrun et al. [1999a] and later Schneider and Kobbelt [2001]. We focus primarily on the simulation of inextensible plates where physics dictates quasi-isometry: membrane stiffness typically is greater than bending stiffness by four or more orders of magnitude [Ciarlet, 2000], and the advantages of isometry can be exploited in full.

2.1.2 Central observation

In the continuous setting, E_b is *quadratic in positions*, which can be shown using the following argument [Bergou et al., 2006; Wardetzky et al., 2007]. A Riemannian metric (first fundamental form) of a smooth surface is defined by assigning a smoothly varying inner product to each tangent plane of the surface. This gives rise to intrinsic notions of gradient and divergence, and hence a Laplace-Beltrami operator $\Delta = -\mathrm{div}\,\mathrm{grad}$ on the surface [do Carmo,

Figure 2.1: These images show snapshots from a simulation of a flag using the isometric bending model to compute bending forces. Despite its economy of cost, the proposed bending model achieves qualitatively the same dynamics as popular nonlinear models.

1992]. If the surface is embedded into Euclidean 3-space, we refer to this metric as the metric inherited from $\mathbb{E}^3$. Now consider deformations of some fixed reference surface S. A deformation corresponds to changing the embedding $\mathbf{x} : S \to \mathbb{E}^3$ and is called isometric if the metric inherited by $\mathbf{x}(S)$ is the same as that of the reference surface S. Assuming isometric deformations, the mean curvature vector of $\mathbf{x}(S)$ is given by $\mathbf{H}(\mathbf{x}) = H\mathbf{n} = \Delta\mathbf{x}$, where $\mathbf{n}$ is the (unit) normal vector to the surface and Δ is the Laplace-Beltrami operator on the *reference* surface. We can therefore rewrite (2.1) as

$$E_b(\mathbf{x}) = \frac{1}{2}\int_S \langle \Delta\mathbf{x}, \Delta\mathbf{x}\rangle \mathrm{d}A \ . \tag{2.2}$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product of $\mathbb{E}^3$. Since the intrinsic Laplacian Δ remains unchanged under isometric deformations of the surface, we infer that $E_b(\mathbf{x})$ is quadratic in positions under isometric deformations. We will refer to $E_b(\mathbf{x})$ written in the form of (2.2) as the *isometric bending model* (IBM).

2.1.3 Overview

The main purpose of the current chapter is twofold: showing the transition of the continuous IBM to a discrete one and exploiting this discrete model for speeding up computations involving inextensible thin materials. Our aim is to seek operators that satisfy a discrete IBM: a bending energy quadratic in positions under the class of discrete isometric deformations. In the discrete setting, one begins with a discrete mean curvature or discrete Laplacian

if one seeks a discrete version of (2.1) or (2.2), respectively. In analogy to the smooth case, a discrete IBM will be based on Laplacians that are—by construction—invariant under discrete isometric deformations. Fortunately, DDG operators are extensively studied [Pinkall and Polthier, 1993; Mercat, 2001; Meyer et al., 2003; Bobenko and Schröder, 2005], and the necessary discrete operators are readily available. These are obtained by building on a mixture of conforming (vertex-based) and nonconforming (edge-based) linear finite elements.

Contributions Our main contributions are:

- We describe a family of discretizations of the IBM that are (*i*) *invariant under rigid transformations and uniform scaling* and (*ii*) *quadratic in mesh positions*.
- We show that virtually any cloth or thin plate simulator that requires the computation of bending forces can easily incorporate this discrete IBM, and in doing so, we observed a 2–3× net speedup in computing the dynamics for representative solvers.
- We remark that using the IBM as a surrogate energy in the semi-implicit solution of Willmore flow enables hole filling and non-shrinking smoothing applications at nearly interactive rates.

2.2 Integrated vs. pointwise quantities

In the smooth setting, certain quantities are defined *pointwise* over some domain, $\mathcal{D}$: there is a *function* that assigns a value to each point in the domain. On the other hand, in the discrete setting, certain quantities often emerge naturally in an *integrated* way as a *functional* that associates the integral of the function value to the domain. For example, relating the discrete curvature of a piecewise linear curve to the turning angle between incident edges corresponds to an integration over the curve's Gauss image [Grinspun and Wardetzky, 2008]. The need for a clear distinction between functions and functionals becomes immediate when applying operations (e.g., squaring the pointwise mean-curvature and then integrating, $\int_{\mathcal{D}} H^2$, is *not* the same as integrating the mean curvature and then squaring, $\left(\int_{\mathcal{D}} H\right)^2$).

In more general terms, we think of a functional F as associating to a function f its integrated value over some domain $\mathcal{D}$. The opposite direction, going from a functional F to a function f, involves *averaging* (dividing) by the area of $\mathcal{D}$. This underscores the relevance of Voronoi regions and barycentric area in the geometric modeling literature. Extending this notation, we may evaluate the functional over an array of domains, $\mathcal{D}_i$. In matrix-vector form, this reads $f = M^{-1}F$, where M is a matrix with diagonal entries $|\mathcal{D}_i|$ corresponding to areas. The key observation is that M^{-1} projects evaluations of a functional to their corresponding ("averaged") function values. This relation may be derived in a finite-element framework where M (now not necessarily diagonal) retains the meaning shown here.

2.3 Related work

2.3.1 Physical models of cloth and thin plates

Much of the work in computer graphics on cloth simulation traces its roots back to the pioneering work of Terzopoulos et al. [1987], who introduced tensorial treatment of elastica in graphics, and of Baraff and Witkin [1998], who demonstrated the power of a semi-implicit integration framework for cloth. These methods spurred numerous significant developments in the treatment of numerics [Ascher and Boxerman, 2003; Hauth et al., 2003], contact response [Bridson et al., 2002; Baraff et al., 2003], and constitutive models [Choi and Ko, 2002]. For a more comprehensive overview of the graphical simulation of cloth, we refer the reader to the surveys [Choi and Ko, 2005; Magnenat-Thalmann and Volino, 2005; Zhu et al., 2004; Ng and Grimsdale, 1996] and the text of House and Breen [2000].

In general, computer graphics methods have formulated and discretized separately the membrane and bending modes of deformation (an approach mirrored by treatments of thin plates in the finite element community [Zienkiewicz and Taylor, 2000; Hughes, 1987]). Although bending forces are much weaker than stretching forces, their interaction with membrane forces determines the shape of folds and wrinkles that we associate with garments and other thin flexible bodies [Thomaszewski and Wacker, 2006; Ciarlet, 2000].

Membrane modes Feynman [1986] introduced discretizations of membrane energy for computer animation, and other models followed [Terzopoulos et al., 1987; Haumann, 1987; Carignan et al., 1992; Baraff and Witkin, 1998]. Any good treatment of membrane response satisfies the assumption of small in-plane strains. To achieve this, some practitioners use a stiff membrane response, whereas others prefer a reduced stiffness combined with a strain limiting procedure [Provot, 1995; Bridson et al., 2002; Desbrun et al., 1999b]. We view the planar strain (membrane) forces as a mechanism (in the spirit of Lagrange multipliers or penalty forces) that enforces an isometry constraint on the manifold of permissible deformations (see §6.3.3 for a more detailed discussion of constraints). In this light, our concern here is impartial to the choice of membrane model.

Bending modes Discretizations of bending may be loosely categorized as involving (a) mass-spring particle systems [Breen et al., 1994; Volino et al., 1995; Choi and Ko, 2002], (b) linear or higher-order finite-elements [Etzmuss et al., 2003; Cirak et al., 2000], or (c) dihedral angles [Baraff and Witkin, 1998; Bridson et al., 2003; Grinspun et al., 2003]. For an overview of bending models in graphics, refer to the survey of Thomaszewski and Wacker [2006].

For comparison, we implemented the widely adopted, "nonlinear hinge" bending model of Baraff and Witkin [1998]. To our knowledge, all the commonly used models are either inherently nonlinear in positions—they usually involve expressions in terms of lengths or angles—or they assume small displacements and factor out a rigid motion at each time step. The bulk of our performance advantage results from the observation that IBM's forces are naturally linear in position, without assuming small displacements or disposing of rotational invariance.

2.3.2 Discrete mean curvature

Several discrete versions of mean curvature have been brought forward recently. We review some of the most important models for triangulated meshes here. Observe that in all these models, discrete curvature is an *integrated quantity* (see §2.2).

Edge-based Edges are the atomic entities across which a triangulated surface may be considered flexible. In graphics, a common model for integrated discrete mean curvature associated with an edge e is given by the product of dihedral angle and edge length, $H(e) = (\pi - \theta_e) \cdot |\mathbf{e}|$. This version was derived in the context of geometric measure theory [Cohen-Steiner and Morvan, 2003] and found high-quality applications in modeling thin shells [Grinspun et al., 2003]. Bridson et al. [2003] used a similar model, replacing θ_e by $\cos(\theta_e/2)$. Obviously, these two versions agree (up to a factor of two) in the limit of small angles between normals ($\theta_e \to \pi$). Coming from a different direction, Bobenko and Schröder [2005] introduced a version of integrated mean curvature *squared*, based on the intersection angles of circumcircles. This model is a priori a functional version of H^2, as opposed to starting out with a functional representation of H—a nontrivial insight. Note that this is done in a functional perspective, so it does not a priori give a model of discrete mean curvature itself. However, this concept is still found to be intimately linked to the previous two in the limit of the planar case [Bobenko and Schröder, 2005].

Vertex-based Vertex-centered discrete mean curvatures are obtained by choosing one of the above edge-based models and summing the contributions from all edge-based curvatures of incident edges,

$$\mathbf{H}(v) = \frac{1}{2} \sum_{e \ni v} \mathbf{H}(e).$$

The factor $1/2$ is due to the fact that we treat integrated quantities here, and each triangle in the vertex star of a vertex v is seen by exactly two edges.

2.4 Discrete isometric bending model

Our guiding principles for discretization are: (a) to maintain the geometric description of the continuous operators (i.e., mean curvature and Laplacian), (b) to maintain the quadratic structure of bending energy when expressed in these terms, and (c) to preserve key symmetries of the continuous energy, namely, invariance under rigid transformations and uniform scaling.

We provide the necessary background on conforming and nonconforming finite elements. The key difference is in the degrees of freedom (DOFs): conforming elements have their DOFs at vertices, and nonconforming ones have their DOFs at edges. Moreover, we relate the functional perspective of the linear model for the mean curvature vector to a discrete Laplacian. Finally, we derive a representation of the mean curvature normal as a (3-valued) function and write the discrete energy in terms of this representation.

2.4.1 Conforming setting

Our surfaces are discretized by *piecewise linear* (PL) elements, consisting of vertices, edges, and triangular faces. Along this line, it is natural to discretize functions over such surfaces in a piecewise linear fashion, with degrees of freedom assigned to vertices. The embedding of a mesh $\mathbf{x} : S \to \mathbb{E}^3$ becomes a (3-valued) PL function,

$$\mathbf{x} = \sum_p \mathbf{x}_p \cdot \Phi_p \ ,$$

where $\mathbf{x}_p$ are vertex positions, and Φ_p are nodal Lagrange basis functions taking value 1 at vertex p and 0 at all other vertices.

Using PL functions, it naturally follows to discretize the Laplace-Beltrami according to a finite element (FEM) representation. Disregarding boundary vertices, this reads

$$\Delta u(p) = -\int_S \langle \nabla u, \nabla \Phi_p \rangle \mathrm{d}A \ ,$$

where u is some linear combination of the conforming basis functions, $u = \sum u_p \Phi_p$, associated with interior (i.e., non-boundary) vertices p of the surface S. This exactly corresponds to the *cotan representation* [Pinkall and Polthier, 1993],

$$\Delta u(p) = -\frac{1}{2} \sum_q (\cot \alpha_{pq} + \cot \beta_{pq})(u(p) - u(q)),$$

where the sum is taken over all vertices q sharing an edge with p, and α_{pq} and β_{pq} are the opposite angles to the edge $\overline{pq}$ in the two triangles sharing that edge. It is important to note that Δu is no longer a function since differentiating a PL function twice takes values

in the space of distributions. Instead, according to (2.1), $\Delta u(p) = (\Delta u)(\Phi_p)$ represents the value of the *functional* Δu applied to the Lagrange basis function Φ_p.

The conforming *stiffness matrix* $\mathbf{L}_{pq}$ and *mass matrix* $\mathbf{M}_{pq}$ are given by

$$\mathbf{L}_{pq} = \int_S \langle \nabla \Phi_p, \nabla \Phi_q \rangle \mathrm{d}A \qquad \mathbf{M}_{pq} = \int_S \Phi_p \cdot \Phi_q \mathrm{d}A \ .$$

Dealing with zero-boundary conditions throughout, the number of rows of $\mathbf{L}_{pq}$ corresponds to interior vertices and its columns to all vertices of the mesh. The mass matrix $\mathbf{M}_{pq}$ serves as the metric for the L^2 inner product of two functions in basis representation (see §2.4.3). It is square, symmetric and invertible, its dimension being the number of interior vertices.

2.4.2 Nonconforming setting

In the nonconforming setting, the DOFs are associated with edges. A basis is given by PL mid-edge functions Φ_e, which are obtained by putting the value 1 at the midpoint of edge e and the value 0 at all other edge midpoints, followed by PL interpolation over triangles [Crouzeix and Raviart, 1973]. Unlike the conforming functions Φ_p, the mid-edge functions Φ_e are *not continuous*. Instead they are merely mid-edge continuous.

Note that the space of conforming elements is a subspace of the space of nonconforming elements. Thus, the *continuous* embedding $\mathbf{x}$ of the mesh can be written in the nonconforming basis,

$$\mathbf{x} = \sum_e \mathbf{x}_e \cdot \Phi_e \ ,$$

using a basis change from conforming to nonconforming elements. This basis change is simply obtained by a matrix whose rows correspond to the number of edges and whose columns correspond to the number of vertices. The only nonzero entries of this matrix take the value $1/2$, where each edge sees exactly its two boundary vertices (see §2.4.4).

The discrete Laplacian associated with the nonconforming elements is edge-based,

$$\Delta u(e) = -\int_S \langle \nabla u, \nabla \Phi_e \rangle \mathrm{d}A,$$

where u is some linear combination of nonconforming basis functions, $u = \sum u_e \Phi_e$, associated with *interior edges*. Again, the object Δu is a functional rather than a function.

The nonconforming stiffness matrix and mass matrix are given by

$$\mathbf{L}_{e_i e_j} = \int_S \langle \nabla \Phi_{e_i}, \nabla \Phi_{e_j} \rangle \mathrm{d}A \qquad \mathbf{M}_{e_i e_j} = \int_S \Phi_{e_i} \cdot \Phi_{e_j} \mathrm{d}A \ .$$

The number of rows of $\mathbf{L}_{e_i e_j}$ corresponds to interior edges and the columns to all edges of the mesh. The mass matrix $\mathbf{M}_{e_i e_j}$ is again square, symmetric and invertible, and its dimension is the number of interior edges.

2.4.3 Mean curvature function

Recall that in the continuous setting, the mean curvature normal is intimately related to the Laplacian: $\mathbf{H} = \Delta \mathbf{x}$. Given a *discrete* Laplacian (such as obtained by finite elements), we can *define* $\Delta \mathbf{x}$ as a discrete mean curvature *functional*. Using conforming (resp. nonconforming) elements, at interior vertices (resp. edges) one obtains

$$\mathbf{H}(p) = \Delta \mathbf{x}(p) \qquad (\text{resp. } \mathbf{H}(e) = \Delta \mathbf{x}(e)) \ .$$

Though the functional viewpoint of the mean curvature normal is indispensable for a mathematical analysis, such as treating convergence [Hildebrandt et al., 2005], for the bending energy defined in (2.1), we must find a representation of the mean curvature normal as a (3-valued) PL *function* over S. Given a PL function u that vanishes at the boundary, this requires that the L^2 inner product of $\mathbf{H}_d$ with u gives the same value as the functional $\mathbf{H}$ applied to u. By definition, this amounts to finding the unique conforming (resp. nonconforming) function $\mathbf{H}_d$ with zero boundary conditions that satisfies for each interior vertex p (resp. edge e)

$$\mathbf{H}(p) = \int_S \mathbf{H}_d^c \cdot \Phi_p \qquad \left(\text{resp. } \mathbf{H}(e) = \int_S \mathbf{H}_d^{nc} \cdot \Phi_e \right) \ .$$

The representation of $\mathbf{H}_d$ in terms of basis functions reads

$$\mathbf{H}_d^c = \sum_p (\mathbf{H}_d^c)_p \cdot \Phi_p \qquad \left(\text{resp. } \mathbf{H}_d^{nc} = \sum_e (\mathbf{H}_d^{nc})_e \cdot \Phi_e \right) ,$$

summing over all interior vertices (resp. interior edges). In both cases, the result may be written as $\mathbf{H}_d = (\mathbf{M}^{-1}\mathbf{L})\mathbf{x}$, where the vector $\mathbf{H}_d$ corresponds to pointwise (average, not integrated) mean curvature normals and $\mathbf{L}$ and $\mathbf{M}$ are the conforming (resp. nonconforming) stiffness and mass matrix.

2.4.4 Discrete bending energy

Given a discrete mean curvature normal, $\mathbf{H}_d$, we can—in perfect analogy to the smooth case—define the discrete IBM

$$\begin{aligned} E_b(\mathbf{x}) &= \frac{1}{2}\int_S \langle \mathbf{H}_d, \mathbf{H}_d \rangle \mathrm{d}A \\ &= \frac{1}{2}(\mathbf{M}^{-1}\mathbf{L}\mathbf{x})^T\mathbf{M}(\mathbf{M}^{-1}\mathbf{L}\mathbf{x}) \end{aligned}$$

which leads to

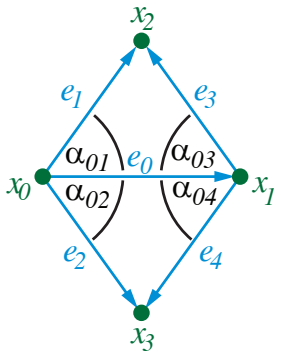
$$\boxed{E_b(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T(\mathbf{L}^T\mathbf{M}^{-1}\mathbf{L})\mathbf{x}\ .}$$

The form of $E_b(\mathbf{x})$ reveals two crucial properties: (i) it is invariant under rigid transformations and uniform scaling, and (ii) it is quadratic in vertex positions in the class of isometric deformations, which include arbitrarily large bending away from the initial shape. A third property is that the energy Hessian $\mathbf{K} = \mathbf{L}^T\mathbf{M}^{-1}\mathbf{L}$ allows for a geometric interpretation: $\mathbf{L}$ is invariant under *conformal* deformations (it only depends on *angles*), whereas the entries of $\mathbf{M}$ correspond to *area*.

The concrete value of this energy will depend on which version of $\mathbf{H}_d$, conforming or nonconforming, is used. We use the *nonconforming* version since this associates bending to *edges*, with the corresponding hinge-stencil used in popular nonlinear models [Baraff and Witkin, 1998]. The idea of employing nonconforming elements for treating discrete operators has previously been employed for a theory of discrete minimal surfaces [Polthier, 2002]. The *conforming* case of bending energy was treated in the exact same manner as outlined above by Clarenz et al. [2004] and used therein for Willmore flow.

We briefly lay out the complete set of tools required to assemble the hinge-stenciled IBM. The Hessian, $\mathbf{K} = \mathbf{L}^T\mathbf{M}^{-1}\mathbf{L}$, is assembled in the usual manner of finite-element stiffness

matrices [Zienkiewicz and Taylor, 2000], i.e., by considering contributions from each *local* stiffness matrix, $\mathbf{K}_i$, centered about edge e_i with stencil consisting of the incident triangle(s), $t_0,\ t_1$, and vertices, $v_0,\ v_1,\ v_2,\ v_3$. With reference to the illustration,



$$\mathrm{A} = \frac{1}{2}\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix},$$

we build $\mathbf{K}_i = \mathrm{A}^T(\mathbf{L}_i^T\mathbf{M}_i^{-1}\mathbf{L}_i)\mathrm{A}$, where $\mathbf{M}_i = \frac{1}{3}(\mathrm{area}(t_0) + \mathrm{area}(t_1))$ is the *scalar* entry of the mass matrix $\mathbf{M}$ associated to e_i, A is the change of basis matrix taking vertex DOFs to edge DOFs, and $\mathbf{L}_i$ corresponds to the i^{th} row of the edge-based Laplacian,

$$\mathbf{L}_i = 2(c_{01} + c_{02} + c_{03} + c_{04}, -c_{01}, -c_{02}, -c_{03}, -c_{04}),$$

where $c_{jk} = \cot\alpha_{jk} = \cot\angle e_j, e_k$. This can be simplified to

$$\mathbf{K}_i = \frac{3}{(\mathrm{area}(t_0) + \mathrm{area}(t_1))}\widetilde{\mathbf{L}}_i^T\widetilde{\mathbf{L}}_i\ ,$$

where $\widetilde{\mathbf{L}}_i$ is the row vector

$$\widetilde{\mathbf{L}}_i = (c_{03} + c_{04}, c_{01} + c_{02}, -c_{01} - c_{03}, -c_{02} - c_{04})\ .$$

The *local* energy is obtained by $E_d(i) = \frac{1}{2}(x_0, x_1, x_2, x_3)\mathbf{K}_i(x_0, x_1, x_2, x_3)^T$; the *global* (total) energy of the system is obtained by summing over all local contributions corresponding to interior edges e_i.

2.5 Application to the bending of cloth and plates

2.5.1 Numerical treatment

Whether in the high-fidelity or interactive setting, rapid simulation requires an efficient numerical integrator for second order initial value problems (IVPs) of the form

$$\mathbf{M}\ddot{\mathbf{x}}(t) = \mathbf{F}(t, \mathbf{x}(t), \dot{\mathbf{x}}(t)) \ , \tag{2.3}$$

where $\mathbf{M}$ is the (physical) mass matrix; $\mathbf{F}(t, \mathbf{x}(t), \dot{\mathbf{x}}(t))$ are forces depending on time, position, and velocity; and the initial position and velocity of the cloth are prescribed [Hauth, 2004]. The temporal discretization of this system is a well-studied and active area of applied mathematics and computational mechanics, with a host of attendant methods. For treatments tailored to cloth simulation we refer the reader to [Baraff and Witkin, 1998; Etzmuss et al., 2000; Hauth and Etzmuss, 2001; Choi and Ko, 2002; Ascher and Boxerman, 2003; Bridson et al., 2003; Hauth et al., 2003; Boxerman and Ascher, 2004; Hauth, 2004]).

As a general observation, each force may be treated *explicitly* or *implicitly*. An explicit treatment requires the (possibly repeated) evaluation of the force per discrete time step; an implicit method requires additionally the (possibly repeated) evaluation of the force Jacobian per discrete time step. In the case of a conservative force, such as elastic bending, the force Jacobian is minus the energy Hessian. In the case of a dissipative Rayleigh force, such as damping of the bending modes, the Jacobian is expressed as a linear combination of the physical mass and energy Hessian.

We implemented both the implicit solver framework of Baraff and Witkin [1998] as well as a simple explicit Euler solver. Our choice of solvers is motivated by a desire to estimate profitability of incorporating IBM whether or not a framework requires bending force Jacobians. The test framework incorporates: (a) the usual constant strain linear finite element for membrane response [Zienkiewicz and Taylor, 2000; Hughes, 1987], which computationally is at least as costly as the membrane model of Baraff and Witkin [1998]; (b) the robust collision detection and response methods of Bridson et al. [2002], augmented with k-DOP trees [Klosowski et al., 1998]; and (c) the PETSc solver library [Balay et al., 2009]. Further

Draping problem		regular mesh (resolution, in no. vertices)				irregular mesh (resolution, in no. vertices)			
		400	1600	6400	25600	450	2100	6500	22500
Gradient	nonlinear hinge	0.937	3.45	16.4	66.6	1.10	5.43	17.6	67.8
cost (ms)	quadratic IBM	0.081	0.338	2.19	9.15	0.098	0.494	2.32	9.68
Hessian	nonlinear hinge	12.8	54.2	218	890.	15.2	77.2	246	888
cost (ms)	quadratic IBM	0.237	0.963	3.87	15.7	0.266	1.28	3.99	13.6
Explicit step	nonlinear hinge	3.81	6.64	27.5	112.	2.16	9.53	31.4	140.
cost (ms)	quadratic IBM	2.63	2.90	11.9	48.8	0.964	4.35	15.2	76.5
Implicit step	nonlinear hinge	28.6	138	470.	1730	33.9	219	557	1880
cost (ms)	quadratic IBM	11.0	62.7	168	505	13.6	103	219	612

Flag problem		regular mesh (resolution, in no. vertices)				irregular mesh (resolution, in no. vertices)			
		400	1600	6400	25600	450	2100	6500	22500
Gradient	nonlinear hinge	0.975	3.99	16.0	64.0	1.10	5.43	17.8	68.7
cost (ms)	quadratic IBM	0.085	0.341	2.14	8.75	0.099	0.490	2.31	9.28
Hessian	nonlinear hinge	13.4	54.8	212	849	15.2	77.4	247	887
cost (ms)	quadratic IBM	0.251	0.974	3.79	14.99	0.267	1.30	3.96	13.7
Explicit step	nonlinear hinge	1.73	7.05	27.7	112.	1.97	9.80	32.7	134
cost (ms)	quadratic IBM	0.780	3.26	13.3	53.4	0.900	4.54	16.1	70.0
Implicit step	nonlinear hinge	27.6	106	420.	1680	33.5	155	513	1880
cost (ms)	quadratic IBM	9.53	32.9	127	490	12.5	50.4	166	608

Table 2.1: We compare the computational cost per time step of our quadratic energy to the popular nonlinear hinge energy. Costs reported include the time required for collision detection and response. These tests were conducted on a single process, Pentium D 3.4GHz, 2GB RAM.

optimization of our collision detection (using, e.g., [Govindaraju et al., 2005]) is likely to reduce the overhead of collision computations for large meshes. Our results indicate that IBM accelerates both explicit and implicit treatments of bending. For a more detailed comparison, see Table 2.1.

Implementing the IBM is straightforward, requiring the (precomputed) assembly of the Hessian matrix (coefficients of $\mathbf{L}^T\mathbf{M}^{-1}\mathbf{L}$ are given in §2.4.4), and a matrix-vector multiplication to compute the forces $(-\mathbf{L}^T\mathbf{M}^{-1}\mathbf{L})\mathbf{x}$. By contrast, an efficient implementation of a nonlinear model such as a hinge spring requires both significant (human and computer) gradient calculations, or leads to adoption of costly automatic-differentiation techniques [Grinspun et al., 2003]. For comparison of computational efficiency, our implementation of the hinge forces and Jacobians was hand tuned and not based on automated methods.

2.5.2 Cloth simulation

We compare the computational cost and visual quality of the IBM to the widely used "hinge" bending model [Baraff and Witkin, 1998; Bridson et al., 2003; Grinspun et al., 2003], for regular and irregular meshes ranging from 400 to 25600 vertices, on a draping problem as

well as a dynamic billowing flag. Figures 2.1–2.2 provide a qualitative point of comparison between IBM and the nonlinear hinge model.

Replacing the nonlinear hinge model with quadratic IBM reduces bending force computation cost by seven- to eleven-fold. Where a force Jacobian is required, IBM's constant Hessian is precomputed once and stored in a sparse matrix datastructure, eliminating the computational cost of bending Jacobian assembly. Of course, the actual net performance improvement in any given application depends on the fraction of total computation associated to bending.

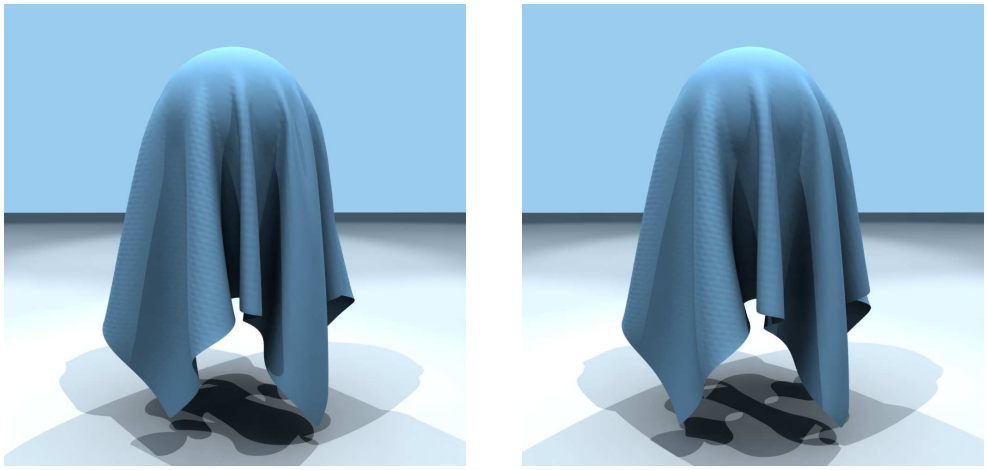


Figure 2.2: Final rest state of a cloth draped over a sphere, for *(left)* the isometric bending model and *(right)* the widely adopted nonlinear hinge model.

Draping cloth We simulated the draping of a square sheet over a sphere. As shown in Figure 2.2, the draped cloths are qualitatively similar in their configuration and distribution of wrinkles and folds. Since only the final draped state was important, we introduced a dissipative Rayleigh force allowing for larger time steps [Hughes, 1987]. For this example we might also consider a quasistatic method, as considered by Teran et al. [2005]. Since quasistatics requires repeated evaluation of forces and their Jacobian, the proposed bending model will also be profitable in the quasistatic setting. Together Figures 2.2–2.3 capture the behavior of IBM under a range of bending stiffnesses.

Billowing flag We simulated the dynamics of a flag under wind. As shown in the accompanying movie (see also Fig. 2.1) the motion of the flag is qualitatively unaffected when we substitute the more economical quadratic bending energy. We modeled wind by a constant homogeneous velocity field, with force proportional to the projection of the wind velocity onto the area-weighted surface normal. For more sophisticated effects of wind fields on cloth see [Keckeisen et al., 2004]. Purely for contrast with the draping example, we did not introduce any specific damping forces, although the implicit Euler method is known to exhibit numerical damping.

IMEX methods We stress that the proposed model is not specialized to our test framework. Consider for example the integration of the IBM into the framework described by Bridson et al. [2003], which treats elastic (position-dependent, velocity-independent) forces explicitly and treats damping forces implicitly. With IBM both force and Jacobian computation is greatly reduced; therefore explicit, implicit, and semi-implicit treatments are all prime candidates. Bridson et al. [2003] describe a very efficient matrix-free solver strategy, involving as few as one to two conjugate gradient iterations: this heightens the relevance of fast (elastic and damping) bending force computations, since with fewer conjugate gradient iterations, the balance of computation shifts further to the computation of the cached nonlinear position-dependent terms as well as the algebraic system's right hand side. In incorporating IBM in this context, the requisite matrix-vector multiplication may be distributed, $(\mathbf{K}_m + \mathbf{K}_b)\mathbf{x} = \mathbf{K}_m\mathbf{x} + \mathbf{K}_b\mathbf{x}$, where the precomputed $\mathbf{K}_b$ is a multiple of the constant bending Jacobian, and $\mathbf{K}_m$ is the membrane Jacobian.

In any practical application, the final decision on incorporating a proposed technique hinges on the visual quality, performance benefit, and implementation cost of the method. We have presented a model for bending whose visual quality compares favorably with the widely adopted nonlinear hinge model and which involves simply computing the entries of a sparse matrix. This computation is straightforward, and incorporation of the proposed model is a minimally invasive task. Given these considerations, we suggest that it will be simple to evaluate the efficacy of the proposed model within the reader's preferred cloth simulation framework.

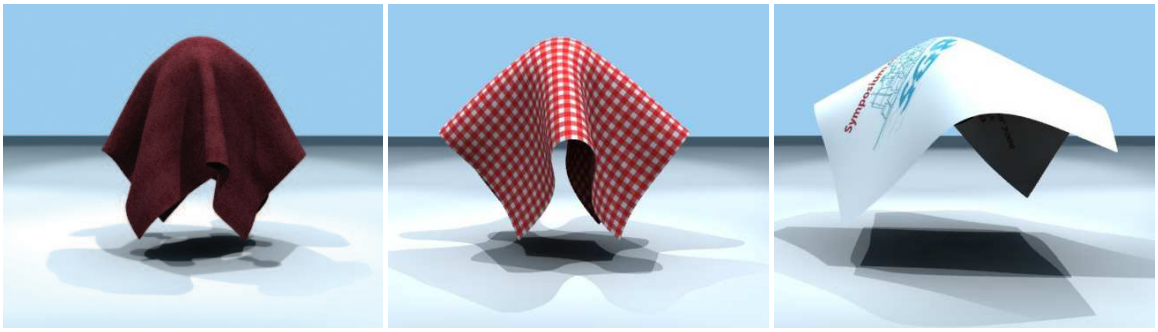


Figure 2.3: Deformations are primarily characterized by the ratio of bending to membrane stiffness. Despite having forces linear in positions, the quadratic bending model is valid over a broad range of stiffness values. Shown here *(left to right)*: $10^{-5} : 1$, $10^{-3} : 1$, and $10^{-2} : 1$.

2.5.3 Geometric modeling

As noted by Bobenko and Schröder [2005], immersions of a surface that minimize its *Willmore energy*, given by

$$E_W = \int_S (H^2 - K)\mathrm{d}A \;=\; \frac{1}{4}\int_S (\kappa_1 - \kappa_2)^2 \mathrm{d}A \;,$$

are of interest in a range of areas, including the study of conformal geometry [Blascke, 1929; Willmore, 2000], physical modeling of fluid membranes [Canham, 1970; Helfrich, 1973], and, our focus in this example, geometric modeling. For surfaces without boundary and surfaces whose boundary is fixed up to first order, the Willmore functional is variationally equivalent to the functional (2.1). The corresponding geometric flow

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) = -\nabla E_b(\mathbf{x}), \tag{2.4}$$

has spurred many applications for surface fairing and restoration [Yoshizawa and Belyaev, 2002; Bobenko and Schröder, 2005; Clarenz et al., 2004; Schneider and Kobbelt, 2001]. One of the earliest examples of discrete Willmore flow was implemented in Ken Brakke's *Surface Evolver* [Hsu et al., 1992]. This work uses the cotangent formula to discretize mean curvature and an explicit time stepper for integrating the flow. Yoshizawa and Belyaev [2002] discretize explicit expressions for forces obtained from the variation of the Willmore energy and use the cotangent formula to discretize both the mean curvature and Laplace-Beltrami operators in the force expression. Explicit time-stepping is used and an additional

tangential component is added to the forces to improve the quality of the evolving mesh. Clarenz et al. [2004] discretize the variation of the Willmore energy in terms of finite elements and treat the corresponding L^2-flow by a coupled system of second order equations. Finally, Bobenko and Schröder [2005] introduced a circle-based conformally invariant Willmore energy discretization in their flow formulation.

In geometric hole filling applications (see Fig. 2.5) the flow is integrated to its stationary limit. In smoothing applications (see Figure 2.4) the flow is integrated over a prescribed duration, and longer integration times smooth progressively coarser spatial frequencies. Thus the flow duration effectively controls a trade-off between keeping and discarding the initial geometry in exchange for the "smoothest" immersion of a given genus satisfying prescribed G^1 boundary conditions.

Discrete IBM in Willmore solver Our discrete IBM can be readily used for an efficient Willmore solver. We implemented an implicit integrator for the geometric flow problem given by (2.4), using a Newton-like method to find a root of

$$\mathbf{G}(\mathbf{x}_{k+1}) = \mathbf{x}_{k+1} - \mathbf{x}_k - h\mathbf{f}(\mathbf{x}_{k+1}) \ .$$

Note that in this case, there is no stiff membrane term preventing in-plane deformation of the surface, so $\mathbf{f}(\mathbf{x}_{k+1})$ requires evaluating the full nonlinear gradient of E_b. In addition, standard Newton's method requires the Jacobian of $\mathbf{G}$, given by $\mathbf{J} = \nabla\mathbf{G} = \mathbf{I} + h\mathrm{Hess}(E_b)(\mathbf{x})$, which involves computing the full nonlinear Hessian of E_b. However, replacing the Jacobian, $\mathbf{J} = \nabla\mathbf{G}$, with an approximation, $\tilde{\mathbf{J}} \approx \mathbf{J}$, does not change the roots of the method, only the speed and radius of convergence. Thus, we approximate the full Jacobian by replacing the nonlinear Hessian with IBM's constant Hessian. Since $\tilde{\mathbf{J}}$ is constant, we need to compute it only *once* at the beginning of time, allowing us to pre-factor (symbolically) the Jacobian matrix before the flow begins. This results is a formulation that facilitates rapid computation at nearly interactive rates while visually maintaining good surface quality (see Figs. 2.4 and 2.5).

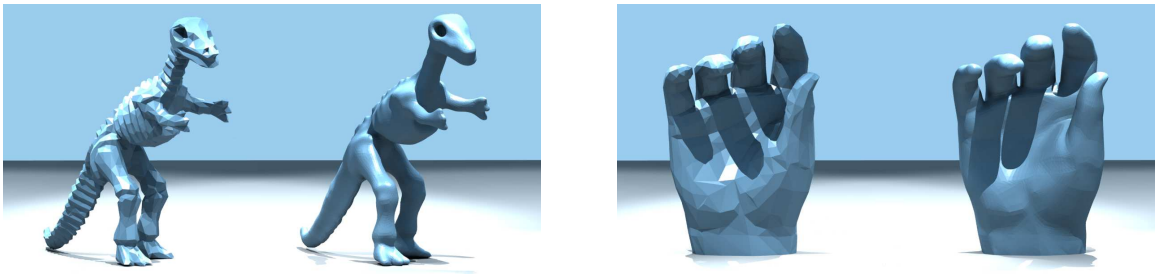


Figure 2.4: Initial and final frames of Willmore flow applied to smooth *(left)* a 44928 triangle dinosaur and *(right)* a 24192 triangle hand at interactive rates. 16 smoothing steps require a total of 7.47s and 4.42s, with one-time factorization costing 8.77s and 5.31s, for the dinosaur and the hand, respectively. Images rendered with flat shading.

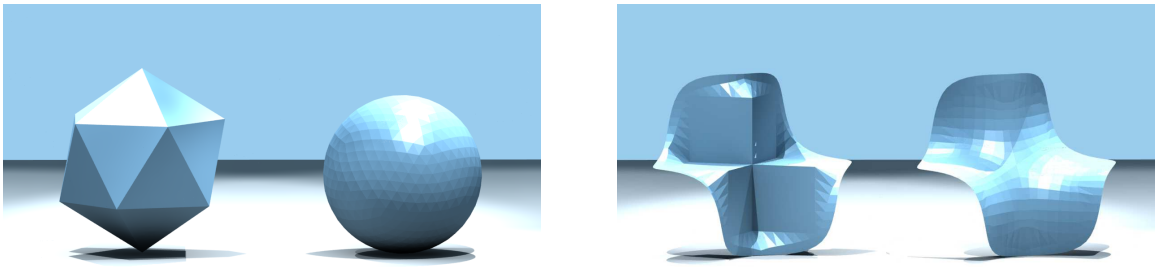


Figure 2.5: Initial and final frames of Willmore flow applied to solve two tasks posed by Bobenko and Schröder [2005]. *(left)* Smoothing a 4-times subdivided icosahedron into a sphere and *(right)* a hole filling problem. The sphere converged in 120ms (12ms $\times$ 10 smoothing steps), with 200ms for Hessian prefactorization. The hole filling problem required 640ms after 120ms for prefactorization. Rendered with flat shading.

Implementation Our implementation uses semi-implicit backwards Euler time integration using as a linear solver the PARDISO [Schenk and Gärtner, 2006] LL^T direct solver. We report computation times for a single process running on a 2GHz notebook with 2GB RAM.

One avenue for extending our application would be to incorporate a reparametrization flow, such as the special tangent speed component that Yoshizawa and Belyaev [2002] used to improve the quality of the evolving mesh.

Chapter 3

Adapted framed curves

Simulating the dynamics of long, thin objects such as elastic rods (e.g., hair, rope, and sutures) or viscous threads (e.g., honey poured from a bottle) using general volumetric methods is difficult due to the large ratio of length to cross-sectional width of these objects. Volumetric methods can result in badly shaped or very small elements, overly stiff equations, and instability in the associated numerics. By contrast, both viscous threads and elastic rods are amenable to a reduced coordinate model based on an *adapted framed curve*, which consists of a space curve and an adapted, orthonormal frame. This alleviates the ill-conditioning inherent in the volumetric approaches but leaves open the question of how best to represent an adapted framed curve from a computational perspective.

The purpose of the present chapter is twofold. First, we provide background material on the geometry of an adapted framed curve in the smooth setting, arriving at the definitions for the kinematic quantities (i.e., velocities and strains) needed for a treatment of the dynamics of elastic rods and viscous threads. Second, we define a discrete adapted framed curve along with its associated kinematic quantities, with the emphasis being on how the coordinate representation affects the computational efficiency of the model. Because our ultimate goal is to apply the formulation presented in this chapter to simulating elastic rods and viscous threads, we seek a representation that minimizes the expense of computing these kinematic quantities, along with their gradients and Hessians. We present a development

of the discrete setting alongside the smooth setting. The discrete picture not only helps to spell out the actual implementation, but it often simplifies and provides intuition for the corresponding smooth derivation.

3.1 Related work

The study of adapted framed curves has a long history in differential geometry [Frenet, 1847; Serret, 1851]. Standard analysis focuses on the Frenet frame to establish the kinematic quantities associated to space curves that are invariant under rigid motions [do Carmo, 1976]. More recently, Bishop [1975] showed that other framings can be used and introduced the parallel transport frame (referred to here as the space-parallel frame), which, unlike the Frenet frame, is free of twist.

Adapted framed curves have applications in a variety of different fields. For example, they are used in roller coaster design [Kecskeméthy and Tändl, 2006]. For visualization purposes, they can be used to construct tubes, ribbons, and extrusions [Hanson, 2006]. In addition, adapted framed curves serve as the geometric basis for many physical systems, including describing the configuration of an elastic rod [Kirchhoff, 1859], the motion of a smoke ring [Hasimoto, 1971], and the spinning of a top [Bobenko and Suris, 1999].

For representing adapted framed curves, many choices about, ranging from reduced to maximal coordinates. Reduced coordinates, such as curvature and twist degrees of freedom [Bertails et al., 2006] or joint angles in a rigid body chain [Hadap, 2006], offer a compact representation that disallows for extension of the centerline and shear of the material frame, but they require integrating an ordinary differential equation over the length of the curve in order to recover the position of the centerline. On the other hand, maximal coordinates offer an explicit centerline representation, making the integration unnecessary. The material frame can be encoded in a maximal coordinate representation using, e.g., quaternions coupled to the centerline [Grégoire and Schömer, 2007]. This representation allows for both extensible deformations and shear of the material frame.

The hybrid approach that we use, called the curve-angle representation, consists of an ex-

plicit centerline but encodes the material frame using a single angle by relating it to a reference frame at each point on the centerline. Within this representation, the fundamental question is how to define the reference frame, and various options have been explored in previous work. Goldstein and Langer [1995] observed that using the space-parallel (Bishop) frame simplifies both the analytical formulation and the numerical implementation of the dynamics of an uncurved elastic rod with symmetric cross-section. Theetten et al. [2007] propose the use of the Frenet frame for simulating elastic rods, which has the advantage over the Bishop frame that its support at each point is local (i.e., the frame at each point only depends on a small neighborhood around the point) but is undefined for straight segments of the curve and changes non-smoothly at points of inflection (see, e.g., [Hanson, 2006]). The time-parallel reference frame, which we advocate, combines the benefits of these two cases by offering a frame with a locally supported stencil while avoiding issues associated with straight segments and inflection points. In concurrent work, Kaldor et al. [2010] also propose the use of the time-parallel reference frame for representing the configuration of an adapted framed curve and present an efficient contact handling algorithm for simulating character-scale, yarn-based cloth. While we focus on a compact stencil for the representation with the ultimate goal of using this in an implicit framework for simulation of dynamics, they call attention to the fact that the local support of the stencil allows one to easily parallelize force computations, resulting in an efficient model even for explicit time integration, which does not require Hessian computations. In the finite element community, Boyer and Primault [2004] also discuss reference frames for adapted framed curves but avoid parametrizing the configuration by advocating the use of a Lie group integrator for simulating elastic rods.

When using a representation built on parallel transport for simulating dynamics, one must consider the variation of parallel transport and a related quantity called holonomy for computing forces and Jacobians. The relation of holonomy to variations of the centerline presented in §3.5 can be viewed as an extension of the Călugăreanu-White-Fuller theorem [Fuller, 1971] for discrete curves. Fuller [1978] noted that this theorem is applicable to the equilibria of closed elastic rods with isotropic cross-sections. Our development extends to the case of open boundaries and anisotropic cross-sections.

3.2 Definition of adapted framed curves

The geometry of an elastic rod or viscous thread is conveniently described by an adapted framed curve. Although this insight is not new [Kirchhoff, 1859], we will show that not all representations for adapted frames are created equal when it comes to numerical performance and ease of implementation.

3.2.1 Smooth setting

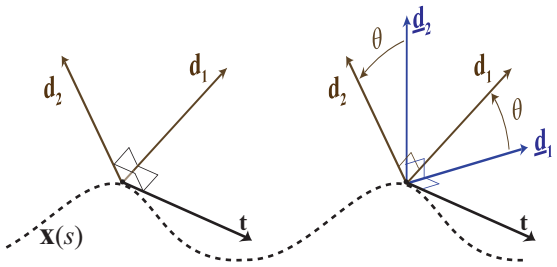


Figure 3.1: *Left*: An adapted framed curve consists of a curve $\mathbf{x}(s)$ and an adapted material frame $\{\mathbf{d}_1(s), \mathbf{d}_2(s), \mathbf{t}(s) = \mathbf{x}'(s)\}$. *Right*: We encode the material frame by an angle of rotation $\theta(s)$ relative to the reference frame $\{\underline{\mathbf{d}}_1(s), \underline{\mathbf{d}}_2(s), \mathbf{t}(s)\}$.

A time-evolving adapted framed curve, $\Gamma = \{\mathbf{x}; \mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3\}$, consists of a *centerline* curve and *material frame*,

$$\mathbf{x}(s,t) \in \mathbb{R}^3 \quad \text{and} \quad [\mathbf{d}_1(s,t) \quad \mathbf{d}_2(s,t) \quad \mathbf{d}_3(s,t)] \in \mathrm{SO}(3) \ ,$$

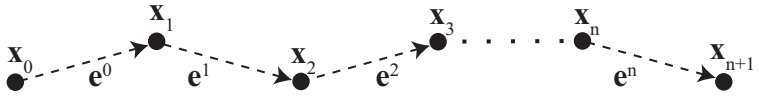
where s is the material coordinate, t is time, and $\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3$ are the three orthonormal director vectors of the material frame. The frame is *adapted* to the centerline, which disallows shearing of the material frame by requiring $\mathbf{d}_3$ to lie tangent to the centerline, $\mathbf{d}_3 \equiv \mathbf{t} = \mathbf{x}'/|\mathbf{x}'|$ (where the prime indicates differentiation with respect to the s coordinate). Thus, $\mathbf{d}_1$ and $\mathbf{d}_2$ span the plane normal to the centerline's tangent, called the cross-section, and are referred to as the material directors. We denote quantities associated to the *undeformed* configuration of the adapted framed curve with an overline. Further, we assume that s is an arc-length parametrization of the undeformed configuration, so that $\overline{\mathbf{x}}'(s,t)$ is unit length but $\mathbf{x}'(s,t)$ may not be unit length.

3.2.2 Discrete setting

We define a discrete adapted framed curve as a centerline given by a set of $n+2$ vertices (lower indices) and a set of $n+1$ adapted, orthonormal frames attached to each edge (upper indices),

$$\mathbf{x}_i \in \mathbb{R}^3 \quad \text{and} \quad \begin{bmatrix} \mathbf{d}_1^j & \mathbf{d}_2^j & \mathbf{d}_3^j \end{bmatrix} \in SO(3)$$

for $0 \leq i \leq n+1$ and $0 \leq j \leq n$. We define adapted in the discrete case to mean that $\mathbf{d}_3^j$ lies along the edge formed by the adjacent vertices, $\mathbf{e}^j = \mathbf{x}_{j+1} - \mathbf{x}_j$, and is thus given by $\mathbf{d}_3^j \equiv \mathbf{t}^j = \mathbf{e}^j / |\mathbf{e}^j|$. In the discrete setting, we distinguish between primal quantities, associated with vertices and decorated with lower indices, and dual quantities, associated with edges and decorated with upper indices. This diagram summarizes our indexing and notation conventions:



Before we discuss the coordinates we use to represent an adapted frame curve, we first define the purely geometric, kinematic quantities needed for dynamic simulation. Once we have these definitions, we will choose coordinates in §3.4 that most simplify the computation of these quantities.

3.3 Kinematics

The velocities and strains associated to the adapted framed curve are closely related, since the former depends on how the centerline and material frame change over time t, while the latter depends on how they change along arclength s, or space.

We differentiate between *pointwise* quantities associated to each point of the adapted framed curve and *integrated* quantities that correspond to an integral of the quantity over some domain, $\mathcal{D}$ (see §2.2). To convert an integrated quantity back to a pointwise one, we simply divide by the length, $|\mathcal{D}|$, of the domain of integration. In the discrete case, we consider

integrated quantities associated to edges and vertices. Thus, to convert these to pointwise quantities, we divide by edge lengths, $\left|\mathcal{D}^j\right| = \left|\mathbf{e}^j\right|$, or lengths of Voronoi regions,

$$l_i = |\mathcal{D}_i| = \frac{1}{2}\left(\left|\mathbf{e}^{i-1}\right| + \left|\mathbf{e}^i\right|\right) \ , \tag{3.1}$$

associated to vertices.

3.3.1 Centerline

In the smooth setting, the kinematics of the centerline is governed by its velocity, $\dot{\mathbf{x}}(s,t) = \partial\mathbf{x}/\partial t$, and deformation gradient, $\mathbf{x}'(s,t) = \partial\mathbf{x}/\partial s$. The deformation gradient is used to define the relative axial strain,

$$\varepsilon = \left|\mathbf{x}'\right| / \left|\overline{\mathbf{x}}'\right| - 1 \ . \tag{3.2}$$

In the discrete setting, we keep track of the velocities of the vertices, $\dot{\mathbf{x}}_i$, and we define the edge-based relative axial strain as

$$\varepsilon^j = \left|\mathbf{e}^j\right| / \left|\overline{\mathbf{e}}^j\right| - 1 \ . \tag{3.3}$$

Note that this expression corresponds to the *pointwise* strain along the edge.

3.3.2 Smooth material frame kinematics

The curvature vector, $\kappa\mathbf{n} = \mathbf{t}'$, measures the deviation of the centerline curve from being straight. Since this vector is orthogonal to the tangent, it can be expressed as

$$\boldsymbol{\kappa} = (\kappa_1, \kappa_2)^T = (\kappa\mathbf{n}\cdot\mathbf{d}_1, \kappa\mathbf{n}\cdot\mathbf{d}_2)^T \ , \tag{3.4}$$

referred to as the *material curvature*. The two components, κ_1 and κ_2, measure the bending of the centerline along $\mathbf{d}_1$ and $\mathbf{d}_2$, respectively.

Orthonormality of the material frame allows us to write the spatial derivative of the material directors as

$$\mathbf{d}_\alpha' = \boldsymbol{\Omega}\times\mathbf{d}_\alpha \ , \quad \text{where} \quad \boldsymbol{\Omega} = m\mathbf{t} + \mathbf{t}\times\mathbf{t}' \ . \tag{3.5}$$

The *Darboux vector*, $\mathbf{\Omega}(s,t)$, is the axial vector associated with the infinitesimal rotation of the frame, written as the sum of the *twist* of the material frame about the tangent

$$m = \mathbf{d}_1' \cdot \mathbf{d}_2 = (\mathbf{\Omega} \times \mathbf{d}_1) \cdot \mathbf{d}_2 \tag{3.6}$$

and the component needed to keep the frame adapted to the centerline, $\kappa\mathbf{b} = \mathbf{t} \times \mathbf{t}'$, referred to as the curvature binormal vector. We can rewrite (3.4) in terms of the curvature binormal vector as

$$\boldsymbol{\kappa} = (\kappa_1, \kappa_2)^T = (\kappa\mathbf{b} \cdot \mathbf{d}_2, -\kappa\mathbf{b} \cdot \mathbf{d}_1)^T \ , \tag{3.7}$$

which is the form we will use when defining the material curvature in the discrete setting.

In the smooth setting, the material frame evolves over time via

$$\dot{\mathbf{d}}_\alpha = \boldsymbol{\omega} \times \mathbf{d}_\alpha \ , \quad \text{where} \quad \boldsymbol{\omega} = \omega_3\mathbf{t} + \mathbf{t} \times \dot{\mathbf{t}} \ . \tag{3.8}$$

Note the space-time symmetry of the terms in (3.8) and (3.5): like the Darboux vector, the angular velocity $\boldsymbol{\omega}$ of the material frame can be decomposed into $\mathbf{t} \times \dot{\mathbf{t}}$, which is fully determined by the motion of the centerline, and the rotation rate of the material frame about the tangent, $\omega_3 = \dot{\mathbf{d}}_1 \cdot \mathbf{d}_2$.

3.3.3 Rotations and parallel transport

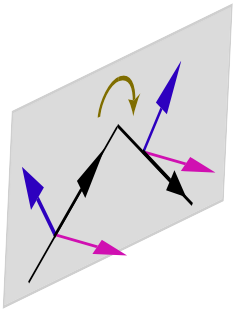
To transition to the discrete setting, we require a key concept: the *parallel transport* of a frame along the centerline is the minimum rotation needed to keep the frame adapted to the tangent (i.e., parallel transport does not rotate the frame about the tangent). Thus, m and ω_3 can be viewed as the deviation of the material frame away from parallel transport. This rotation is entirely determined by the centerline.

We define discrete parallel transport from a unit vector $\mathbf{a}$ to a unit vector $\mathbf{b}$ as the rotation that must be applied to $\mathbf{a}$ to align it with $\mathbf{b}$, which can be written as a rotation about the normal to the plane spanned by $\mathbf{a}$ and $\mathbf{b}$. This rotation can be computed as

$$\mathbf{P}_{\mathbf{a}}^{\mathbf{b}} = d\,\mathbf{I}_3 + \frac{\mathbf{c} \otimes \mathbf{c}}{1+d} + [\mathbf{c}]_\times \ , \tag{3.9}$$

where $\mathbf{c} = \mathbf{a} \times \mathbf{b}$, $d = \mathbf{a} \cdot \mathbf{b}$, and $\mathbf{I}_3$ is the 3×3 identity matrix.

We define discrete parallel transport at a vertex, $\mathbf{P}_i$, in analogy to the smooth case as the minimal rotation needed to align $\mathbf{t}^{i-1}$ to $\mathbf{t}^i$ (shown as the black vectors on the right), so that



$$\mathbf{P}_i \cdot \mathbf{t}^{i-1} = \mathbf{t}^i, \quad \mathbf{P}_i \cdot \left(\mathbf{t}^{i-1} \times \mathbf{t}^i\right) = \mathbf{t}^{i-1} \times \mathbf{t}^i \ .$$

Using the notation in (3.9), the discrete parallel transport is given by $\mathbf{P}_i = \mathbf{P}^{\mathbf{t}^i}_{\mathbf{t}^{i-1}}$. By convention, $\mathbf{P}_i$ is the identity if $\mathbf{t}^{i-1} = \mathbf{t}^i$, whereas $\mathbf{P}_i$ is not defined if $\mathbf{t}^{i-1} = -\mathbf{t}^i$. Note that, as in the smooth setting, parallel transport is entirely determined by the centerline.

3.3.4 Discrete material frame kinematics

In adapting the material frame, the discrete follows the smooth picture, now as a two-step process. Consider the tangent at two consecutive edges, $\mathbf{t}^{i-1}$ and $\mathbf{t}^i$. We first parallel transport the material frame, which aligns the tangents, and we then rotate by the angle m_i about $\mathbf{t}^i$ to align the material frames. This results in relating the material directors at the two adjacent edges via a *finite rotation*, $\mathbf{\Omega}_i$, so that $\mathbf{d}^i_\alpha = \mathbf{\Omega}_i \cdot \mathbf{d}^{i-1}_\alpha$ for $\alpha \in \{1, 2, 3\}$. Instead of replacing the finite rotation by an infinitesimal rotation (i.e., a skew-symmetric matrix), as often done in linearized elasticity, this geometrically nonlinear approach guarantees that our model remains objective (i.e., invariant under rigid motions).

The rotation $\mathbf{\Omega}_i$ can be decomposed into

$$\mathbf{\Omega}_i = \mathbf{R}\left(\mathbf{t}^i, m_i\right) \circ \mathbf{P}_i = \mathbf{P}_i \circ \mathbf{R}\left(\mathbf{t}^{i-1}, m_i\right) \ , \tag{3.10}$$

where $\mathbf{R}\left(\mathbf{a}, \phi\right)$ is a rotation about axis $\mathbf{a}$ by angle ϕ. This defines the discrete twist, m_i, as an integrated quantity associated to a vertex. As in the smooth setting, the discrete twist measures the deviation of the material frame away from parallel transport,

$$\sin\left(m_i\right) = \left(\mathbf{\Omega}_i \cdot \mathbf{d}^{i-1}_1\right) \cdot \left(\mathbf{P}_i \cdot \mathbf{d}^{i-1}_2\right) \ , \tag{3.11}$$

which is the discrete analogue of (3.6). Note that if the frame is twist-free (i.e., $\mathbf{\Omega}_i = \mathbf{P}_i$), then this definition gives $m_i = 0$.

In the discrete setting, the temporal evolution of the material frame, just as its spatial evolution, can be seen as a two-step process: Consider a unit tangent $\mathbf{t}^j$ at consecutive instants in time, $\mathbf{t}^j(t_{k-1})$ and $\mathbf{t}^j(t_k)$. We first parallel transport to align the tangents, and we then rotate by $\omega_3(t_k)$ about $\mathbf{t}^j(t_k)$ to align the material frames, resulting in a finite rotation $\boldsymbol{\omega}^j(t_k)$. This again defines the discrete angular velocity as the material frame's deviation away from parallel transport in the temporal direction,

$$\sin\left(\omega_3^j(t_k)\right) = \left(\boldsymbol{\omega}^j(t_k)\cdot\mathbf{d}_1^j(t_{k-1})\right)\cdot\left(\mathbf{P}^j(t_k)\cdot\mathbf{d}_2^j(t_{k-1})\right) \ , \tag{3.12}$$

where $\mathbf{P}^j(t_k) = \mathbf{P}^{\mathbf{t}^j(t_k)}_{\mathbf{t}^j(t_{k-1})}$.

To define the discrete material curvature, we follow the smooth setting and first define a discrete curvature binormal. One such measure assigns to each *interior* vertex $1 \le i \le n$

$$(\kappa\mathbf{b})_i = 2\tan\left(\frac{\phi_i}{2}\right)\mathbf{b}_i = \frac{2\mathbf{t}^{i-1}\times\mathbf{t}^i}{1+\mathbf{t}^{i-1}\cdot\mathbf{t}^i} \ , \tag{3.13}$$

where $\mathbf{b}_i = \mathbf{t}^{i-1}\times\mathbf{t}^i/\left|\mathbf{t}^{i-1}\times\mathbf{t}^i\right|$ is the discrete binormal vector associated to the vertex and ϕ_i is the turning angle between $\mathbf{t}^{i-1}$ and $\mathbf{t}^i$. Note that this defines the discrete curvature binormal as an integrated quantity. Though other definitions are certainly possible (see, e.g., [Bobenko et al., 2008]), we find this definition computationally convenient since it arises naturally in the discrete picture (see §3.5).

Since the curvature binormal, $(\kappa\mathbf{b})_i$, is associated to vertices, and the material directors, $\mathbf{d}_\alpha^j$, are associated to edges, we must choose how to mix the two quantities to arrive at a definition for the discrete material curvatures. In making this choice, we are motivated by the fact that in the smooth setting, the strains (elongation, twist, and material curvature) can be used to reconstruct the adapted framed curve up to rigid motions [do Carmo, 1992]. One choice that allows such a reconstruction in the discrete setting is to use a simple averaging and define a vertex-based discrete material curvature,

$$\boldsymbol{\kappa}_i = \frac{1}{2}\sum_{j=i-1}^{i}\left(\kappa\mathbf{b}_i\cdot\mathbf{d}_2^j, -\kappa\mathbf{b}_i\cdot\mathbf{d}_1^j\right)^T \ . \tag{3.14}$$

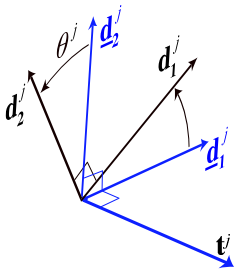
Since the discrete curvature binormal is defined by (3.13) as an integrated quantity, the discrete material curvature is also an integrated quantity.

3.4 Choosing among curve and angle representations

To specify the configuration of an adapted framed curve, some set of coordinates must be chosen (see the discussion in §3.1). Reduced coordinate representations have unshearability and inextensibility constraints built in to the coordinates so that no explicit constraint-handling algorithm is needed. On the other hand, maximal coordinate representations often result in simpler expressions for the energy gradient and Hessian as well as simpler collision handling algorithms due to the explicit representation of the positional degrees of freedom of the centerline. The curve-angle formulation discussed below, a hybrid between these two extremes, uses an explicit centerline representation and reduced coordinates for the adapted frames, which allows for simpler handling of collisions and direct manipulation while sidestepping complications due to enforcing the unshearability constraint.

3.4.1 Reference directors

Thus far, we have been focusing on arbitrary directors, which, intuitively, we associate with the material's principal axes. However, in order to separate material properties from geometry, it is useful to introduce another set of directors, which we call *reference directors*, which represent adapted frames that only depend on the (spatial or temporal) evolution of the centerline. We denote reference directors, and all quantities associated to the reference frame, with an underline, $(\underline{\mathbf{d}}_1(s,t), \underline{\mathbf{d}}_2(s,t))$. Notice that any other adapted frame, such as the material frame, can then be represented by an angle $\theta(s,t)$ which rotates (about the shared tangent) from the reference to the material frame (see Fig. 3.1). The discrete picture (see incident figure) assigns an angle θ^j to each edge:



$$\mathbf{d}_1^j = (\cos\theta^j)\,\underline{\mathbf{d}}_1^j + (\sin\theta^j)\,\underline{\mathbf{d}}_2^j \quad \text{and} \quad \mathbf{d}_2^j = -(\sin\theta^j)\,\underline{\mathbf{d}}_1^j + (\cos\theta^j)\,\underline{\mathbf{d}}_2^j\,. \tag{3.15}$$

Using this representation, any combination of coordinates for the centerline $\mathbf{x}$ and the angles θ specifies a legal configuration for an adapted framed curve. Using these coordinates, we

can express the angular velocity and twist as

$$\omega_3 = \dot{\theta} + \underbrace{\dot{\underline{\mathbf{d}}}_1 \cdot \underline{\mathbf{d}}_2}_{\underline{\omega}_3} \qquad \text{and} \qquad m = \theta' + \underbrace{\underline{\mathbf{d}}_1' \cdot \underline{\mathbf{d}}_2}_{\underline{m}} \ .$$

Here, $\underline{m}$ is the *reference twist* and $\underline{\omega}_3$ is the *reference angular velocity*, quantities associated to the reference frame. In the discrete setting, these quantities can be expressed as

$$\omega_3^j(t_k) = \Delta\theta^j + \underline{\omega}_3^j(t_k) \quad \text{and} \quad m_i(t_k) = \Delta\theta_i + \underline{m}_i(t_k) \ ,$$

where $\Delta\theta^j = \theta^j(t_k) - \theta^j(t_{k-1})$, $\Delta\theta_i = \theta^i(t_k) - \theta^{i-1}(t_k)$, and $\underline{\omega}_3^j(t_k)$ and $\underline{m}_i(t_k)$ are defined analogously to (3.12) and (3.11).

Not all reference directors are created equal How should we define the reference directors $\underline{\mathbf{d}}_1$ and $\underline{\mathbf{d}}_2$? We evaluate the choice of reference frame by favoring ease of computations required by dynamics simulations. In this setting, computational efficiency relies on minimizing the time needed to solve the equations of motion governed by the forces acting on the physical system. Computing the forces on the centerline requires the gradient of the strains defined in §3.3 with respect to vertex positions, and implicit integrators additionally require the Hessian of the strains. The bending and twisting strains contain reference-frame dependent quantities, so we must consider the variation of the reference directors, $\delta\underline{\mathbf{d}}_\alpha$, due to a variation in vertex positions, $\delta\mathbf{x}$, which is given in the continuous and discrete settings respectively by

$$\int \frac{\partial \underline{\mathbf{d}}_\alpha(s,t)}{\partial \mathbf{x}(\sigma,t)} \cdot \delta\mathbf{x}(\sigma,t)\,\mathrm{d}\sigma \quad \text{and} \quad \sum_{i=0}^{n+1} \frac{\partial \underline{\mathbf{d}}_\alpha^j(t_k)}{\partial \mathbf{x}_i(t_k)} \cdot \delta\mathbf{x}_i(t_k) \ . \tag{3.16}$$

If the reference directors $\underline{\mathbf{d}}_\alpha^j(t_k)$ have *nonlocal support* (i.e., depend on vertices arbitrarily far away), the terms in this summation do not vanish, leading to an expensive to compute, dense Hessian. Thus, we seek to minimize this dependence and make the support as compact as possible.

3.4.2 Time-parallel frame

The reference frame must stay adapted to the centerline—for any change in the centerline's tangent, the reference directors must follow, so $\underline{\mathbf{d}}_\alpha(s,t)$ must at the very least depend on

the tangent, $\mathbf{t}(s,t)$. Thus, a natural choice is to define the frame such that it is always rotated by the minimum amount needed to keep it adapted. This selects the frame having zero tangential angular velocity, $\underline{\omega}_3 = 0$, corresponding to parallel transport in *time* using $\dot{\underline{\mathbf{d}}}_\alpha = \left(\mathbf{t} \times \dot{\mathbf{t}}\right) \times \underline{\mathbf{d}}_\alpha$. Given the initial reference directors $\underline{\mathbf{d}}_\alpha(s,0)$, we integrate over *time* to obtain the current reference directors $\underline{\mathbf{d}}_\alpha(s,t)$.

Analogously, for the discrete case, if $\underline{\omega}_3^j = 0$, the discrete ODE

$$\underline{\mathbf{d}}_\alpha^j(t_k) = \mathbf{P}^j \cdot \underline{\mathbf{d}}_\alpha^j(t_{k-1}) \tag{3.17}$$

with initial conditions $\underline{\mathbf{d}}_\alpha^j(t_0)$ governs the frame's temporal evolution. Here, we have defined $\mathbf{P}^j = \mathbf{P}^{\mathbf{t}^j(t_k)}_{\mathbf{t}^j(t_{k-1})}$ (see (3.9)). During an actual simulation, this integration can be done incrementally—only the reference frames at the current time-step must be known to advance the frames to the next time-step. In fact, given the reference frame at time t_{k-1}, the time-parallel reference frame $\underline{\mathbf{d}}_\alpha^j(t_k)$ only depends on $\mathbf{t}^j(t_k)$, and the only nonzero terms in (3.16) are $i \in \{j, j+1\}$. Thus, the force stencil is local and the energy Hessian is banded.

Accumulation of reference twist The ODE governing the (temporal) evolution of the time-parallel frame given by (3.17) is integrated independently at each edge. Consequently, due to the motion of the centerline during the course of the simulation, the reference frame may accumulate twist, denoted by $\underline{m}_i$, which must be accounted for when computing the twist of the material frame, $m_i = \theta^i - \theta^{i-1} + \underline{m}_i$. The change in the reference twist whenever the centerline moves is easy to account for algorithmically: we parallel transport $\underline{\mathbf{d}}_1^{i-1}$ from $\mathbf{t}^{i-1}$ to $\mathbf{t}^i$, rotate about $\mathbf{t}^i$ by angle $\underline{m}^i$, and measure the angle, $\Delta\underline{m}_i$, between the result and $\underline{\mathbf{d}}_1^i$. In §3.5.3, we give an interpretation of these procedure within the context of a compatibility condition relating angular velocity to twist.

3.4.3 Space-parallel frame

We can also consider using the *space-parallel* frame instead, also known as the Bishop frame [1975], which requires $\underline{m} = 0$, corresponding to parallel transporting the reference frame in *space* using $\underline{\mathbf{d}}_\alpha' = \kappa\mathbf{b} \times \underline{\mathbf{d}}_\alpha$. In this case, given $\underline{\mathbf{d}}_\alpha(0,t)$ (the boundary conditions),

an integration over the spatial dimension, s, defines $\underline{\mathbf{d}}_\alpha(s,t)$. This implies the reference directors $\underline{\mathbf{d}}_\alpha(s,t)$ depend on $\kappa\mathbf{b}(\sigma,t)$, and therefore on $\mathbf{x}(\sigma,t)$, for all $\sigma \leq s$.

In the discrete case, the twist-free frame requires $\underline{m}_i = 0$ and is defined via

$$\underline{\mathbf{d}}^i_\alpha(t_k) = \mathbf{P}_i \cdot \underline{\mathbf{d}}^{i-1}_\alpha(t_k) \ , \tag{3.18}$$

with the boundary conditions $\underline{\mathbf{d}}^0_\alpha(t_k)$. Since by definition, the space-parallel reference frame has no twist, the twist of the material frame at a vertex is given by $m_i = \theta^i - \theta^{i-1}$. Because the space-parallel frame $\underline{\mathbf{d}}^j_\alpha(t_k)$ is defined iteratively in terms of the tangents of all edges $\{0,\ldots,j\}$, the terms in the summation in (3.16) are nonzero for all $i \leq j+1$. Thus, to compute any force acting on vertex i that contains reference-dependent terms, we must add up contributions from all edges $j \geq i-1$, which in general leads to a costly to compute, dense energy Hessian.

Applicability of space-parallel frame to dynamics A dense energy Hessian is undesirable from a computational view, and as a result, the space-parallel formulation seems ill-suited for building efficient physical simulations. However, under a certain set of conditions, the space-parallel formulation provides a viable alternative to the time-parallel formulation. If the derivatives of the reference directors do not need to be computed, such as when the material curvature and angular velocity of the material frame do not enter into a description of the dynamics, then the energy Hessian for the space-parallel formulation is also sparse. In fact, as we will show in §4, this case can arise in practice when considering thin, uncurved elastic rods whose cross-sectional inertia can be neglected and whose bending response is isotropic. In this case, when using the space-parallel reference frame, the description of the rod's configuration can be reduced to representing the centerline along with only a single angle for the entire rod (rather than one for each edge) denoting the rod's total (accumulated) twist.

3.5 Gradient of twist

For efficiency of computation within the framework of an implicit time-stepping scheme (where the nonlinear equations of motion are solved using, e.g., Newton's method), we require explicit expressions for both the gradient and Hessian of the strains with respect to the coordinates $\{\mathbf{x}_i\}$ and $\{\theta^j\}$. While those are straightforward to obtain for the relative axial strain, the calculation of gradients and Hessians for the material curvature and twist with respect to vertex positions turns out to be nontrivial, requiring consideration of the variation of parallel transport. While the final expressions for the gradients and Hessians are provided in §4.4, in this section, we show how the concept of holonomy can be used to aid in the computation of the gradient of twist with respect to vertex positions, ultimately allowing us to make use of previous work in performing this computation.

For the time-parallel formulation, the gradient is given by

$$\nabla m_i = \nabla \left(\theta^i - \theta^{i-1} + \underline{m}_i \right) = \nabla \underline{m}_i \ .$$

Thus, we require the expression for the gradient of the twist of the reference frame (see §3.5.4).

For the space-parallel formulation, the expression for the twist at a vertex, $m_i = \theta^i - \theta^{i-1}$, does not explicitly depend on the centerline coordinates, $\{\mathbf{x}_i\}$, but only on the angular coordinates, $\{\theta^j\}$. However, for any *prescribed* boundary condition on the material frame at an edge, $\mathbf{d}_\alpha^n$, the angle, θ^n, between the reference frame and the material frame is no longer an independent coordinate. This is due to the fact that any motion of the centerline will rotate the space-parallel reference frame, $\mathbf{d}_\alpha^n$, and in order to match the prescribed boundary condition, θ^n must compensate for this rotation. Thus, for the space-parallel formulation, we require the expression for the gradient of the angle, θ^n, between the reference frame and the (prescribed) material frame (see §3.5.5).

3.5.1 Holonomy

Holonomy, a classical concept from differential geometry, measures the deficit of geometric data (frames) to close up when parallel transported around a closed loop. In our case (of discrete adapted framed curves), holonomy depends—just like parallel transport itself—only on the centerline. It measures the (scalar) angle when parallel transporting an adapted frame along a closed loop of tangents. Indeed, the fact that holonomy can be expressed as a *scalar* is the reason why it is such a useful concept for computing the requisite gradient.

The concept of holonomy emerges naturally when considering how variations of the centerline positions (or, equivalently, edges) affect parallel transport. Consider the variation of two consecutive edges $\mathbf{e}^{i-1}(\varepsilon)$ and $\mathbf{e}^{i}(\varepsilon)$ with tangents $\mathbf{t}^{i-1}(\varepsilon)$ and $\mathbf{t}^{i}(\varepsilon)$, where ε denotes the variation parameter. As before, we denote the parallel transport from $\mathbf{t}^{i-1}(\varepsilon)$ to $\mathbf{t}^{i}(\varepsilon)$ by $\mathbf{P}_i(\varepsilon)$. We introduce $\widetilde{\mathbf{P}}^j(\varepsilon)$ to denote the parallel transport from $\mathbf{t}^j(0)$ to the deformed configuration, $\mathbf{t}^j(\varepsilon)$, i.e., the rotation that satisfies $\widetilde{\mathbf{P}}^j(0) = \mathbf{I}_3$ and

$$\widetilde{\mathbf{P}}^j(\varepsilon)\left(\mathbf{t}^j(0)\right) = \mathbf{t}^j(\varepsilon) \quad \text{and} \quad \widetilde{\mathbf{P}}^j(\varepsilon)\left(\mathbf{t}^j(0) \times \mathbf{t}^j(\varepsilon)\right) = \mathbf{t}^j(0) \times \mathbf{t}^j(\varepsilon) \ .$$

Now consider the concatenation of parallel transports given by

$$\mathbf{R}^{i-1}(\varepsilon) = \left(\widetilde{\mathbf{P}}^{i-1}(\varepsilon)\right)^T \circ \left(\mathbf{P}_i(\varepsilon)\right)^T \circ \widetilde{\mathbf{P}}^{i}(\varepsilon) \circ \mathbf{P}_i(0) \ , \tag{3.19}$$

depicted by the following mnemonic diagram:

$$\begin{array}{ccc} \mathbf{t}^{i-1}(\varepsilon) & \longrightarrow & \mathbf{t}^{i}(\varepsilon) \\ \uparrow & \psi_i(\varepsilon) & \uparrow \\ \mathbf{t}^{i-1}(0) & \longrightarrow & \mathbf{t}^{i}(0) \end{array}$$

Here, arrows represent parallel transport. Observe that $\mathbf{R}^{i-1}(\varepsilon)$ corresponds to traversing this diagram in counter-clockwise order, starting at $\mathbf{t}^{i-1}(0)$. It follows from the definitions that $\mathbf{R}^{i-1}(\varepsilon)$ maps the tangent $\mathbf{t}^{i-1}(0)$ to itself and is therefore a rotation by angle $\psi_i(\varepsilon)$ about axis $\mathbf{t}^{i-1}(0)$. In the language of differential geometry, $\psi_i(\varepsilon)$ is the *holonomy* of the *connection* induced by parallel transport around the depicted closed loop.

3.5.2 Variation of holonomy

The ingredient that we require is the variation of ψ_i for $1 \leq i \leq n$, which we can obtain from the literature of a related quantity known as the *writhe* of polygonal curves [de Vries, 2005], resulting in

$$\delta\psi_i = \frac{-2\mathbf{t}_{i-1} \times \mathbf{t}_i}{1 + \mathbf{t}_{i-1} \cdot \mathbf{t}_i} \cdot \left(\frac{1}{2} \frac{\delta\mathbf{x}_i - \delta\mathbf{x}_{i-1}}{|\mathbf{e}^{i-1}|} + \frac{1}{2} \frac{\delta\mathbf{x}_{i+1} - \delta\mathbf{x}_i}{|\mathbf{e}^i|} \right) .$$

It can be shown that the corresponding expression in the smooth setting [Le Bret, 1979] is $\delta(\int \psi \mathrm{d}s) = -\int \kappa\mathbf{b} \cdot (\delta\mathbf{x})_s \mathrm{d}s$ where the subscript s denotes differentiation with respect to s. Comparing the discrete expression to the integrand of the smooth case, we can identify the second factor of the discrete expression as a finite-difference approximation of $(\delta\mathbf{x})_s$. In analogy to the smooth form, we take the first factor to be the integrated curvature binormal. Note that this definition coincides with (3.13), and allows us to rewrite the discrete result for the gradient as

$$\nabla_{i-1}\psi_i = \frac{\kappa\mathbf{b}_i}{2\,|\mathbf{e}^{i-1}|} , \qquad \nabla_{i+1}\psi_i = -\frac{\kappa\mathbf{b}_i}{2\,|\mathbf{e}^i|} , \tag{3.20}$$

and $\nabla_i\psi_i = -(\nabla_{i-1} + \nabla_{i+1})\psi_i$.

We pave the way for the computation of the gradient of twist by first deriving a discrete compatibility condition that must be obeyed by the frame, which will relate the frame's twist and angular velocity to the concept of holonomy.

3.5.3 Compatibility condition

For any frame, we can show the relationship between angular velocity and twist using the following diagram.

$$\begin{array}{ccc} \mathbf{d}_\alpha^{i-1}(\varepsilon) & \xrightarrow{m_i(\varepsilon)} & \mathbf{d}_\alpha^{i}(\varepsilon) \\ \uparrow \omega_3^{i-1}(\varepsilon) & \psi_i(\varepsilon) & \uparrow \omega_3^{i}(\varepsilon) \\ \mathbf{d}_\alpha^{i-1}(0) & \xrightarrow[m_i(0)]{} & \mathbf{d}_\alpha^{i}(0) \end{array}$$

Here, the frames $\mathbf{d}_\alpha^j$ are displayed in place of the tangents to show that we are parallel-transporting these frames from one edge to the next. Additionally, each labeled arrow indicates the angle needed to align the *result* of parallel transporting the frame at the tail of the arrow with the frame at the head. Thus, the vertical arrows are labeled with $\omega_3^j(\varepsilon)$ to indicate the angle (as a function of the variation parameter) required to align $\widetilde{\mathbf{P}}^j\left(\mathbf{d}_\alpha^j(0)\right)$ to $\mathbf{d}_\alpha^j(\varepsilon)$.

If we start at $\mathbf{d}_\alpha^{i-1}(0)$ (bottom left) and move the frame first horizontally and then vertically, the result must be the same as moving first vertically and then horizontally. More precisely, the following compatibility condition must be obeyed by m and ω_3:

$$(m_i(\varepsilon) - m_i(0)) - \left(\omega_3^i(\varepsilon) - \omega_3^{i-1}(\varepsilon)\right) = -\psi_i(\varepsilon)\ . \tag{3.21}$$

Note that this compatibility condition must be obeyed by both the material frame and the reference frame.

The smooth analogue to the discrete compatibility condition is that when considering the mixed (spatial and temporal) partial derivatives of the directors of an adapted frame, the order of differentiation does not matter. Taking this into account yields the following equation[1] relating the twist of the (material or reference) frame to its angular velocity.

$$\dot{m} - \omega_3' = -2(\mathbf{t}' \times \dot{\mathbf{t}}) \cdot \mathbf{t}\ .$$

Viewed in this light, the accumulation of the reference twist explained algorithmically in §3.4.2 is the consequence of ensuring that this compatibility condition is maintained.

[1] In the theory of smooth framed surfaces immersed into Euclidean 3-space, this relation is known as the Codazzi-Mainardi equation.

3.5.4 Gradient of twist: time-parallel frame

With our new tool in hand, we consider the time-parallel reference frame and derive the change in its twist when varying the rod's centerline.

$$\begin{array}{ccc} \underline{\mathbf{d}}_\alpha^{i-1}(\varepsilon) & \xrightarrow{\underline{m}_i(\varepsilon)} & \underline{\mathbf{d}}_\alpha^{i}(\varepsilon) \\ \uparrow 0 & \psi_i(\varepsilon) & \uparrow 0 \\ \underline{\mathbf{d}}_\alpha^{i-1}(0) & \xrightarrow[\underline{m}_i(0)]{} & \underline{\mathbf{d}}_\alpha^{i}(0) \end{array}$$

In this diagram, we have explicitly labeled the vertical arrows with 0 because, by definition, the time-parallel reference frame has no tangential angular velocity. We are interested in the angle of rotation, $\underline{m}_i(\varepsilon)$, required to align $\mathbf{P}_i\left(\mathbf{F}^{i-1}(\varepsilon)\right)$ to $\mathbf{F}^i(\varepsilon)$. It follows from the discrete compatibility condition in (3.21) that the resulting angle is $\underline{m}_i(\varepsilon) = \underline{m}_i(0) - \psi_i(\varepsilon)$. For computing forces, we require the gradient of twist with respect to vertex positions. In the case of using the time-parallel reference frame, twist is simply $m_i = \theta^i - \theta^i + \underline{m}_i$, so we can use (3.20) to obtain

$$\nabla_{i-1} m_i = -\frac{\kappa \mathbf{b}_i}{2\left|\mathbf{e}^{i-1}\right|}\,, \quad \nabla_{i+1} m_i = \frac{\kappa \mathbf{b}_i}{2\left|\mathbf{e}^{i}\right|}\,, \tag{3.22}$$

and $\nabla_i m_i = -\left(\nabla_{i-1} + \nabla_{i+1}\right) m_i$.

3.5.5 Gradient of twist: space-parallel frame

As noted in §3.5, for the space-parallel formulation, we are interested in the angle of rotation about the tangent of the reference frame at an edge with prescribed boundary conditions on the material frame. This situation is depicted in the following diagram:

$$\begin{array}{ccccccc} \underline{\mathbf{d}}_\alpha^{0}(\varepsilon) & \xrightarrow{0} & \underline{\mathbf{d}}_\alpha^{1}(\varepsilon) & \xrightarrow{0} & \cdots & \underline{\mathbf{d}}_\alpha^{n-1}(\varepsilon) & \xrightarrow{0} & \underline{\mathbf{d}}_\alpha^{n}(\varepsilon) \\ \uparrow 0 & \psi_1(\varepsilon) & \uparrow & \psi_2(\varepsilon) & & \uparrow & \psi_n(\varepsilon) & \uparrow \Psi^n \\ \underline{\mathbf{d}}_\alpha^{0}(0) & \xrightarrow[0]{} & \underline{\mathbf{d}}_\alpha^{1}(0) & \xrightarrow[0]{} & \cdots & \underline{\mathbf{d}}_\alpha^{n-1}(0) & \xrightarrow[0]{} & \underline{\mathbf{d}}_\alpha^{n}(0) \end{array}$$

Here, Ψ^n denotes the angle of rotation of the reference frame about the tangent caused by varying the centerline. The horizontal arrows in this case are labeled with a 0 to indicate that the space-parallel frame is twist-free, and the first vertical arrow is labeled with a 0 because we ensure that $\underline{\mathbf{d}}_1^0$ is always updated via parallel transport in time.

We can easily solve for Ψ^n by making repeated use of the compatibility condition (3.21) to obtain $\Psi^n = \sum_{i=1}^n \psi_i$. Since θ^n must compensate for this angle, the requisite gradient is given by

$$\nabla_i \theta^n = -\nabla_i \Psi^n = -\sum_{k=1}^{n} \nabla_i \psi_k \ . \tag{3.23}$$

By (3.20), this sum will have at most three non-zero terms.

Chapter 4

Elastic rods

Elastic rods have many interesting applications in a variety of fields including biology, such as modeling DNA [Yang et al., 1993]; medicine, such as simulating needles or sutures in a virtual surgery environment [Chentanez et al., 2009]; and animation, such as simulating hair [Hadap and Magnenat-Thalmann, 2001]. They are ideal for modeling the stretching, bending, and twisting deformations of such long, thin elastic materials. The motion of an elastic rod is governed by the competition between external forces and the material's resistance to these deformations.

In the present chapter, we build a discrete model for simulating elastic rods based on the widely accepted and utilized Kirchhoff rod model, which describes the dynamics of an unshearable, inextensible elastic rod. We consider the special case of an uncurved rod with isotropic bending response and the general case of a curved, anisotropic rod. Our main goal is to show how the two formulations for describing adapted frame curves introduced in §3, based on parallel transport either in time or in space, are ideally suited to simulating these two cases. We describe two approaches for handling the (in)extensibility of the rod within this model using either a physically-based stretching potential or a manifold projection method to enforce exact inextensibility. We show how the manifold projection method can also be applied to coupling rigid bodies to segments of an elastic rod. Our model additionally makes use of the quasistatic assumption, which posits that the material frame can be treated

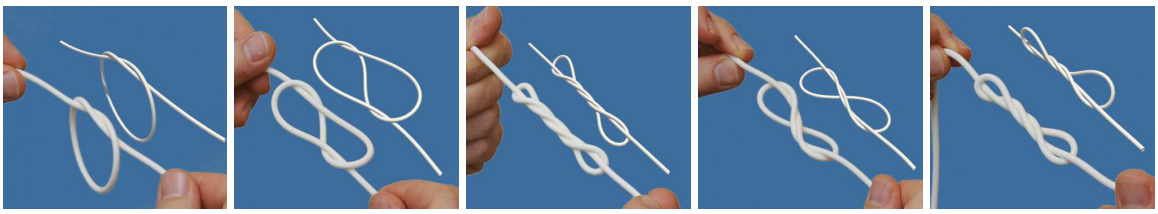


Figure 4.1: A simple (trefoil) knot tied on an elastic rod can be turned into a number of fascinating shapes when twisted. Starting with a twist-free knot *(left)*, we observe both continuous and discontinuous changes in the shape, for both directions of twist. Using our discrete model of elastic rods, we are able to reproduce experiments with high accuracy.

quasistatically rather than dynamically, leading to a more efficient computational model.

4.1 Related work

Mathematical analysis, modeling, and simulation of elastic rods is an active field in mechanics [van der Heijden et al., 2003; Goyal et al., 2008], numerical analysis [Falk and Xu, 1995], and geometry [Bobenko and Suris, 1999; Lin and Schwetlick, 2004], with applications spanning medicine [Lenoir et al., 2006], biology [Yang et al., 1993; Klapper, 1996], the study of knots [Phillips et al., 2002; Brown et al., 2004], and computer graphics [Chang et al., 2007]. For a broad starting point, refer to the expositions of Maddocks [1984; 1994], Rubin [2000], and Antman [2005]. For an overview of elastic rods in the context of strand and hair simulation, refer to the survey by Ward et al. [2007] and the course notes of Hadap et al. [2007].

In mechanical engineering, elastic rods are typically treated with a finite difference [Klapper, 1996] or finite element [Yang et al., 1993; Goyal et al., 2008] discretization of the smooth equations. In the finite element community, Boyer and Primault [2004] also discuss reference frames for adapted framed curves but avoid parametrizing the configuration by advocating the use of a Lie group integrator. Goldstein and Langer [1995] observed that using the Bishop (space-parallel) frame simplifies both the analytical formulation and the numerical implementation of the dynamics of symmetric rods.

In graphics, Pai [2002] applied a discretization of the Cosserat rod model to simulate a strand. Bertails et al. [2006] used a piecewise helical discretization to produce compelling animations of curly hair using few elements per strand. Hadap [2006] considered a serial multi-body chain and used differential algebraic equations to treat the attendant numerical stiffness. These later works used an *implicit* centerline representation based on reduced (curvature) coordinates.

By contrast, Choe et al. [2005] represented the centerline *explicitly* by a sequence of edges connected by linear and torsional springs. Grégoire and Schömer [2007] proposed an explicit centerline discretization coupled to a quaternionic material frame representation. Spillmann and Teschner [2007, 2008] built on this idea in their development of algorithms for dynamics and contact. Theetten et al. [2008] presented a geometric spline-based approach that can model large-displacement plastic deformations. These authors argue that an explicit (displacement) representation of the centerline facilitates the simulation of closed elastic loops (e.g., rubber bands) and complex contact scenarios. When using an explicit centerline representation, loops simply require one to consider the indexing of the degrees of freedom, whereas simulating loops using reduced coordinates requires integrating an ODE subject to boundary conditions. Likewise, impulses and forces on the centerline due to contact are simpler to directly incorporate when the centerline is represented explicitly.

4.2 Elastic energy

The Kirchhoff theory of elastic rods assigns an elastic energy, $E(\Gamma)$, to an adapted framed curve $\Gamma = \{\mathbf{x}; \mathbf{d}_1, \mathbf{d}_2, \mathbf{t}\}$ (see §3.2). The elastic energy is given as a function of the geometric strains of the elastic rod relative to an undeformed configuration (denoted with an overline). The simplest linear constitutive model—the equivalent of Hooke's law for continuum mechanics—results in an energy that is quadratic in the strains.

This energy is assembled from the measures of strain introduced in §3.3 and is written as the sum of stretching, bending, and twisting contributions:

$$E(\Gamma) = E_{\text{stretch}}(\Gamma) + E_{\text{bend}}(\Gamma) + E_{\text{twist}}(\Gamma) \ .$$

The classical Kirchhoff equations for rods are obtained from this type of energy using Lagrangian mechanics (see, e.g., [Audoly and Pomeau, 2010]).

We consider two cases: the *general case* allowing for a naturally curved elastic rod with anisotropic bending response, and the *special case* of a naturally straight, isotropic rod (i.e., no preferred bending direction).

4.2.1 Stretching energy

For both the special and general cases, the expression for the stretching energy in the smooth setting is given by

$$E_{\text{stretch}}(\Gamma) = \frac{1}{2}\int k_{\text{s}}\,\varepsilon^2 \mathrm{d}s \ ,$$

where ε is the axial strain defined in (3.2), and k_{s} is the stretching stiffness. In the discrete setting, recalling that the discrete stretching strain corresponds to a pointwise measure, we can write the stretching energy of a discrete elastic rod as a sum over stencils associated to edges,

$$E_{\text{stretch}}(\Gamma) = \frac{1}{2}\sum_{j=0}^{n} k_{\text{s}}^{j}\left(\varepsilon^{j}\right)^2 \left|\overline{\mathbf{e}}^{j}\right| \ ,$$

where ε^j is the discrete axial strain defined in (3.3), and k_{s}^j is the stretching stiffness associated with the edge (see §4.2.4).

4.2.2 Bending energy

General case The bending energy in the smooth setting is given by

$$E_{\text{bend}}(\Gamma) = \frac{1}{2}\int (\boldsymbol{\kappa}-\overline{\boldsymbol{\kappa}})^T \mathbf{B}\,(\boldsymbol{\kappa}-\overline{\boldsymbol{\kappa}})^T \,\mathrm{d}s \ ,$$

where $\boldsymbol{\kappa}$ is the material curvature defined in (3.7), and the metric $\mathbf{B}$ measures the rod's bending response. This formulation describes the general case allowing for a naturally curved elastic rod with an anisotropic bending response. In the discrete setting, the equivalent expression for the energy in this case is given by a sum over stencils associated to

(interior) vertices,

$$E_{\text{bend}}(\Gamma) = \frac{1}{2}\sum_{i=1}^{n} \frac{1}{\bar{l}_i} (\boldsymbol{\kappa}_i - \overline{\boldsymbol{\kappa}}_i)^T \mathbf{B}_i (\boldsymbol{\kappa}_i - \overline{\boldsymbol{\kappa}}_i) \ ,$$

where $\boldsymbol{\kappa}_i$ is the discrete material curvature defined in (3.14), $\mathbf{B}_i$ is the bending metric associated to the vertex (see §4.2.4), and $\bar{l}_i$ is defined in (3.1) as the length of the domain associated to a vertex in the undeformed configuration of the rod. The scaling in this equation accounts for the fact that the discrete material curvature, $\boldsymbol{\kappa}_i$, is an integrated quantity.

If the material directors are aligned with the principal bending directions of the rod in the undeformed configuration, then $\mathbf{B}_i$ is diagonal. This slightly simplifies computation since it eliminates the mixed term, $\kappa_1 \cdot \kappa_2$, from the energy.

Special case If we consider the special case of a naturally straight rod ($\overline{\boldsymbol{\kappa}} = 0$) with isotropic bending response ($\mathbf{B} = \alpha \mathbf{I}_2$, where $\mathbf{I}_2$ is the 2×2 identity matrix), the bending energy simplifies to

$$E_{\text{bend}}(\Gamma) = \frac{1}{2}\int \alpha \boldsymbol{\kappa}^2 \mathrm{d}s = \frac{1}{2}\int \alpha \kappa^2 \mathrm{d}s \ ,$$

where $\kappa = t'$ is the (scalar) curvature of the centerline. In the discrete setting, we can write this energy as

$$E_{\text{bend}}(\Gamma) = \frac{1}{2}\sum_{i=1}^{n} \frac{\alpha_i \kappa_i^2}{\bar{l}_i} \ .$$

where $\kappa_i = 2\tan(\phi_i/2)$ is the (scalar) curvature associated to the vertex, given as a function of the turning angle, ϕ_i. Note that in the special case, the bending energy does *not* depend on the material frame.

4.2.3 Twisting energy

General case The twisting energy in the smooth setting is given by

$$E_{\text{twist}}(\Gamma) = \frac{1}{2}\int \beta (m - \overline{m})^2 \, \mathrm{d}s \ ,$$

where m is the twist defined in (3.6) and β is the twist stiffness. The discrete equivalent of this expression is also given by a sum over stencils associated to (interior) vertices,

$$E_{\text{twist}}(\Gamma) = \frac{1}{2}\sum_{i=1}^{n} \frac{\beta_i (m_i - \overline{m}_i)^2}{\overline{l}_i} ,$$

where m_i is the discrete twist formally defined by (3.11), β_i is the twist stiffness associated to the vertex (see §4.2.4), and $\overline{l}_i$ again accounts for the scaling of the (integrated) discrete twist (3.1).

Special case In the special case, we can use the following argument to slightly simplify the expression for the twisting energy. Recall from §4.2.2 that the bending energy for the special case does not depend on the material frame, meaning we are free to choose any initial orientation for the material frame without the choice affecting the bending term. Unlike in the general case, where we choose the material directions such that $\mathbf{B}_i$ is diagonal, we instead choose the material frame such that it is twist-free in the undeformed configuration ($\overline{m}_i = 0$), leading to the expression

$$E_{\text{twist}}(\Gamma) = \frac{1}{2}\sum_{i=1}^{n} \frac{\beta_i m_i^2}{\overline{l}_i} .$$

4.2.4 Elastic force stiffnesses

Physical force stiffnesses depend on the geometry of the rod's cross-section. In our discrete rods model, we assume that each edge has an elliptical cross-section with major and minor radii given by a^j and b^j, respectively, so that the cross-sectional area is $A^j = \pi a^j b^j$. At vertices, we define $a_i = (a^{i-1} + a^i)/2$ (and likewise for b_i) so that the cross-sectional area satisfies $A_i = \pi a_i b_i$. The stretching, twisting, and bending stiffness can be written as [Landau and Lifshitz, 1981]

$$k_{\text{s}}^j = EA^j, \quad \beta_i = \frac{GA_i\left(a_i^2 + b_i^2\right)}{4}, \quad \mathbf{B}_i = \frac{EA_i}{4}\begin{pmatrix} a_i^2 & 0 \\ 0 & b_i^2 \end{pmatrix},$$

respectively. Here E denotes Young's modulus and G denotes the shear modulus of the material. Notice that stretching stiffness is associated with edges, while bending and twisting stiffnesses are associated with vertices.

4.3 Quasistatic material frame postulation

Our goal is to arrive at the equations of motion describing the time-evolution of the elastic rod. We first pave the way with an important simplifying assumption.

The speed of a twist wave increases without bound as the rotational inertia of the cross section vanishes. By contrast, bending waves are dispersive [Sommerfeld, 1964]—their speed depends on their wavelength, λ—with velocity vh/λ, where h is the rod radius, and v is solid material's speed of sound. Consequently, twist waves travel much faster than bending waves with large wavelengths $\lambda \gg h$ (i.e., much larger than the rod radius). For the problems we consider, the relevant *dynamics* are in (large-wavelength) bending modes, and it is wasteful of computation to resolve the temporal evolution of the twist waves.

Consider the limit of a vanishing cross-sectional rotational inertia: twist waves propagate instantly. At any instant in time, the material frames are the minimizer of elastic energy, subject to any given boundary conditions on the material frame:

$$\frac{\partial E(\Gamma)}{\partial \theta^j} = 0 \ , \tag{4.1}$$

Boundary conditions The value of θ^j for an edge may be *prescribed* by a given boundary condition on the material frame. For the time-parallel reference frame, the orientation of the material frame at an edge, $\mathbf{d}_\alpha$, determines the value of θ^j. By contrast, for the space-parallel reference frame, the value of θ^j that matches the given boundary condition also depends on the centerline because the reference frame, $\underline{\mathbf{d}}^j_\alpha$, depends on $\mathbf{x}_i$ for $i \leq j$ (see §3.4).

In practice, we consider *stress-free* ends (i.e., no boundary conditions) or *clamped* ends (i.e., assigning the material frame for $j = 0$ and $j = n$). We use (4.1) for all θ^j variables whose value is not prescribed by a boundary condition, ensuring that the number of equations matches the number of unknowns for the angles θ^j.

Figure 4.2: Anisotropy of the cross-sections and natural curvature of the centerline "conspire" to produce non-uniform twist distribution. *From top to bottom*: reference configuration, moderate twist, large twist.

Special case For an isotropic rod with a naturally straight undeformed configuration, (4.1) implies that a clamped rod has uniform twist moment,

$$\frac{\beta_i m_i}{\bar{l}_i} = \frac{\theta^n - \theta^0}{\sum_{k=1}^{n} \bar{l}_k/\beta_k} = \widetilde{\beta}\left(\theta^n - \theta^0\right) = constant \ , \tag{4.2}$$

where $\widetilde{\beta} = \left(\sum_{k=1}^{n} \bar{l}_k/\beta_k\right)^{-1}$. For a rod with uniform cross-section (i.e., for which all β_i are equal), this further implies that the pointwise twist, $m_i/\bar{l}_i = \left(\theta^n - \theta^0\right)/\sum_{k=1}^{n} \bar{l}_i$, is also constant.

Substituting (4.2) into the formula for $E(\Gamma)$ gives the simplified expression

$$E(\Gamma) = \frac{1}{2}\sum_{j=0}^{n} k_{\mathrm{s}} \left(\varepsilon^j\right)^2 \left|\overline{\mathbf{e}}^j\right| + \frac{1}{2}\sum_{i=1}^{n} \frac{\alpha_i \kappa_i^2}{\bar{l}_i} + \frac{1}{2}\widetilde{\beta}\left(\theta^n - \theta^0\right)^2 \ . \tag{4.3}$$

General case For the general case, (4.1) includes contributions from both the bending and twisting energy terms, so the expression for the energy cannot be simplified as in the special case. As illustrated in Fig. 4.2, the distribution of twist is no longer uniform along the length of the rod. In this case, we enforce the quasistatic condition (4.1) as part of the time-stepping algorithm, which requires both the gradient and Hessian of the energy with respect to the θ^j variables, which can be constructed from the gradient and Hessian of the

twist, m_i, and material curvature, $\boldsymbol{\kappa}_i$. These gradients are given by

$$\frac{\partial m_i}{\partial \theta^i} = -\frac{\partial m_i}{\partial \theta^{i-1}} = 1 \quad \text{and} \quad \frac{\partial \boldsymbol{\kappa}_i}{\partial \theta^j} = -\frac{1}{2}\left(\kappa \mathbf{b}_i \cdot \mathbf{d}_1^j, \kappa \mathbf{b}_i \cdot \mathbf{d}_2^j\right)^T \text{ for } j \in \{i-1, i\} \ . \tag{4.4}$$

The Hessian of m_i with respect to the θ^j variables is 0. The Hessian of the material curvature is given by

$$\frac{\partial^2 \boldsymbol{\kappa}_i}{\partial \left(\theta^j\right)^2} = \frac{1}{2}\left(-\kappa \mathbf{b}_i \cdot \mathbf{d}_2^j, \kappa \mathbf{b}_i \cdot \mathbf{d}_1^j\right)^T \text{ for } j \in \{i-1, i\} \ . \tag{4.5}$$

4.4 Gradients and Hessians

In order to simulate the dynamics of an elastic rod, we require explicit expressions for the forces in the rod. Since the elastic energies are all quadratic in the strains introduced in §3.2, the forces are simple to write down given the strains and their gradients. In addition, for efficient implementations of implicit methods, we require the Hessians of the strains. Here, we provide the required expressions for the gradients and Hessians.

4.4.1 Stretching

The gradient of the axial strain, $\varepsilon^j = \left|\mathbf{e}^j\right| / \left|\overline{\mathbf{e}}^j\right| - 1$, with respect to vertex positions is given by

$$\partial \varepsilon^j / \partial \mathbf{x}_{j+1} = -\partial \varepsilon^j / \partial \mathbf{x}_j = \mathbf{t}^j / \left|\overline{\mathbf{e}}^j\right| \ ,$$

and the Hessian is given by

$$\frac{\partial^2 \partial \varepsilon^j}{\partial \mathbf{x}_j^2} = \frac{\partial^2 \partial \varepsilon^j}{\partial \mathbf{x}_{j+1}^2} = -\frac{\partial^2 \partial \varepsilon^j}{\partial \mathbf{x}_{j+1} \partial \mathbf{x}_j} = -\frac{\partial^2 \partial \varepsilon^j}{\partial \mathbf{x}_j \partial \mathbf{x}_{j+1}} = \frac{1}{\left|\mathbf{e}^j\right| \left|\overline{\mathbf{e}}^j\right|}\left(\mathbf{I}_3 - \mathbf{t}^j \otimes \mathbf{t}^j\right)$$

4.4.2 Special case

For the special case, we use explicit time integration (see §4.6), so only the gradient of the bending and twisting energy is needed. For the twisting energy, we require the gradient of

θ^n with respect to vertex positions, given by (3.23). The gradient of the (scalar) curvature at a vertex, $\kappa_i = 2\tan(\phi_i/2) = |\kappa \mathbf{b}_i|$, is given by

$$\frac{\partial \kappa_i}{\partial \mathbf{x}_{i-1}} = \frac{2\kappa \mathbf{b}_i \times \mathbf{t}^i + \kappa_i^2 \left(\mathbf{t}^{i-1} + \mathbf{t}^i\right)}{\kappa_i \left|\mathbf{e}^{i-1}\right| \left(1 + \mathbf{t}^{i-1} \cdot \mathbf{t}^i\right)} , \qquad \frac{\partial \kappa_i}{\partial \mathbf{x}_{i+1}} = \frac{2\kappa \mathbf{b}_i \times \mathbf{t}^{i-1} - \kappa_i^2 \left(\mathbf{t}^{i-1} + \mathbf{t}^i\right)}{\kappa_i \left|\mathbf{e}^i\right| \left(1 + \mathbf{t}^{i-1} \cdot \mathbf{t}^i\right)}$$

with $\partial \kappa_i / \partial \mathbf{x}_i = -\left(\partial/\partial \mathbf{x}_{i-1} + \partial/\partial \mathbf{x}_{i+1}\right) \kappa_i$.

4.4.3 General case

For the general case, when using time-parallel directors, bending and twisting energy, gradient, and Hessian are *local* quantities. It hence suffices to consider three consecutive vertices $\mathbf{x}_0$, $\mathbf{x}_1$, and $\mathbf{x}_2$ along the centerline and corresponding consecutive edges $\mathbf{e}$ and $\mathbf{f}$, with unit tangents $\mathbf{t}^e$ and $\mathbf{t}^f$, respectively. For compactness, we will give the derivatives of the strains with respect to the two edges, $\mathbf{e}$ and $\mathbf{f}$, instead of with respect to vertex positions. The derivatives with respect to the angles, θ^j, are given by (4.4) and (4.5).

The curvature binormal (associated with the middle vertex) is given by $\kappa \mathbf{b} = 2(\mathbf{t}^e \times \mathbf{t}^f)/\chi$, where $\chi = 1 + (\mathbf{t}^e \cdot \mathbf{t}^f)$. Denoting material frames by $(\mathbf{d}_1^e, \mathbf{d}_2^e)$ and $(\mathbf{d}_1^f, \mathbf{d}_2^f)$, the vertex-based material curvatures take the form

$$\kappa_1 = \frac{1}{2}\left(\mathbf{d}_2^e + \mathbf{d}_2^f\right) \cdot \kappa \mathbf{b} \quad \text{and} \quad \kappa_2 = -\frac{1}{2}\left(\mathbf{d}_1^e + \mathbf{d}_1^f\right) \cdot \kappa \mathbf{b} \ .$$

Finally, we denote twist by m (associated with the middle vertex), and we abbreviate $\tilde{\mathbf{t}} = (\mathbf{t}^e + \mathbf{t}^f)/\chi$ and $\tilde{\mathbf{d}}_i = (\mathbf{d}_i^e + \mathbf{d}_i^f)/\chi$.

Gradients The twist gradient (see §3.5) is given by

$$\frac{\partial m}{\partial \mathbf{e}} = \frac{\kappa \mathbf{b}}{2\left|\mathbf{e}\right|} \quad \text{and} \quad \frac{\partial m}{\partial \mathbf{f}} = \frac{\kappa \mathbf{b}}{2\left|\mathbf{f}\right|} \ .$$

The gradients of material curvatures take the form

$$\frac{\partial \kappa_1}{\partial \mathbf{e}} = \frac{1}{|\mathbf{e}|}\left(-\kappa_1 \tilde{\mathbf{t}} + \mathbf{t}^f \times \tilde{\mathbf{d}}_2\right), \quad \frac{\partial \kappa_1}{\partial \mathbf{f}} = \frac{1}{|\mathbf{f}|}\left(-\kappa_1 \tilde{\mathbf{t}} - \mathbf{t}^e \times \tilde{\mathbf{d}}_2\right) \ ,$$

and similarly for the gradient of κ_2.

Hessians For the second variation of twist we obtain

$$\begin{aligned}
\frac{\partial^2 m}{\partial \mathbf{e}^2} &= -\frac{1}{4\left|\mathbf{e}\right|^2}\left(\kappa\mathbf{b}\otimes\left(\mathbf{t}^e+\tilde{\mathbf{t}}\right)+\left(\mathbf{t}^e+\tilde{\mathbf{t}}\right)\otimes\kappa\mathbf{b}\right)\ , \\
\frac{\partial^2 m}{\partial \mathbf{f}^2} &= -\frac{1}{4\left|\mathbf{f}\right|^2}\left(\kappa\mathbf{b}\otimes\left(\mathbf{t}^f+\tilde{\mathbf{t}}\right)+\left(\mathbf{t}^f+\tilde{\mathbf{t}}\right)\otimes\kappa\mathbf{b}\right)\ , \\
\frac{\partial^2 m}{\partial \mathbf{e}\partial \mathbf{f}} &= \left(\frac{\partial^2 m}{\partial \mathbf{f}\partial \mathbf{e}}\right)^T = \frac{1}{2|\mathbf{e}||\mathbf{f}|}\left(\frac{2}{\chi}\left[\mathbf{t}^e\right]_\times-\kappa\mathbf{b}\otimes\tilde{\mathbf{t}}\right)\ .
\end{aligned}$$

The second variations of material curvatures are given by

$$\begin{aligned}
\frac{\partial^2 \kappa_1}{\partial \mathbf{e}^2} &= \frac{1}{|\mathbf{e}|^2}\left(2\kappa_1\tilde{\mathbf{t}}\otimes\tilde{\mathbf{t}}-(\mathbf{t}^f\times\tilde{\mathbf{d}}_2)\otimes\tilde{\mathbf{t}}-\tilde{\mathbf{t}}\otimes(\mathbf{t}^f\times\tilde{\mathbf{d}}_2)\right) \\
&\quad -\frac{1}{\chi|\mathbf{e}|^2}\kappa_1(Id-\mathbf{t}^e\otimes\mathbf{t}^e)+\frac{1}{4|\mathbf{e}|^2}\left(\kappa\mathbf{b}\otimes\mathbf{d}_2^e+\mathbf{d}_2^e\otimes\kappa\mathbf{b}\right)\ , \\
\frac{\partial^2 \kappa_1}{\partial \mathbf{f}^2} &= \frac{1}{|\mathbf{f}|^2}\left(2\kappa_1\tilde{\mathbf{t}}\otimes\tilde{\mathbf{t}}+(\mathbf{t}^e\times\tilde{\mathbf{d}}_2)\otimes\tilde{\mathbf{t}}+\tilde{\mathbf{t}}\otimes(\mathbf{t}^e\times\tilde{\mathbf{d}}_2)\right) \\
&\quad -\frac{1}{\chi|\mathbf{f}|^2}\kappa_1(Id-\mathbf{t}^f\otimes\mathbf{t}^f)+\frac{1}{4|\mathbf{f}|^2}\left(\kappa\mathbf{b}\otimes\mathbf{d}_2^f+\mathbf{d}_2^f\otimes\kappa\mathbf{b}\right)\ , \\
\frac{\partial^2 \kappa_1}{\partial \mathbf{e}\partial \mathbf{f}} &= \left(\frac{\partial^2 \kappa_1}{\partial \mathbf{f}\partial \mathbf{e}}\right)^T = \frac{-\kappa_1}{\chi|\mathbf{e}||\mathbf{f}|}\left(Id+\mathbf{t}^f\otimes\mathbf{t}^e\right) \\
&\quad +\frac{1}{|\mathbf{e}||\mathbf{f}|}\left(2\kappa_1\tilde{\mathbf{t}}\otimes\tilde{\mathbf{t}}+(\mathbf{t}^e\times\tilde{\mathbf{d}}_2)\otimes\tilde{\mathbf{t}}-\tilde{\mathbf{t}}\otimes(\mathbf{t}^f\times\tilde{\mathbf{d}}_2)+\left[\tilde{\mathbf{d}}_2\right]_\times\right)\ ,
\end{aligned}$$

and similarly for κ_2.

4.5 Constraints

For simulating extensible rods, it suffices to include the stretching term in the energy (see §4.2.1); however, simulating truly inextensible rods would require an infinitely stiff stretching force. In addition, many interesting physical systems can be modeled by the coupling of elastic rods to rigid bodies. Canonical examples include the torsional and Wilberforce pendulums, comprised of a rigid body suspended under gravity by a thin elastic rod. To maintain inextensibility and rigid body coupling, we can add auxiliary constraints to our system.

4.5.1 Constraint formulation

Inextensibility constraints For each edge of the rod, we use the quadratic constraint equations

$$\left|\mathbf{e}^j\right|^2 - \left|\overline{\mathbf{e}}^j\right|^2 = 0 \ ,$$

where (the precomputed) $\overline{\mathbf{e}}^j$ refers to the undeformed configuration.

Rigid body coupling constraints The welding of the body to the rod requires the body's position and orientation, and the rod edge's position and material frame, to be in one-to-one correspondence. Attaching the rigid body to the first edge of the rod gives three constraints:

$$\mathbf{q} \cdot \mathbf{q} - 1 = 0 \ , \qquad \mathbf{q}\,\overline{\mathbf{x}}_0\,\mathbf{q}^* + \mathbf{r} - \mathbf{x}_0 = 0 \ , \qquad \mathbf{q}\,\overline{\mathbf{x}}_1\,\mathbf{q}^* + \mathbf{r} - \mathbf{x}_1 = 0 \ ,$$

where $\mathbf{r} \in \mathbb{R}^3$ is the translation vector and $\mathbf{q} \in \mathbb{H}$ is the unit quaternion [Hanson, 2006] mapping the body's center of mass and orientation, respectively, from the reference to the current configuration, and $\mathbf{q}^*$ is the conjugate of $\mathbf{q}$. The first equation ensures that $\mathbf{q}$ is unit length, so $\mathbf{q}\overline{\mathbf{x}}_0\mathbf{q}^*$ is a rotation of $\overline{\mathbf{x}}_0$ that is summed with $\mathbf{r}$ using vector addition. The second and third equations ensure that the rod's first edge and the rigid body transform identically.

4.5.2 Enforcing constraints

A more general discussion of constraint enforcement can be found in §6.3.3. We use a manifold projection method, since this allows us to reuse an existing rigid body code [Smith, 2008] without modifying its time integration, collision response, or other internal structures. However, any other method for enforcing these constraints could be used, and we include a discussion of our approach for completeness.

Fast manifold projection For the enforcement step, we adopt and extend the *fast projection* method of Goldenthal et al. [2007]. This method takes an unconstrained configuration

and finds a nearby constrained configuration. The notion of "nearby" is made precise by the natural metric on the configuration manifold. We extend the application of fast projection to coupled systems involving both positional as well as rotational degrees of freedom by considering the metric induced by the kinetic energy $\frac{1}{2}\mathbf{y}\widetilde{\mathbf{M}}\mathbf{y}^T$, where the $(3n+12)\times(3n+12)$ *generalized mass matrix* and the *generalized velocity* $\mathbf{y}\in\mathbb{R}^{3n+12}$ are defined respectively by

$$\widetilde{\mathbf{M}} = \begin{pmatrix} 4\cdot\mathbf{I}_{\text{body}} & & \\ & \mathcal{M}\cdot\mathbf{I}_3 & \\ & & \mathbf{M} \end{pmatrix} \qquad \text{and} \qquad \mathbf{y} = (\mathbf{q}^{-1}\dot{\mathbf{q}}, \dot{\mathbf{r}}, \dot{\mathbf{x}}) \ .$$

Here $\mathbf{M}$ is the rod's (diagonal) $3(n+2)\times 3(n+2)$ mass matrix, $\mathcal{M}$ is the body's total (scalar) mass, and $\mathbf{I}_{\text{body}}$ is the body's (symmetric) moment of inertia tensor expressed as a 3×3 matrix in the reference coordinates. Since $\mathbf{q}$ is a *unit* quaternion, $\mathbf{q}^{-1}\dot{\mathbf{q}}$ corresponds to a vector in $\mathbb{R}^3$ [Hanson, 2006]. The kinetic energy has contributions from the body rotation, body translation, and centerline translation.

We initialize the first iteration of fast projection with the results of an unconstrained timestep. Each step of fast projection improves on the guess by computing the first Newton iteration for the minimization of the functional $\mathbf{y}\widetilde{\mathbf{M}}\mathbf{y}^T - \mathbf{c}^T\boldsymbol{\lambda}$, where (using Goldenthal's notation) $\mathbf{c}$ is the vector of constraint equations, and $\boldsymbol{\lambda}$ is the vector of corresponding Lagrange multipliers. Thus, by formulating the kinetic energy using a generalized mass matrix in a high-dimensional configuration space, we are able to directly apply fast projection iterations to rigid body coupling. In all of our examples, we found that after 3–5 steps the method converged to within a tolerance of 10^{-8} in satisfying the constraint $\mathbf{c}=0$.

After the iterations converge, fast projection requires a velocity update [Goldenthal et al., 2007]. Using our generalized coordinates, the appropriate update is

$$(\dot{\mathbf{q}}, \dot{\mathbf{r}}, \dot{\mathbf{x}}) \leftarrow (\dot{\mathbf{q}}, \dot{\mathbf{r}}, \dot{\mathbf{x}}) - \frac{1}{h}(2\mathbf{q}_0\mathbf{q}^{-1}, \mathbf{r}_0 - \mathbf{r}, \mathbf{x}_0 - \mathbf{x}) \ .$$

Our discussion above immediately generalizes to the case of multiple rigid bodies attached to multiple rods. As a corollary, we can couple a rigid body to *any* edge on the rod simply by splitting the rod at that edge. However, care must be taken to understand the *induced boundary conditions*: each rigid body's position and orientation serves to clamp the

position and orientation of the centerline and material frame for each coupled rod end-edge. After fast projection, the material frame *vector* induced by the rigid body orientation is $\mathbf{d}_1^0 = \mathbf{q}\overline{\mathbf{m}}_1^0\mathbf{q}^*$.

4.5.3 Torque transfer

We consider a methodical categorization of those interface forces that are automatically transferred between the rod and rigid body via projection and those that remain to be transferred explicitly. Observe that forces that correspond to gradients of the independent variables, $\mathbf{q}, \mathbf{r}, \mathbf{x}$, are automatically transferred between the rod and the body by the projection step. The material frame is *not* an independent variable—it is a function of the centerline position and of the boundary conditions (see §4.3). Therefore, we explicitly transfer the torque acting on the material frame $\mathbf{d}_1^0$, to the coupled rigid body. Note that the rigid body's relatively large moment of inertia ensures that the explicit coupling is non-stiff.

The force acting on the material frame corresponds to the gradient of energy with respect to the *dependent* angles, θ^i. By the quasistatic material frame assumption, the torque, $\boldsymbol{\tau} = |\boldsymbol{\tau}|\,\mathbf{t}^0$, exerted by the rod on the body is equal and opposite to the torque exerted by the body on the rod. To derive the magnitude of the torque we turn to the principle of virtual work [Lanczos, 1986]. Holding the centerline fixed, consider a *virtual displacement*, $\delta\theta^0$, which varies the material frame about $\mathbf{t}^0$. Note that $\delta\theta^0$ also affects the body's orientation, due to the rod-body constraint. The work performed by the body on the rod equates to the change in the rod's stored energy: $|\boldsymbol{\tau}|\,\delta\theta^0 = (\mathrm{d}E/\mathrm{d}\theta^0)\delta\theta^0$. Since this holds for *any* $\delta\theta^0$, it follows that $|\boldsymbol{\tau}| = \mathrm{d}E/\mathrm{d}\theta^0$. In the special case, this yields

$$|\boldsymbol{\tau}| = \widetilde{\beta}(\theta^n - \theta^0) \ .$$

In the general case, we expand the full derivative as $\mathrm{d}E/\mathrm{d}\theta^0 = \sum_i (\partial E/\partial\theta^i)(\partial\theta^i/\partial\theta^0)$. By the quasistatic assumption, most terms in this summation are zero, leading to $|\boldsymbol{\tau}| = \partial E/\partial\theta^0$.

4.6 Implementation

4.6.1 Special case

For the special case of an uncurved elastic rod with isotropic bending response, we integrate the dynamics of the rod using the symplectic Euler method [Hairer et al., 2006] by including only the bending and twisting energy terms, and we enforce inextensibility of the centerline using the fast manifold projection method described in §4.5.2. The quasistatic material frame update requires only the determination of θ^0 and θ^n from the boundary conditions (see §4.3). Enforcing inextensibility avoids the instabilities that would otherwise be caused by including the stiff stretching energy as part of the formulation.

Algorithm 1 Uncurved, isotropic elastic rod simulation

Require: $(\mathbf{x}_0, \dot{\mathbf{x}}_0) \dots (\mathbf{x}_{n+1}, \dot{\mathbf{x}}_{n+1})$ // initial position/velocity of centerline

Require: $\underline{\mathbf{d}}_1^0$ // initial director for reference frame $\{\mathbf{t}^0, \underline{\mathbf{d}}_1^0, \underline{\mathbf{d}}_2^0\}$ at first edge

set quasistatic material frame from boundary conditions (§4.3)
while simulating **do**
    apply torque to rigid body // §4.5.3
    integrate rigid body // external library [Smith, 2008]
    compute forces on centerline
    integrate centerline // [Hairer et al., 2006]
    enforce inextensibility and rigid body coupling (§6.3)
    collision detection and response // [Spillmann and Teschner, 2008]
    update reference frame // (3.18)
    update quasistatic material frame (§4.3)
end while

4.6.2 General case

For the general case, which includes elastic rods with curved undeformed configuration and anisotropic bending response, we integrate the rod's dynamics using the implicit Euler time-stepping method (external forces may optionally be integrated explicitly). We use Newton's

method to solve

$$\mathbf{M}\Delta\mathbf{v} - h_k F_{\text{int}}\left(\mathbf{q}(t_k) + \Delta\mathbf{q}\right) = h_k F_{\text{ext}}\left(t_k, \mathbf{q}(t_k), \mathbf{v}(t_k)\right)$$
$$\Delta\mathbf{q} - h_k\Delta\mathbf{v} = h_k\mathbf{v}(t_k)$$

for $\Delta\mathbf{q} = \mathbf{q}(t_{k+1}) - \mathbf{q}(t_k)$ and $\Delta\mathbf{v} = \mathbf{v}(t_{k+1}) - \mathbf{v}(t_k)$, where $\mathbf{q} = (\mathbf{x}_0, \theta_0, \mathbf{x}_1, \theta_1, \cdots, \theta_n, \mathbf{x}_{n+1})^T$, and $h_k = t_{k+1} - t_k$. The lumped mass matrix $\mathbf{M}$ assigns each vertex the mass $m_i = \rho\left(V^{i-1} + V^i\right)/2$ of its Voronoi cell, where ρ is the (volumetric) density of the material and V^{i-1} and V^i are the volumes of the incident edges. To enforce the quasistatic equilibrium of the material frame (see §4.3), $\mathbf{M}$ assigns zero mass to the $\{\theta^j\}$ variables.

Newton update Each Newton iteration, we adapt the reference frame using by following the motion of centerline due to that iteration (see (3.17)); the attendant twisting $\Delta\underline{m}_i$ of the reference frame is given by parallel transporting $\underline{\mathbf{d}}_1^{i-1}$ from $\mathbf{t}^{i-1}$ to $\mathbf{t}^i$, rotating about $\mathbf{t}^i$ by angle $\underline{m}_i$, and measuring the angle $\Delta\underline{m}_i$ between the result and $\underline{\mathbf{d}}_1^i$: $\Delta\underline{m}_i = \angle\left(\mathrm{R}\left(\mathbf{t}^i, \underline{m}_i\right) \circ \mathbf{P}_i \cdot \underline{\mathbf{d}}_1^{i-1}, \underline{\mathbf{d}}_1^i\right)$.

4.7 Results

The interaction between twisting and bending modes produces surprising instability phenomena, such as spontaneous buckling of rods. These instabilities are important for a variety of applications, such as for modeling DNA. When applied to some of these phenomena, our model not only reproduces the correct behavior qualitatively, but in fact shows good convergence to analytical solutions (for those models where an analytical treatment is available in the literature). In the following we describe several example problems; refer to Table 4.1 for corresponding performance measurements.

4.7.1 Special case

Localized helical buckling When a naturally straight, isotropic rod is clamped at its two ends with one end rotated by a given angle, the rod buckles as the two ends are quasi-

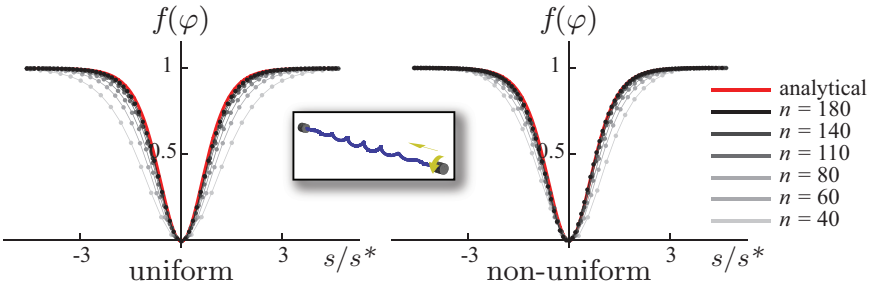


Figure 4.3: When a fixed angle of twist is imposed on a naturally straight, isotropic rod, the rod buckles into a helix with modulated amplitude (*inset*). Convergence of the envelope of this helix toward the analytical solution in the smooth case is observed for both a uniformly sampled rod (*left*) and a rod where one half is twice as refined as the other (*right*). The parameters for the simulation are: rod length $L = 9.29$, bending modulus $\alpha = 1.345$, twist modulus $\beta = .789$, clamp rotated by 27 turns, imposed axial shortening of 0.3 units, corresponding to a theoretical maximal deviation $\varphi_0 = 0.919$.

statically translated towards one another. In the smooth case, the buckling of a long rod under twist is described by an exact solution of the equations of equilibrium. This analytical solution describes nonlinear (finite amplitude) buckling away from the straight configuration and mixes bending and twisting effects. We studied convergence of the equilibria of our discrete model towards this solution in the limit of refinement. The geometry is shown in the inset of Fig. 4.3. We denote by s the arc length, $\mathbf{e}_x$ the unit vector parallel to the axis passing through the clamps, $\varphi(s) = \cos^{-1}(\mathbf{t} \cdot \mathbf{e}_x)$ the angular deviation of the tangent away from this axis, and $\varphi_0 = \max_s \varphi(s)$ the maximal deviation at the center of the pattern. The envelope of the helix is given by $f(\varphi(s)) = \tanh^2\left(\frac{s}{s^*}\right)$, where s/s^* is the dimensionless arc length given by $s/s^* = \left(\frac{\beta m}{2\alpha}\sqrt{\frac{1-\cos\varphi_0}{1+\cos\varphi_0}}\right)\ s$, which simplifies to $f(\varphi) = \frac{\cos\varphi - \cos\varphi_0}{1-\cos\varphi_0}$ (see, e.g., [van der Heijden and Thompson, 2000]). With a fixed loading geometry, we obtain convergence towards this analytical solution under refinement (see Fig. 4.3).

Michell's instability Another famous example of a rod becoming unstable when subjected to twist is Michell's instability, also known as Zajac's instability (see Fig. 4.4). When the ends of a naturally straight, isotropic rod are closed into a ring in a twist-free manner, the resulting shape is circular. When these ends are twisted by an angle θ^n before they

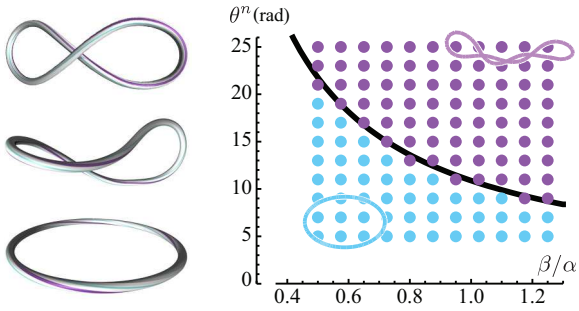


Figure 4.4: An isotropic, uncurved elastic rod is deformed into a ring with imposed internal twist θ^n. *Left:* above a critical value of θ^n, the planar, circular shape loses stability and buckles to a non-planar shape. *Right:* domain of stability of the circular shape with radius $R = 1$: simulations (dots) compared to theoretical threshold (black curve). Each dot corresponds to a simulation run with particular values of β and θ^n ($\alpha = 1$ and $n = 50$ are fixed), initialized with a slightly perturbed circular shape; dots are colored in light blue when the amplitude of the perturbation decreases in time (stable) and in purple when it increases (unstable).

are closed up, and when this angle is progressively increased starting from zero, the ring suddenly starts to writhe (buckle) out of plane, at a well-defined *critical* value of twist. This effect is a perfect illustration of the coupling of the twist with the shape of the centerline. This critical twist can be computed analytically [Goriely, 2006] as $\theta^n_c = 2\pi\sqrt{3}/(\beta/\alpha)$. In Fig. 4.4–*right*, we study the stability of the circular shape numerically for different values of the rod parameters and compare to the analytical prediction. Our model displays excellent agreement with the predictions of the smooth theory.

Instability of knots Although knots have been extensively studied as mathematical objects, the understanding of their mechanical behavior is far less advanced. We ran a simulation of a knotted rod with both ends clamped and were able to reproduce the typical equilibrium shape of the knot [Audoly et al., 2007], which is made of a large, almost circular loop connected to two flat tails by a braid region (see Fig. 4.1–*left*). We next twisted the ends of the rod and found that the shape of the knot first changes smoothly and then jumps at a critical value of the twist. This jump happens both for positive and negative applied twist, although the final shapes of the knot are qualitatively different (see Fig. 4.1). This fascinating behavior can be easily reproduced using a small tube of an elastic material (a

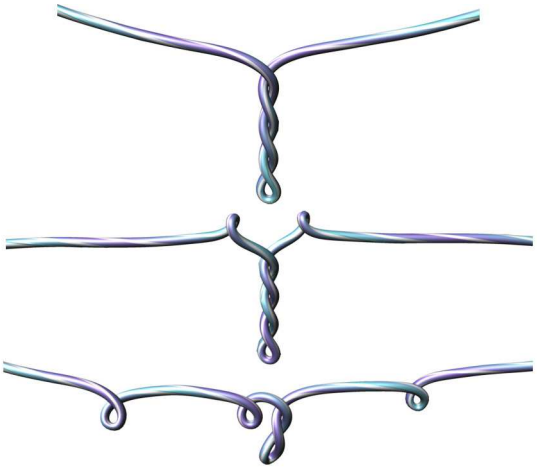


Figure 4.5: When the ends of a hanging elastic rod are twisted, it takes on structures known as *plectonemes*. The formation of plectonemes is governed by physical parameters, such as the twist rate, viscosity of the ambient fluid, and gravity.

silicone wire in our experiments), and we encourage the reader to try it!

Plectonemes A rod generally forms entangled structures called plectonemes when subjected to large twist. Typically, the pattern formed by a twist-driven instability such as Michell's instability grows from a weakly helical shape at short times to fully developed plectonemes at long time. Such structures have been well-studied, for instance in the context of DNA supercoiling [Yang et al., 1993]. In this experiment, we started with a naturally straight rod and deformed it into a parabola in a twist-free manner. Fixing the positions of the endpoints of the rod and progressively increasing twist, we find that the rod starts to writhe out of the plane. Letting the instability develop fully, we observe plectonemes (see Fig. 4.5). Depending on the viscous drag and gravity, we found that one or many plectonemes can be obtained. Single plectonemes have been described both analytically [van der Heijden et al., 2003] and numerically [Goyal et al., 2008] in several papers; we observed an interesting phenomenon of plectonemes merging at long times, which has seemingly not yet been studied.

The simulation of plectonemes requires the treatment of rod self-contact; for the state of the art, refer to the treatment by Spillmann and Teschner [2008].

4.7.2 General case

The coupling of twisting and bending modes of naturally curved or anisotropic rods is fairly more complex than in the isotropic case, as demonstrated by in following experiments.

Asymmetry of twist The combination of anisotropy and non-zero rest curvature can result in a non-uniform distribution of twist along the rod. In Fig. 4.2, we show this effect for a rod with anisotropic cross section, whose centerline is either V-shaped or semi-circular in the reference configuration. We clamp one end of each of these rods and twist the other. In the V-shaped case, we observe that twist is first confined on one half of the rod—it takes a significant amount of twist before twist manages to "jump over" the kink in the middle of the rod. A similar phenomenon occurs with the semi-circular shape, but it is less marked. In either case, the twist concentrates near the end that is rotated, although the rod properties are uniform along its length; this symmetry-breaking points to the fact that that the equations governing the quasi-static twist are nonlinear.

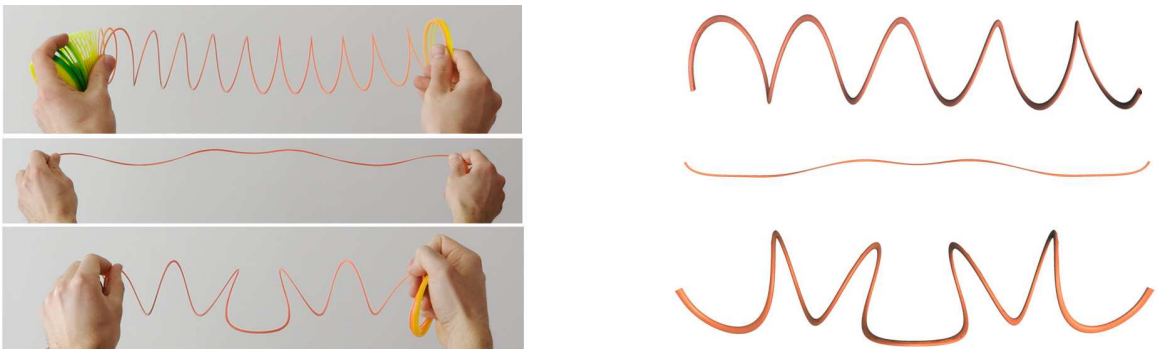


Figure 4.6: Starting from the natural shape of a Slinky® *(top)*, we first remove writhe by pinching the flat cross-section between two fingers and traveling from one end to the other, arriving at a fully stretched, almost flat ribbon *(middle)*. By bringing the ends together, a persistent perversion forms *(bottom)*, whose shape is surprisingly different from the natural shape.

Helical perversion: Perversion is a classical pattern that can be observed in naturally curved rods. It consists of a junction between two helices with opposite chiralities. It has been described by Darwin in the tendrils of climbing plants and can be often observed

in tangled phone cords [Goriely and Tabor, 1998]. Perversions can be produced by first flattening a helical spring such as a Slinky® into a flat, straight ribbon by pulling on both of its ends (see Fig. 4.6–*middle*); next, by progressively releasing its ends from this straight configuration, one obtains a shape made of two mirror-symmetric helices (see Fig. 4.6–*bottom*). The final shape is surprisingly different from the natural one as the chirality has been reversed in a half of the spring, as revealed by comparison with Fig. 4.6–*top*. The existence of perversions is due to the nonlinear behavior of the rod—perversions belong to a general class of solutions that can be derived in nonlinear analysis by connecting two competing equilibrium configurations, which are here the right-handed and left-handed helices in the presence of natural curvature.

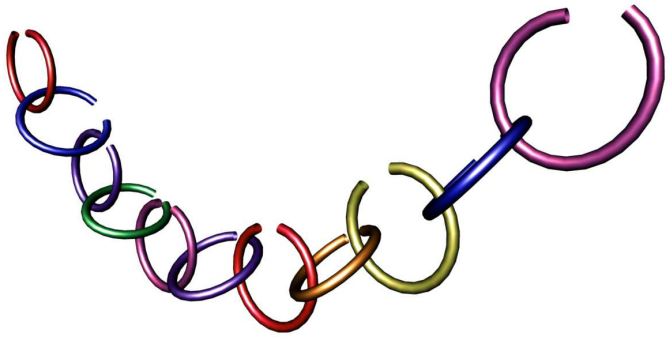


Figure 4.7: A chain consisting of curved elastic rods as links hangs under the influence of gravity. The material parameters for each rod are $\alpha = 2$ and $\beta = 2$, with each link in the chain having a radius of approximately 1 unit.

Hanging chain: free boundaries In Fig. 4.7, we show a chain hanging from its two ends under the influence of gravity. Each link is an elastic rod with curved undeformed configuration with stress-free boundary conditions on the material frames. The two ends are pulled apart until the chain breaks and the links fly apart. Even though we are not applying a twist to any of the links in the chain, the material frame is still needed in order to represent the undeformed curvature of the rod—recall that $\overline{\boldsymbol{\kappa}}_i$ is defined as a 2-vector in material coordinates. In larger-scale examples involving character-scale, yarn-based cloth simulations, Kaldor et al. [2010] likewise require the formulation involving the material frame in order to represent curved undeformed configurations for the yarn.

Figure 4.8: Using a time-parallel reference frame enables our implicit solver to operate on a banded linear system, leading to fast, stable simulations.

Hair dynamics The implicit solver afforded by the time-parallel formulation results in a multi-fold performance benefit over our explicit implementation based on the space-parallel formulation. For a 300 frame simulation of an elastic rod with 200 vertices, the total time spent in computing the dynamics of the rod (without collisions) was 0.488 seconds using our implicit code and 4.22 seconds using our explicit code. Using the implicit code, the stable time-step was 70× larger, which allowed for far fewer sub-steps per frame. From a production standpoint, such as for creating scenes like that depicted in Fig. 4.8, the implicit formulation offers two advantages over existing production-quality alternatives: it is based on a continuum model, which greatly aids in setting up material parameters and consistent visual styles, and it is stable, allowing the user to focus on how the simulation looks without simultaneously adjusting discretization parameters to maintain stability.

4.7.3 Rigid body coupling

When coupling rods with rigid bodies, the transfer of energy between these systems results in complex motions—in particular, through the interplay between bending and (non-uniform) twisting of the rod and the translational and rotational moment of the attached mass. In order to validate our model, we have in particular tested its ability to faithfully reproduce real-world experiments.

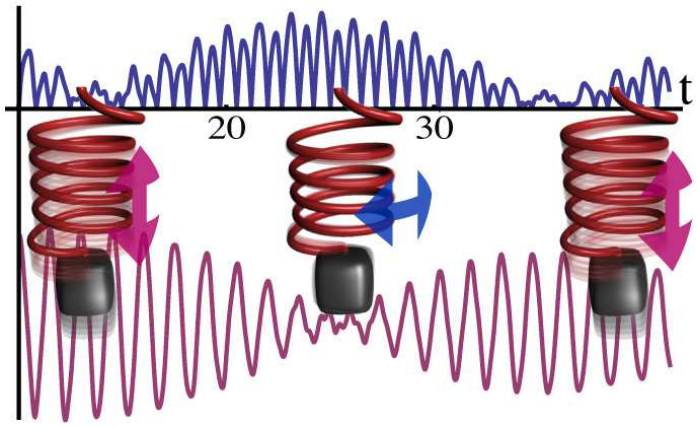


Figure 4.9: Due to a weak coupling between the bending and twisting modes of a stretched spring, the energy of the system is transferred back and forth between the translational (red) and angular (blue) modes of oscillations.

Wilberforce pendulum This experiment reveals fascinating aspects of how bending and twisting interact and thereby affect the motion of an attached rigid body. Consider a helicoidal spring (an isotropic rod whose reference state is a helix) whose upper end is fixed, and attach a rigid body to its lower end. The weight of the body stretches the spring when the whole system is at rest. Moving the mass slightly upwards from this equilibrium and releasing it leads to vertical oscillations. Progressively, the system switches to twist oscillations and the vertical ones are extinguished; later, the twist oscillations start to decrease and the vertical ones reappear, and so on (see Fig. 4.9). The nonlinear behavior of the spring, which is captured accurately by our model, is responsible for this energy transfer between the two modes [Sommerfeld, 1964]: stretching a spring affects its eigenmodes. Note the presence of stationary waves in the spring, which are also clearly visible in our simulation movie.

Rigid bodies for rigid couples Coupling rods with rigid bodies opens the possibility for complex simulations—in particular, for simulating tree-like one-dimensional configurations with several T-junctions. In Fig. 4.10, we show an example of such coupling, where we additionally used external forces (wind and gravity) to increase the dramatic effect. Here the mass of the rigid bodies (but not necessarily their spatial extension) can be tuned to

Figure 4.10: Rigid bodies connect multiple rods together at a single point (reference configuration shown in inlay). Using this method, we can simulate a tree bending and twisting in response to a strong arctic wind. Note that without resistance to twist, the tree would start spinning due to vortices in the wind.

achieve a variety of dramatic effects. This example shows the power of coupling rods with rigid bodies, indicating the attractiveness of this approach to be used in a wide range of graphics applications, such as for simulating plant motion, or the skeletal dynamics of rigged characters.

Figure	Vertices	Time step	Forces	Contact	Quas. update	Fast proj.	Total
4.1	60	4.0	0.026	0.05	0.021	0.21	0.31
4.2	49	1.0	0.025	0.018	0.021	0.12	0.18
4.3	75	2.0	0.043		0.1	0.2	0.34
4.4	67	50.0	0.039		0.096	0.28	0.42
4.5	275	1.0	0.13	0.19	0.17	0.42	0.92
4.7	31	1.0	0.016		0.13	0.13	0.2
4.9	20	1.0	0.0036		0.0043	0.019	0.027
4.10	2158	0.1	0.87		0.64	21.0	22.0

Table 4.1: This table summarizes timing information (in milliseconds per simulation step) for examples depicted in the figures, as measured on a single-threaded application running on a 2.66GHz Core 2 Duo. For examples run without collision detection, collision timings are omitted.

4.8 Discussion

The formulation presented allows us to impose boundary conditions on the material frame at any edge along the rod. We have presented boundary conditions that arise due to

explicitly clamping certain edges or coupling them to rigid bodies. Along this line, the effect of frictional contact on the material frame, in particular allowing contact constraints to dictate boundary conditions for the material frame, would also be interesting to consider.

For enforcing constraints within our elastic rod model (see §6.3), our use of the fast manifold projection method may cause the energy in the system to not be conserved. In many applications, the energy behavior of the system is of interest, in which case constraint-enforcement methods with known good energy behavior could be used (for an overview of such methods, see, e.g., [Hairer et al., 2006]).

Chapter 5

Viscous threads

In this chapter, we present a continuum-based discrete model for thin threads of viscous fluids by drawing upon the Rayleigh analogy to elastic rods, demonstrating canonical coiling, folding, and breakup in dynamic simulations. Because the model is based on the Rayleigh analogy, the model for elastic rods presented in §4 can be re-purposed for simulating viscous threads with very little modification. The result is a fast, implicit treatment of viscous threads that reproduces a variety of physical phenomena.

5.1 Introduction

Viscous threads, such as a thin thread of honey, display a variety of behaviors that are challenging to accurately capture with existing simulation techniques. For example, a thread poured steadily onto a moving belt creates a sequence of "sewing machine" patterns (see Fig. 5.1) whose numerical reproduction has not been reported to date. Understanding the motion of a viscous thread is a gateway to simulation tools whose utility spans film-making, gaming, and engineering. For example, in over 30% of worldwide textile manufacturing processes, threads of viscous liquid polymers (often incorporating recycled materials) are entangled to form nonwoven fabric used in baby diapers, bandages, envelopes, upholstery, air ("HEPA") filters, surgical gowns, high-traffic carpets, erosion control, felt, frost protection,

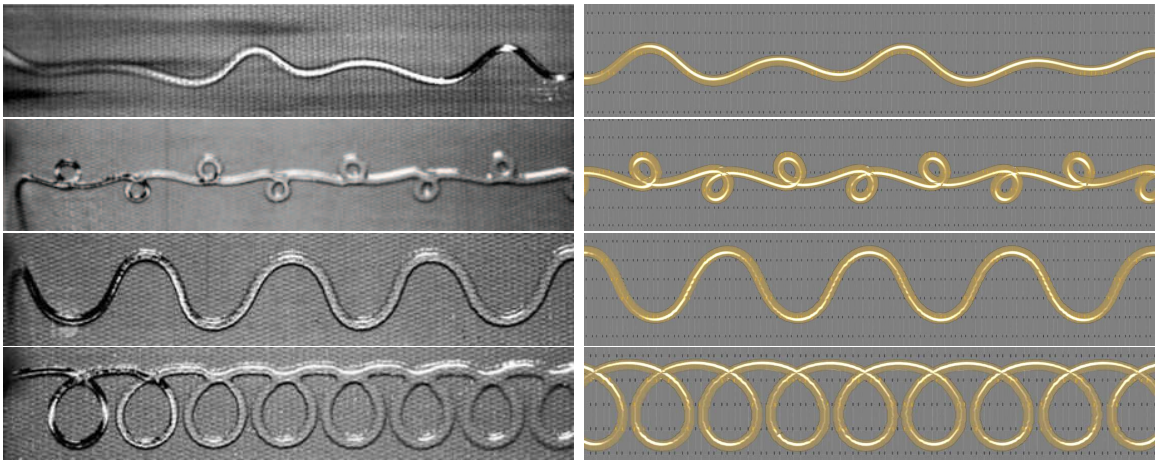


Figure 5.1: A thin thread of viscous fluid is poured onto a moving belt, creating a dazzling array of intricate patterns. Simulations using our model reproduce this rich and complex behavior. Translucent thread: experiment [Chiu-Webster and Lister, 2006]; gold thread: simulation.

and tea sachets [Andreassen et al., 1997].

While in theory, it is possible to accurately compute the motion of a viscous thread using a general, volumetric fluid simulator, there are no reports of successes to date, perhaps because the resolution needed for a sufficiently accurate reproduction requires prohibitively expensive runtimes. In contrast to volumetric approaches, we model viscous threads by their formal analogy to elastic rods. The motion of a viscous thread is governed by the material's resistance to stretching, bending, and twisting *rates*, coupled with the effects of surface tension. Thus, with the exception of surface tension, which generally plays a negligible role for elastic rods, an existing implementation of stretching, bending, and twisting for an elastic rod can be easily repurposed for simulating a viscous thread [Bergou et al., 2008, 2010].

5.2 Related work

Lagrangian approaches, such as ours, are adept at tracking a moving material that occupies a small fraction of space. While we focus specifically on thin threads, others treat Lagrangian simulations of more bulky viscoelastica using a variety of representations such

as points [Müller et al., 2004; Gerszewski et al., 2009], particles [Miller and Pearce, 1989; Terzopoulos et al., 1991; Steele et al., 2004; Clavet et al., 2005], smoothed particle hydrodynamics [Desbrun and Gascuel, 1996; Stora et al., 1999; Müller et al., 2003; Rafiee et al., 2007; Chang et al., 2009], and meshes [Terzopoulos and Fleischer, 1988; O'Brien et al., 2002]. Wojtan and Turk [2008] specifically consider thin-featured viscous and plastic materials, embedding a high resolution surface in a coarse tetrahedral mesh for fast, physically plausible motion.

Eulerian treatments ensure uniform spatial sampling of large volumes, such as those needed for smoke [Foster and Metaxas, 1997; Stam, 1999; Fedkiw et al., 2001]. For liquids, the free surface must be tracked [Foster and Fedkiw, 2001]. Viscoelastic materials require accumulated strain advection and additional elastic force terms [Goktekin et al., 2004; Irving, 2007]. Thin features require special attention: Losasso et al. [2004] employ an octree data structure to capture small-scale visual detail. Hong and Kim [2005] consider the breakup of a liquid sheet and spurting of a liquid jet. Kim et al. [2009] extend the vortex sheet method to enhance fine details such as wiggling fluid sheets.

The Eulerian approach which most complements our Lagrangian setup is presented by Batty and Bridson [2008], who propose an elegant variational principle to enforce the free surface boundary condition, so that viscous fluids coil and fold. While they compute plausible motion for even the largest volumes of thick fluid, our method accurately reproduces the coiling of even the thinnest 3D fluid threads.

In the mechanics literature, the closest Eulerian approach is that of Oishi et al. [2008], who use a finite difference scheme based on an updated marker-and-cell method to study the buckling of a viscous jet based on the numerical solution of the 3D Navier-Stokes equations with free boundaries. Bonito et al. [2006] show results pertaining to 3D jet buckling based on an approach that combines finite elements with the method of characteristics.

Derivation of the reduced viscous-thread model begins with Trouton [1906], who identifies the effective modulus of a viscous string in tension. Entov and Yarin [1984] derive the full *viscous thread* model for 3D viscous flows, adding the effect of twist and curvature; the model is further explored in subsequent decades [Dewynne et al., 1992].

Numerical solution of 3D thin viscous threads is, to our knowledge, considered only once for general configurations, in the context of interactive painting [Lee et al., 2006]. This system produces impressive Jackson Pollock-style paintings, using a heuristic coupling of the large-scale motion of the centerline to the equations of Eggers and Dupont [1994] describing the flow inside the jet. By contrast, our model is based on the reduced coordinate formulation, and we do not need to solve numerically for the flow inside the thread. While we pursue the general case, others develop numerics specialized to specific hypotheses: Skorobogatiy and Mahadevan [2000] consider the 2D, time-dependent problem of folding, where twist is identically zero. Ribe [2004] solves the nonlinear boundary-value problem for viscous coiling in the corotating frame, where the shape becomes stationary. Panda et al. [2008] present dynamic simulations of stretched viscous filaments in a geometry where bending and twist can be neglected.

The sagging of a viscous thread held at two ends under gravity was first studied by Teichman and Mahadevan [2003]. More recently, Le Merrer et al. [2008], showed that the behavior of the thread can be categorized into different regimes depending on the physical parameters of the system.

Coiling is a canonical problem in the study of viscous threads and remains poorly understood. Taylor [1968] qualitatively explains the coils by analogy with the buckling of an elastic column in compression. Ribe et al. [2006] presents a detailed comparison of simulations of steady coiling to experiments. The problem is essentially nonlinear and traditional linear stability analyses have been limited to the description of nascent coils [Radev et al., 1987]. Scaling analyses can successfully describe the coiling behavior in some regimes [Mahadevan et al., 1998]. Recently, the problem of steady coiling in a rotating frame has been formulated and solved numerically [Ribe, 2004], showing excellent agreement with experimental results in a wide range of parameters [Ribe et al., 2006]. Simply replacing the table by a moving belt, Chiu-Webster and Lister [2006] have unveiled a variety of new experimental patterns which have remained, until now, numerically unreproduced.

5.3 The fluid-as-threads paradigm

Our treatment of viscous threads is based on Rayleigh's observation that a viscous formulation derives from an elastic formulation when velocities replace positions, and strain rates replace strains [Strutt, 1945]. This model for viscous threads is justified under assumptions similar to the elastic rod model: the thickness is much smaller than any longitudinal length scale and the curvature of the centerline is everywhere much smaller than the inverse of thickness. Computationally, the thread model offers tremendous advantages over a full-blown fluid treatment as it encapsulates all the small-scale details of the flow into effective parameters, thereby allowing the use of mesh sizes significantly larger than the thickness of the thread without loss of accuracy.

5.3.1 Dissipative potential

Smooth setting The general form of the elastic potentials defined in §4.2 can be written in terms of a geometric measure of deformation ϵ (elongation, material curvature, or twist) as

$$E_{\text{el}} = \frac{1}{2} \int k_{\text{el}} (\epsilon - \overline{\epsilon})^2 \, \mathrm{d}\, s \ .$$

Elastic forces are then computed by taking variations of the elastic potential with respect to the material coordinates. By analogy with the elastic case, the viscous model can be derived starting from the *dissipative potential*

$$E_{\text{vis}} = \frac{1}{2} \int k_{\text{vis}} \, \dot{\epsilon}^2 \, \mathrm{d}\, s \ ,$$

defined in terms of the rate of change of the geometric deformations. The viscous forces are then computed by taking variations of the dissipative potential with respect to the *time derivatives* (velocities) of the material coordinates. Compared to the elastic regime, this amounts to replacing internal elastic forces by internal viscous ones in Newton's equations of motion.

Discrete setting Radovitzky and Ortiz [1999] followed by Kharevych et al. [2006] introduced a variational formulation for viscosity, writing viscous forces as the gradient of the discrete dissipative potential

$$E_{\text{vis}} = \frac{1}{2}\sum_i (k_{\text{vis}})_i \left(\frac{\epsilon_i(t_{k+1}) - \epsilon_i(t_k)}{h_k}\right)^2 .$$

The parenthesized expression is the discrete strain rate. Here, by creating an analogue to the elastic regime, strains at time t_k refer to the undeformed configuration, while strains at time t_{k+1} refer to the deformed one. Considering variations with respect to material coordinates scaled by the inverse of the time-step $h_k = t_{k+1} - t_k$ one obtains for the viscous force

$$F_{\text{vis}} = -\sum_i \frac{(k_{\text{vis}})_i}{h_k} \left(\epsilon_i(t_{k+1}) - \epsilon_i(t_k)\right) \frac{\partial \epsilon_i(t_{k+1})}{\partial \mathbf{x}} .$$

By folding the division by h_k directly into the viscous stiffnesses (see below), this formulation of viscosity requires only a trivial modification to an elastic implementation: before each time-step, we set the undeformed to the current strain (see §4.6).

5.3.2 Viscous force stiffnesses

In the case of viscous threads, we assume a cylindrical shape of the fluid along the edges, with radius given by $(a^j = b^j = r^j)$, and we set $r_i = (r^{i-1} + r^i)/2$ at vertices. Thus, the cross-sectional area at edges and vertices is given by $A^j = \pi(r^j)^2$ and $A_i = \pi r_i^2$, respectively. Requiring incompressibility of the fluid, the stiffnesses then take the form [Ribe, 2004]

$$k_{\text{s}}^j = \frac{3\mu A^j}{h_k}, \quad \beta_i = \frac{\mu A_i r_i^2}{2h_k}, \quad \mathbf{B}_i = \frac{3\mu A_i r_i^2}{4h_k}\mathbf{I}_2,$$

where μ is the *dynamic viscosity* of the fluid, h_k is the time-step size, and $\mathbf{I}_2$ is the 2×2 identity matrix.

5.4 Departure from elastic rod model

In addition to the internal viscous forces, the model for a viscous thread must take into account some additional considerations, which are not needed for the elastic case.

5.4.1 Conservation of volume

Throughout we assume that the fluid is incompressible with uniform density, leading to the volume of the fluid being conserved. In the discrete setting, we consider the volume to be an edge-based quantity given by $V^j = \pi(r^j)^2 \left|\mathbf{e}^j\right|$, corresponding to the volume of the edge-based cylinder. We enforce volume conservation by updating r^j at the end of each time-step using $r^j = \sqrt{V^j/(\pi \left|\mathbf{e}^j\right|)}$, which makes the radius of the edge a function of the position of the two incident vertices. Notice that since the stiffnesses k_s, β, and $\mathbf{B}$ all depend on the radius, they must be updated, too.

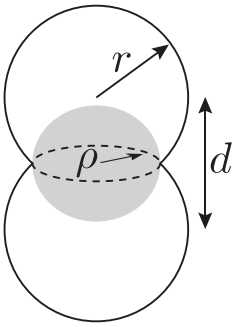
5.4.2 Surface tension

Viscous threads are set apart from elastic rods by the fact that *surface tension*, which reflects the material's desire to minimize its overall surface area, cannot safely be neglected. The energy associated to surface tension is proportional to the surface area of the fluid, with constant of proportionality γ. In the discrete setting, we consider surface area (not to be confused with cross-sectional area) to be a vertex-based quantity, $\mathcal{A}_i = \pi\left(r^{i-1} + r^i\right)\sqrt{l_i^2 + \left(r^i - r^{i-1}\right)^2}$, corresponding to the lateral surface area of a truncated cone with bottom and top given by the cross-sectional circles at edge midpoints. Using the relation between radius r^i and volume V^i from §5.4.1, we can express $\mathcal{A}_i$ in terms of V^{i-1}, V^i, $|\mathbf{e}^{i-1}|$, and $\left|\mathbf{e}^i\right|$ only. Then, $E_\mathrm{surf} = \sum_{i=1}^n \gamma\mathcal{A}_i$ is the requisite energy associated to surface tension.

5.4.3 Merging

When two threads graze, surface tension attracts them, trying to minimize lateral boundary area, whereas viscosity resists their relative motion. While an exact account of merging would require a full 3D simulation, we derive a qualitatively correct treatment here.

Geometry We consider the merging of two balls of identical radii, $r = (r_1 + r_2)/2$, centered at points along the centerline of closest approach in the simulation, separated by a distance $d(t)$. We use r_1 and r_2 to denote the cylinder radii of the current points of closest approach. Let $\mathcal{V}(d)$ be the union of the two spheres, $\partial\mathcal{V}(d)$ its boundary (two portions of a sphere), $\mathcal{A}(d)$ the boundary's area, and $\rho = \sqrt{r^2 - (d/2)^2}$ the radius of of the intersection region, as depicted in the incident figure.



Physics The contribution of surface tension to the merging force derives from the potential energy $\gamma\,\mathcal{A}(d)$; this yields a force $\gamma\,(-\mathcal{A}'(d))$. The second factor is purely geometrical and can be calculated as $(-\mathcal{A}'(d)) = 2\pi\,r$, which happens not to depend on d. On the other hand, the viscous flow involves a typical velocity $\dot{d}$ in a region whose volume is comparable to that of a sphere of radius ρ, shown in light gray in the picture. The power dissipated by this viscous flow can be estimated as $\mu\,\frac{4\pi}{3}\rho^3\,(\dot{d}/\rho)^2$ and by division with respect to $\dot{d}$ we obtain the viscous contribution to the merging force. Summing up the two contributions, we obtain the attractive merging force $F_{\mathrm{m}} = 2\pi\,\gamma\,r - \mu\,\frac{4\pi}{3}\,\rho\,|\dot{d}|$ acting along the direction of the points of closest approach.

5.4.4 Adaptivity

Viscous threads tend to stretch significantly, necessitating remeshing to avoid poor sampling (refer to Spillmann and Teschner [2008] for adaptation of elastic rods). We subdivide any edge j whose length exceeds a prescribed discretization threshold into two edges and insert a new midpoint vertex. The position of the vertex is chosen by interpolating the position of the four vertices $\mathbf{x}_{j-1}, \mathbf{x}_j, \mathbf{x}_{j+1}, \mathbf{x}_{j+2}$ using a cubic Lagrange polynomial, and the velocity of the vertex is chosen to conserve the centerline's momentum. The newly created edges are assigned new radii, with the *ratio* of the radii chosen by interpolating the radii of the three edges r^{j-1}, r^j, r^{j+1} using a quadratic Lagrange polynomial and absolute value chosen to conserve total volume.

In addition to remeshing to avoid excessive stretching, we consider topological changes

caused by the breakup of the thread. We model this by removing any edge whose radius falls below a threshold.

5.5 Results

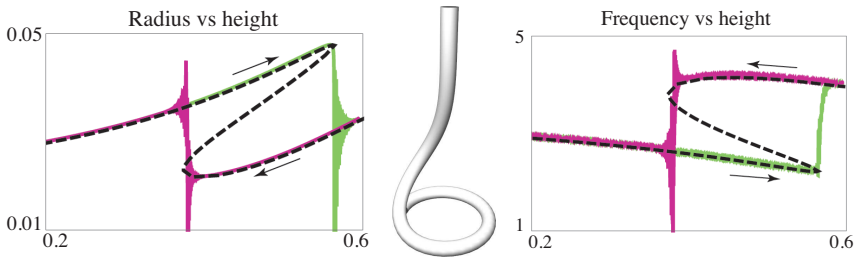


Figure 5.2: A viscous thread falling onto a table coils (*middle*) with the radius and frequency determined by the drop height. We compare computed stable radius (*left*) and coiling frequency (*right*) to that predicted by theory from Ribe [2004] (*dashed lines*) for different drop heights (*horizontal axes*), first by slowly increasing the simulated height (*green line*) and then by decreasing (*purple line*).

Viscous coiling When honey pours from a bottle, the tail of the thread rapidly coils as it hits a table. The radius and frequency of the coil formation depend on the height of the bottle (see Fig. 5.2). Our simulation agrees with results predicted by Ribe [2004] assuming steady coiling, and validated by him against experiments. In addition, our simulation shows a hysteretic effect also seen in real-world experiments [Ribe et al., 2006]: As the height of the container is slowly changed by first moving it upward and then back down, the simulation shows different stable coiling modes for the same heights. **Physical parameters:** volumetric density $\rho = 5 \times 10^{-4}$, viscosity $\mu = 0.2$, gravity $g = 9.81$, surface tension $\gamma = 0$, diameter of the thread at the container $d = 0.09$, speed of thread as it leaves container $U_0 = 0.615$. **Plot units:** nondimensional units measured with respect to $H^* = \left(\mu/(\rho^2 g)\right)^{1/3}$ for length and $\Omega^* = \left(g^2\rho/\mu\right)^{1/3}$ for frequency.

Viscous sewing machine Replacing the steady table by a moving conveyor belt yields a striking array of patterns and, as the belt speed or drop height is changed, transitions between coiling and meandering regimes [Chiu-Webster and Lister, 2006; Morris et al., 2008].

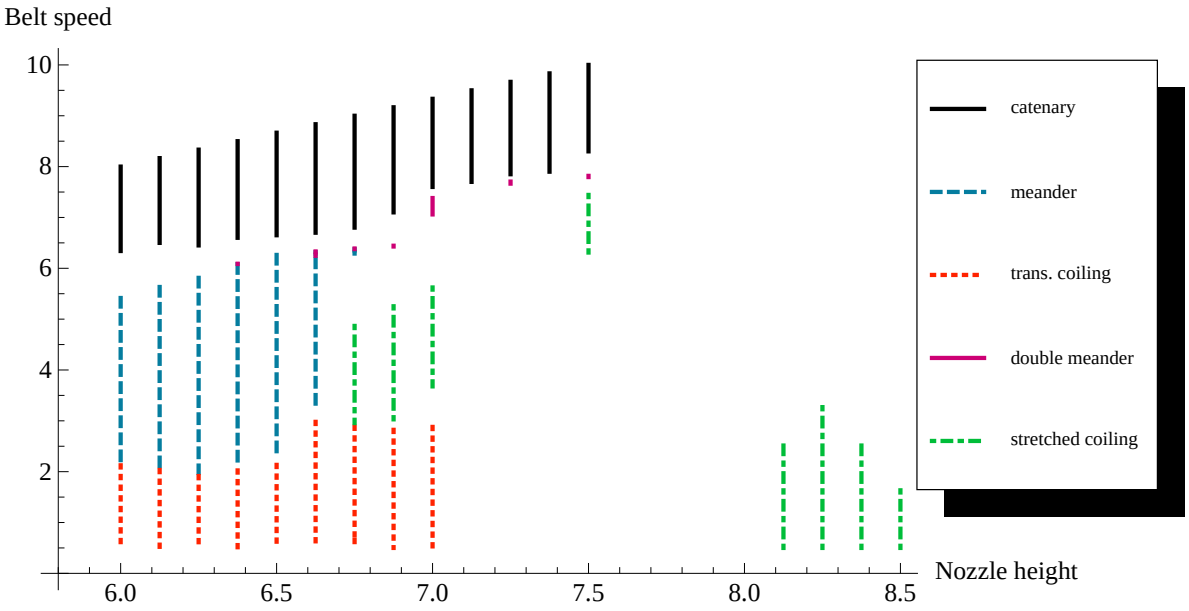
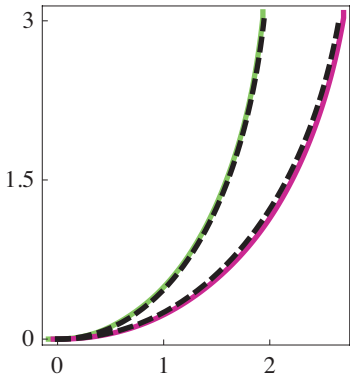


Figure 5.3: This plot shows the patterns observed for large nozzle heights in the fluid-mechanical sewing machine example (cf. the plot from Morris et al. [2008]). For the blank ranges in the plot (e.g., nozzle height 7.125–8.0 and belt speed 0–6) , we observed either disordered behavior or a superposition of multiple patterns.



For sufficiently high belt speed, the thread forms a catenary, whose shape is governed mainly by the interaction of gravity, axial viscosity, and surface tension. The incident figure shows the analytical result, derived by Chiu-Webster and Lister [2006], as a dashed black line both with (*right*) and without (*left*) surface tension. Our simulation accurately reproduces both cases (green corresponds to no surface tension and purple to nonzero surface tension). **Physical parameters:** drop height: 3 cm, belt speed: 0.7 cm/s, density $\rho = 0.996\,\mathrm{g/cm^3}$, viscosity $\mu = 273.6\,\mathrm{g/cm \cdot s}$, surface tension $\gamma = 10\,\mathrm{g/s^2}$, gravity $g = 100\,\mathrm{cm/s^2}$, diameter at nozzle $d = 0.8\,\mathrm{cm}$, speed of flow at nozzle $U_0 = 0.06\,\mathrm{cm/s}$.

As the belt speed is lowered, the thread's tail draws distinct patterns on the belt (see Fig. 5.1 and Fig. 5.3). Our simulation reproduces many of the observed patterns, which have never

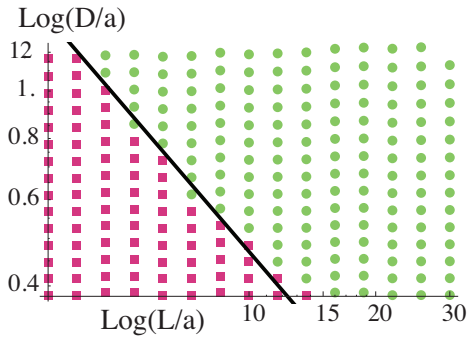
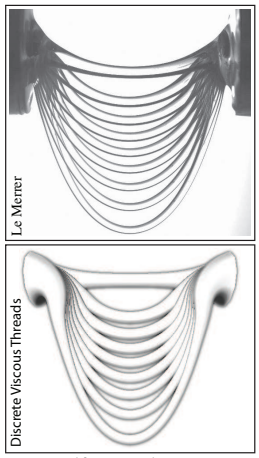


Figure 5.4: As a thread held at two ends falls under gravity, it forms a catenary (green circle) or U (purple square). The boundary fits the law $L\,D = 4.7\,a^2$ in accordance with the theory of Le Merrer et al. [2008]. L: thread length; D: diameter; a: capillary length

been numerically reproduced previously. **Fluid parameters:** see [Morris et al., 2008].

Surface tension and gravity competition When a viscous filament is suspended horizontally between two plates and allowed to sag under gravity, it tends to evolve into one of two shapes: (i) a catenary, which is characterized by near-uniform strain along the centerline at each point in time and a steady decrease in the height of the middle of the thread; or (ii) a "U" shape, where the center of the thread falls a small distance and then stops. In the latter case, surface tension drains fluid away from the center and towards the ends, increasing the strain at the thread's center until it snaps. Our simulation agrees with experiments of Le Merrer et al. [2008] in qualitative behavior, trajectory of the centerline's midpoint, and phase diagram (see Fig. 5.4). **Physical parameters:** oil density $0.907\,\mathrm{g/cm^3}$, viscosity $100\,\mathrm{g/cm\cdot s}$, surface tension $20\,\mathrm{g/s^2}$, plates $2.5\,\mathrm{cm}$ apart, thread diameter at centerline midpoint $0.175\,\mathrm{cm}$ for catenary and $0.033\,\mathrm{cm}$ for U.



Implicit vs. explicit Viscosity (damping) in a physical system results in stiff equations of motion [Hauth et al., 2003], in turn typically imposing stringent limits on the stable time-step. Indeed, for our simulations of viscous materials, we observe severe instabilities under explicit integration. Our implicit implementation allows for four orders of magnitude larger time-steps than our corresponding explicit implementation. Despite the need to solve one or more linear systems each time-step, the vastly fewer number of required steps

translates into three orders of magnitude faster runtimes (see Fig. 5.5 and Table 5.1). The balance in favor of an implicit approach is significantly boosted by the structure of the time-parallel formulation's Hessian, which is symmetric and banded. This tames both storage and computational complexity: the number of nonzero entries that must be computed and stored grows linearly with the number of vertices, and the linear system required for time-stepping can be solved in $O(n)$ time using standard routines from LAPACK.

Figure	Vertices	Time step	Fastest frame	Slowest frame	Average frame
4.8	54460	10ms	23.5s	3m 46s	1m 17s
5.1	180	10ms	0.241ms	59.7ms	16.8ms
5.2	40–350	40ms	0.05ms	35.1ms	3.64ms
5.5	200–1500	0.208ms	0.292ms	0.916s	0.202s
5.4	51	1ms	3.84ms	56.9ms	25.5ms

Table 5.1: This table lists the computational cost for representative examples of the results depicted in the figures. Timings for the hair simulation (Fig. 4.8) were measured on a multi-threaded application running on a 6 core CPUs running at 3.07GHz. The rest of the timings were measured on a single-threaded application running on a 3.166GHz machine.

Figure 5.5: The implicit formulation presented here dramatically reduces the computational time required to simulate scenes such as this.

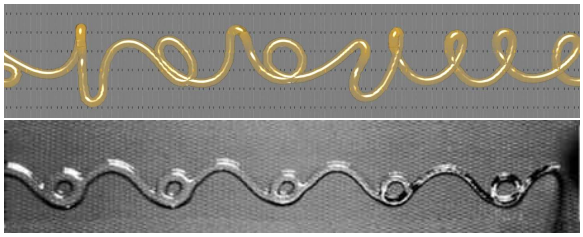


Figure 5.6: Some of the patterns for the fluid-mechanical sewing machine were observed only transiently, such as the "W" pattern shown here. We believe in this case that thread-thread contact may play an important role in forming this pattern, and a more rigorous approach for handling thread-thread contact could lead to more patterns being successfully reproduced.

5.6 Discussion

While we have been able to successfully validate both the elastic rod and viscous thread model against a wide range of theoretical as well as experimental results, there are limitations imposed by our assumptions. We assume that the viscous fluid is Newtonian and that its configuration is well represented using an adapted framed curve. When these assumptions are not valid, as happens for instance if the jet becomes very thick, the formulation presented here is no longer expected to produce valid results.

We additionally assume that the cross section of the thread remains circular (unlike for elastic rods) because of the effect of surface tension, so we neglect any motion of the fluid that would act to change the cross section. This assumption can be violated, for example, when the thread is laid down on a table and flattens out over time due to contact and gravity. In more general terms, we consider the simulation of *anisotropic* viscous threads or viscous *sheets* an interesting direction for future work.

In the viscous sewing machine experiment, we observed some patterns only transiently (see Fig. 5.6); for some belt speeds and nozzle heights, we observed a superposition of patterns and attendant beating frequency (see Fig. 5.3). It is not yet fully understood, both in experiment and in numerics, how the system chooses one specific pattern when several of them are in competition. We hope that numerical tools such as those developed here can help us to gain an understanding of these phenomena.

Chapter 6

Directable thin shells

Simulating thin, flexible materials often means giving up artistic control, yet manually animating their fine folds and wrinkles is an arduous task. How can we provide simultaneous artistic control *and* physical realism for materials like cloth, leather, or metal?

In this chapter, we present a process called *tracking*, which begins with a rough animation already set by the artist and uses physical simulation to add fine-scale details without deviating from the artist's intentions. The artist sets the scale of features to be left intact, and our solver computes the equations of motion at the remaining finer scales.

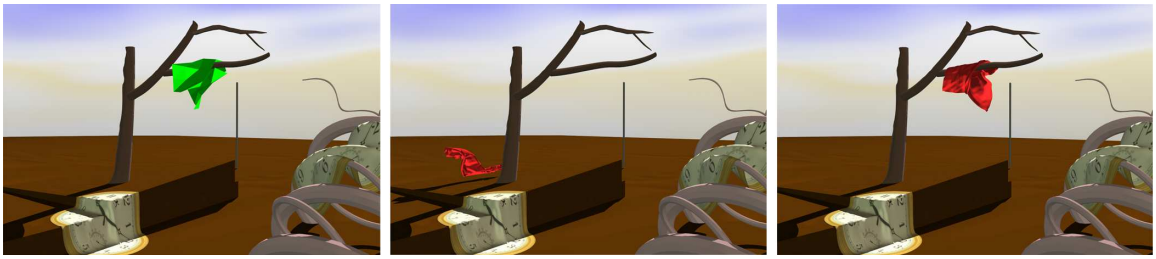


Figure 6.1: Fast simulations with a low-resolution mesh *(left)* enable the technical director to quickly iterate and improve on the setup of a physical simulation. When design is over and final production begins, an attempt to reuse the setup with higher-resolution mesh *(middle)* vividly justifies the usual disclaimer that "past performance is no guarantee of future results." Tracking *(right)* ensures that the output performs as "predicted" by the preview—more precisely, as *prescribed* by the coarse input motion.

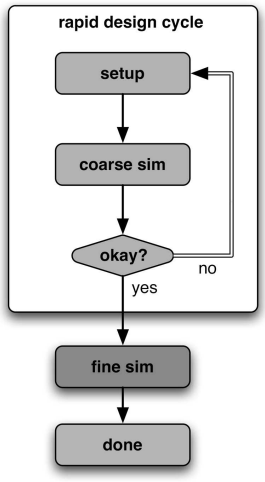
6.1 Introduction

While artistic freedom and mathematical determinism appear to be opposing forces, we aim to find a synergy between them in order to combine the expressiveness and freedom of hand-crafted motion with the realism and facility of physically simulated motion. To achieve this, we present a process called *tracking*, which takes as input a rough animation or simulation and enhances it with physically simulated detail.

6.1.1 Motivating scenarios

Consider two scenarios where tracking is important: (a) fleshing out a rough preview of a physical simulation and (b) adding physical detail to an animated character.

Coarse-to-fine design cycle To accelerate the simulation design process, artists use large time-steps and coarse geometry to generate rapid previews before committing resources to a full-detail physical simulation (see Fig. 6.1). When the technical director approves a promising preview, one might consider reusing the parameters of the preview in a full-resolution simulation; unfortunately, an ordinary simulator's output often does *not* resemble the preview. Our tracking solver, on the other hand, *guarantees* a similarity between input and output while adding physically simulated detail at fine scales.



Enriching animation with physics Our work takes one step toward *collocating* animated and physical behavior (see Fig. 6.2). The animation depicts a puppet-like character whose body consists of a thin, flexible material governed by the laws of physics. In this scenario, there is no distinct spatial or temporal boundary separating art from physics. This must be contrasted with common instances of *disjoint* couplings, e.g., skeleton-driven simulation, simulated fur, or cloth over an animated body (spatially disjoint); and animated

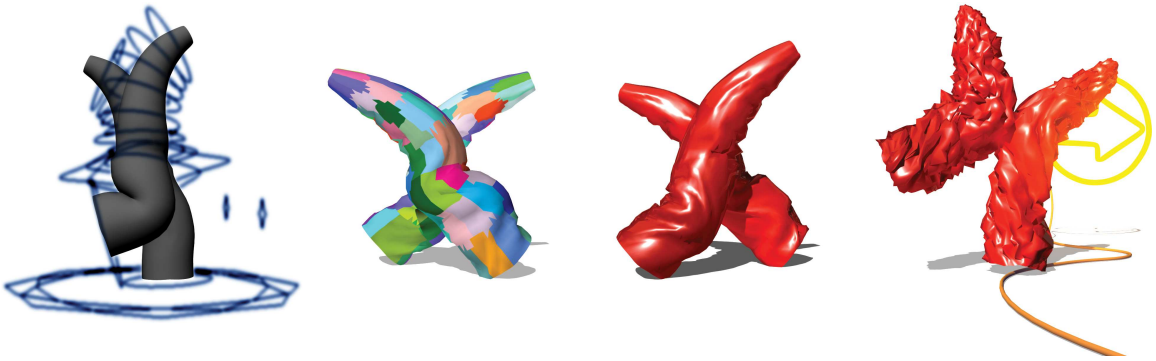


Figure 6.2: Tracking enables artistic expression and physical simulation to work hand-in-hand, as demonstrated in this animation of a character's unfortunate event. *Left to right*: We begin with the artist's animation, automatically generate a set of weak-form constraints (visualized as colored patches), and then solve the constrained equations of motion to flesh out wrinkles and folds.

keyframes interpolated by physics-based optimization (temporally disjoint). To the best of our knowledge, the spatial and temporal collocation of artistic animation and thin shell physics has not been an explicit goal of prior work in the simulation literature.

6.1.2 A tracking solution

Building on the foundation of constrained Lagrangian mechanics, we propose *weak-form constraints* for tracking the input motion. This method allows the artist to choose where to add details such as characteristic wrinkles and folds of various thin shell materials and dynamical effects of physical forces. We demonstrate multiple applications ranging from enhancing an artist's animated character to guiding a simulated inanimate object.

The scenarios we target in this chapter have two main characteristics. On the one hand, we focus on materials governed by the so-called *thin shell equations*: thin shells are flexible surfaces that evolve in time via stretching and bending modes. On the other hand, we assume a given rough input motion—the *guide trajectory*, which our tracking solver enhances with missing material-specific physical detail[1]. The material-dependent physical detail of thin

[1]Here we use the term *trajectory* as the time-evolving shape and position of the surface—as opposed to the path of its center of mass.

shells expresses itself in characteristic folds, wrinkles, and creases—time-evolving geometry that is exceedingly tedious to model manually. Because these details are straightforward to simulate with mathematical models, thin shell problems present an ideal playground to explore the idea of tracking. Since thin shells wrinkle and fold at multiple spatial and temporal scales, we must address the question of *which* physical scales to introduce over the given guide trajectory. In general, the answer to this question may vary over space and time. We provide the artist with several intuitive controls to describe the coarseness of the input trajectory and to set up the space of permissible physical details.

Method-in-brief We propose a simple framework for *track*ing *s*olvers, or *TRACKS*, which enhances a given guide trajectory with physically simulated detail [Bergou et al., 2007]. The framework consists of:

1. establishing a correspondence between *guide* (input) and *tracked* (output) shapes (see §6.4.1),
2. creating weak-form, *averaged constraints* (see §6.3.2 and §6.4.2), and
3. solving the *constrained Lagrangian mechanics* equations to add physical detail while enforcing the matching constraints (see §6.3.3 and §6.4.3).

The key feature that distinguishes our method from previous work lies in our use of the weak-form formulation to create constraints that force the guide and tracked trajectories to match in an overall or averaged sense. To facilitate the creation of the required correspondence, we also provide a novel extension to Lloyd's algorithm for clustering [1982] that automates the process. To the best of our knowledge, our solver is the first to flesh out a given surface animation with simulated dynamic detail, in a single pass, with artistic control over the scale of introduced details and guaranteed matching at coarser scales. However, our work builds on a large body of preceding contributions, which we briefly survey below.

6.2 Related work

Numerous works developed methods for guiding the course of a physical simulation. The directing of fluid simulations was considered by McNamara et al. [2004], Fattal and Lischinski [2004], Rasmussen et al. [2004], Shi and Yu [2005], Angelidis et al. [2006], and Thürey et al. [2006]. Other research focused on controlling the movement of elastic solids; see, e.g., [Smith et al., 2001; Capell et al., 2002, 2005; Kondo et al., 2005; Sifakis et al., 2005]. By contrast, work on directing plate and shell dynamics is comparatively scarce. Cutler et al. [2005] and Bridson et al. [2003] presented techniques for prescribing wrinkles on worn garments. Wojtan et al. [2006] applied the adjoint method to solving for the trajectory of a cloth particle system subject to keyframe constraints.

Control using multiple-pass and single-pass methods The spacetime constraints paradigm introduced by Witkin and Kass [1988] asks for a trajectory that satisfies user-provided keyframes while minimizing a given cost functional (e.g., the amount of work done by the physical system). Methods built on this paradigm can produce convincing motion based on very sparse data (a few constraints). However, the requisite solver—typically a form of multiple shooting [Witkin and Kass, 1988; Popović et al., 2003] or gradient-based optimization [McNamara et al., 2004]—can be computationally much more expensive than an ordinary simulation, requiring multiple iterations that each cost on the order of one simulation. To alleviate these problems, Cohen [1992] and later Treuille et al. [2003] applied windowing and refinement strategies, though multiple-pass methods such as these may still become intractable as the complexity of the physical system increases.

Seeking to avoid multiple iterations, various authors proposed single-pass approaches based on guide particles [Rasmussen et al., 2004] and prescribed force fields [Sifakis et al., 2005; Capell et al., 2005]. Our work fits the single-pass paradigm, since a tracking solver requires only a single pass and has runtime cost on the order of an ordinary simulation. In §6.3.3, we interpret our method in the framework of a force field approach by regarding the constrained Lagrangian as a means to automatically determine fictitious guide forces.

User input Popović et al. [2000, 2003] and Kondo et al. [2005] presented systems for rigid and deformable bodies in which a user sketches a trajectory and sets up key poses, and a physics solver produces a conforming animation. Twigg and James [2007] combined user interaction and quality metrics related to the physical system in guiding the output from multiple rigid body simulations run in parallel toward promising trajectories. Later, Twigg and James [2008] introduced a backward time integrator for final value problems, which used user-suggested paths when the motion was not uniquely determined by the backward integrator.

Enriching surface animations Kircher and Garland [2006] presented a multiresolution signal processing framework for editing surface animations, which included superimposing detail on a moving surface. Hadap et al. [1999] and Ma et al. [2006] added wrinkles to worn garments, and Loviscach [2006] presented a GPU-based method. These works adopted a geometric paradigm suited to a wide array of interactive tasks. Galoppo et al. [2006] introduced a physically based method for simulating high-resolution surface contact and collisions. Employing multiresolution in a physical simulation was previously considered by Grinspun et al. [2002], who adaptively refined basis functions to represent geometric detail. By contrast, our work uses test functions for defining spatially averaged constraints, effectively fixing the *space* of permissible details.

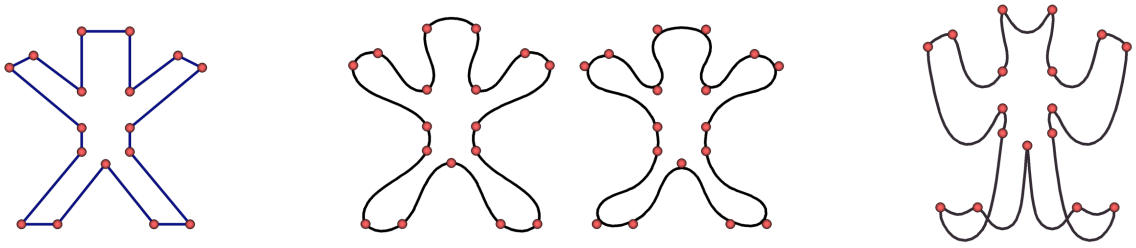
Reuse and data-driven methods Our solver enables an artist to reuse a coarse animation in achieving many different possible appearances for the output. Park and Hodgins [2006] acquired and then reproduced surface wrinkles. Cordier and Magnenat-Thalmann [2005] developed an interactive cloth simulator that reuses wrinkles learned from offline simulations. Both of these methods reuse geometric information and belong to a broader category of approaches that encode and reproduce surface geometry relative to a subject's pose (see [Singh and Kokkevis, 2000; Sumner and Popović, 2004] and references therein). Similarly, reuse of motion sequences was considered by, e.g., Zordan and Hodgins [2002] and Gleicher et al. [2003].

6.3 Control using weak constraints

In this section, we explain our central idea: matching the tracked output trajectory to the user-provided input trajectory by using weak constraints. We start with a simple 1D example in §6.3.1, showing the difference between interpolating (pointwise) and averaged (weak) constraints. In §6.3.2, we make precise the formulation of weak constraints and discuss how to enforce these constraints using the framework of constrained Lagrangian mechanics.

6.3.1 Motivation

We begin with the experiment shown in Fig. 6.3, in which we deform a rubber band (i.e., an elastic curve governed by bending and stretching forces) to "look like" the given guide shape. The *interpolating* approach forces the rubber band to pass through certain anchor points (shown in red) on the guide shape. On the other hand, the *averaging* approach asks for certain mean quantities (such as the centers of mass of corresponding segments between anchors) to be equal, which leads to an approximating curve not passing through the anchor points. As shown in Fig. 6.3, the interpolating approach can lead to undesirable tugging artifacts. The value of averaging in avoiding these artifacts is widely recognized in the geometric modeling community [Zorin and Schröder, 1998].



guide shape interpolated averaged interpolated

Figure 6.3: Starting with a guide shape *(left)*, interpolating and averaging approaches *(center-left and center-right, respectively)* both produce pleasing results in the absence of dynamics. Adding a sharp acceleration, however, results in tugging artifacts for the interpolating approach *(right)*, while the results obtained via averaging are unchanged.

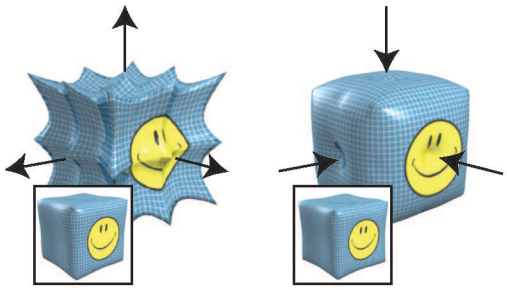


Figure 6.4: As the guide shape rapidly expands and contracts, pointwise constraints result in high-frequency tugging artifacts, whereas our averaged constraint method *(inset)* tracks smoothly.

In Fig. 6.4, we show a similar experiment for a 2D surface, with much the same results as in the 1D setting. This example uses more complicated averaged constraints than equating centers of mass. To better understand this, we formalize the notion of an averaged constraint.

6.3.2 Constraint formulation

Consider the guide and tracked configurations evolving over time in 3-space, given respectively by

$$\overline{\mathbf{q}} : \Omega \times \mathbb{R} \to \mathbb{R}^3 \quad \text{and} \quad \mathbf{q} : \Omega \times \mathbb{R} \to \mathbb{R}^3 \ .$$

Here Ω is the so-called *reference* or *material domain* of the elastic body, $\overline{\mathbf{q}}$ is the deformation mapping of the *guide* configuration, and $\mathbf{q}$ is the deformation mapping of the *tracked* configuration from the material domain into 3-space [Malvern, 1969]. We assume that the guide configuration, $\overline{\mathbf{q}}(\cdot, t)$, is given, whereas we view the tracked configuration, $\mathbf{q}(\cdot, t)$, as the temporally evolving configuration of a physical system. For the remainder of this section, we focus on a fixed instant in time and omit the time argument.

We subject the tracked physical system to *holonomic constraints* by restricting its motion to the space of permissible configurations, $\{\mathbf{q} \in \mathbf{Q} \,|\, \mathbf{c}(\mathbf{q}) = 0\}$, for some configuration space $\mathbf{Q}$ and objective function $\mathbf{c}$ [Lanczos, 1986]. In other words, we only consider solutions satisfying the constraint equation $\mathbf{c}(\mathbf{q}) = 0$ at all instants in time. In choosing $\mathbf{c}$ we encapsulate the requirement that the tracked and guide shapes match at certain coarse spatial scales.

We define a *set* of vector-valued constraints $\{\mathbf{c}^1, \mathbf{c}^2, \dots\}$ by

$$\mathbf{c}^i(\mathbf{q}) = \int_\Omega \overline{\mathbf{q}}(x)\phi_i(x)\mathrm{d}x - \int_\Omega \mathbf{q}(x)\phi_i(x)\mathrm{d}x \ . \tag{6.1}$$

This is a set of Petrov-Galerkin equalities, formed by choosing a specific set of *test functions* $\{\phi_1, \phi_2, \dots, \phi_K : \Omega \to \mathbb{R}\}$ [Strang and Fix, 1973]. For every ϕ_i, we constrain separately the three coordinate functions of the embeddings $\overline{\mathbf{q}}$ and $\mathbf{q}$. Simply stated, each equation requires a weighted average of positions on the guide surface to match a weighted average of positions on the tracked surface, using $\phi_i(x)$ as the weighting function.

We call attention to some of the properties of these averaged constraints: (i) they are *linear* in the unknown positions, $\mathbf{q}$; (ii) they depend explicitly on time since the guide shape moves over time; and (iii) the space of permissible configurations is closed under simultaneous rigid transformation of the tracked and guide shapes.

From a *Petrov-Galerkin* perspective, (6.1) requires a *weak* identification of tracked and guide coordinate functions under the finite-dimensional test space spanned by $\{\phi_i\}$. However, whereas classically the space of test functions is chosen based on differentiability requirements and then held fixed throughout the computation [Strang and Fix, 1973], we consider the support and profile of the test functions as variable in space and time. Indeed, the choice of test functions is a problem in *filter design*, if we consider that (6.1) requires the low-pass filtered version of the guide geometry to match the low-pass filtered version of the tracked geometry. The filter size and shape are linked to the support of the test functions: the proximity of the tracked shape to the guide shape increases with the locality of test function support (see Figs. 6.5 and 6.6).

Reference surface Thus far, we have assumed the existence of a common reference surface, Ω. In practice, explicit access to the material domain may not be possible, e.g., because the software is not finite-element based, or the user asks for tracked and guide geometry of differing genus. Therefore, we now relax this assumption.

Consider $\overline{\mathbf{q}}$ and $\mathbf{q}$ redefined over two *distinct* reference configurations, $\overline{\Omega}$ and Ω. In practice, we use the undeformed shape as the reference configuration. Recall that in (6.1) the test

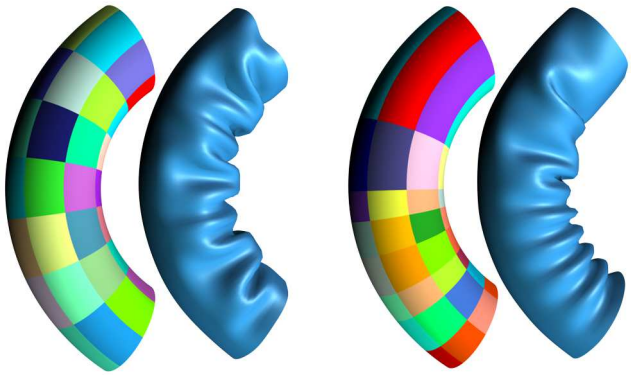


Figure 6.5: Buckling of the tracked surface is directly linked to the distribution of test functions over the mesh. A homogeneous distribution *(left)* results in uniform wrinkling of the surface, whereas a heterogeneous distribution *(right)* causes larger folds to form in coarser regions and smaller wrinkles to form in finer-support regions.

basis induced a set of weak equalities evaluated over a common domain. Having discarded the common domain, we now need a set of mutually *corresponding test functions*, $\{\overline{\phi}_i\}$ over $\overline{\Omega}$, and $\{\phi_i\}$ over Ω (for three ways to establish this correspondence, see §6.4.1). Then the resulting constraint equations are given by

$$\mathbf{c}^i(\mathbf{q}) = \int_{\overline{\Omega}} \overline{\mathbf{q}}(\overline{x})\overline{\phi}_i(\overline{x})\mathrm{d}\overline{x} - \int_{\Omega} \mathbf{q}(x)\phi_i(x)\mathrm{d}x \ , \tag{6.2}$$

where each test function must satisfy the normalization condition

$$\int_{\overline{\Omega}} \overline{\phi}_i(\overline{x})\mathrm{d}\overline{x} = \int_{\Omega} \phi_i(x)\mathrm{d}x \ . \tag{6.3}$$

This condition is always satisfiable by rescaling each ϕ_i. When $\overline{\Omega}$ and Ω have different area measures, $\mathrm{d}\overline{x}$ and $\mathrm{d}x$, then normalization ensures that the space of permissible configurations remains closed under simultaneous rigid transformation of the tracked and guide shapes. We now turn our attention to enforcing the constraints given by (6.2) and (6.3) in a physical simulation.

6.3.3 Enforcing constraints

TRACKS takes as input a user-provided guide to the object's motion and produces as output a physically detailed refinement of that motion. At each instant in time, the solver

evolves the tracked shape under its governing equations, which include physical forces as well as averaged constraints.

To enforce these constraints, we chose the method of *Lagrange multipliers*, which is particularly well suited for a constrained Lagrangian mechanics (CLM) approach. Since (6.2) defines *linear* constraints, we find it straightforward to enforce $\mathbf{c} = 0$ directly in the equations of motion using this framework, entirely avoiding problems associated with alternate methods of enforcements, briefly surveyed below, such as drifting from the constraint manifold or having to choose a predetermined stiffness to enforce the constraints. This framework involves the addition of auxiliary variables called Lagrange multipliers, $\boldsymbol{\lambda}$, to the equations of motion governing the system. Our treatment incorporates m vector-valued constraints, so that $\mathbf{c} \in \mathbb{R}^{3m}$ and $\boldsymbol{\lambda} \in \mathbb{R}^{3m}$.

While we choose the method of Lagrange multipliers to enforce the constraints within *TRACKS*, many other options are available. We give a brief overview of these various options below along with their relative strengths and weaknesses.

Penalty method Numerous approaches are available in the literature for maintaining constraints acting on a mechanical system [Shabana, 2001; Hairer et al., 2006]. Perhaps the most simple and straightforward of these is to use the penalty method, in which penalty forces are used to "pull" the system onto the constraint manifold. An auxiliary potential of the form $E_{\text{cons}} = \frac{k}{2} |\mathbf{c}|^2$ is added to the physical system, where k is a tunable stiffness parameter that needs to be large enough to maintain the constraint.

Requiring only the application of forces, this method is ideal for a black-box approach. Unfortunately, the stiffness parameter is often difficult to choose a priori, and the penalty forces required to closely enforce the constraints may be unacceptably stiff, requiring small time-steps to ensure stability [Goldenthal et al., 2007]. Likewise, critically damping a penalty force can be difficult. For these reasons, the penalty method may suffer from unnecessary numerical stiffness, instability, or unsatisfied constraints [Platt and Barr, 1988; Witkin and Baraff, 2001].

Constrained Lagrangian mechanics An alternative is to employ an augmented Lagrangian formulation. In constrained Lagrangian mechanics, a physical system must satisfy both the laws of motion and the constraint of staying on a certain geometric configuration [Platt and Barr, 1988]. This formulation can be viewed as a way of selecting the most physical trajectory of all possible trajectories that satisfy the constraint equations [Marsden and Ratiu, 1994]. Such formulations in general introduce additional variables (i.e., Lagrange multipliers) to enforce the constraints during the time integration step of the algorithm.

This framework leads to the constrained Euler-Lagrange equations [Lanczos, 1986]

$$\begin{aligned} \mathrm{M}\ddot{\mathbf{q}} - \mathbf{F}(\mathbf{q}) + \boldsymbol{\lambda}^T \frac{\partial \mathbf{c}}{\partial \mathbf{q}} &= 0 \ , \\ \mathbf{c}(\mathbf{q}, t) &= 0 \ . \end{aligned} \tag{6.4}$$

Here, M is the mass matrix, $\ddot{\mathbf{q}}(t)$ is acceleration, $\mathbf{F}$ represents the forces in the system, and $\boldsymbol{\lambda}$ is the Lagrange multiplier.

At each instant in time, (6.4) defines two sets of equations in the two sets of variables $\mathbf{q}(t)$ and $\boldsymbol{\lambda}(t)$. The first equation represents the physical evolution of the system according to Newton's law, incorporating fictitious constraint-maintaining forces $\boldsymbol{\lambda}^T \frac{\partial \mathbf{c}}{\partial \mathbf{q}}$, whereas the second equation is used to determine those forces such that the constraint is exactly maintained. The Lagrange multipliers themselves control the magnitude of the force, and the constraint gradient determines the direction of the force.

Different from this direct approach, the constraints, $\mathbf{c} = 0$, may be maintained differentially by requiring that $\ddot{\mathbf{c}} = 0$ (see, e.g., [Witkin and Welch, 1990]), effectively constraining the allowable accelerations. This approach gives rise to a linear system for determining the Lagrange multipliers, $\boldsymbol{\lambda}$, even if the constraints themselves are nonlinear. In practice, an additional constraint-correcting force is used to prohibit undesirable drifting of the solution away from the constraint manifold and to bring the initial state into an allowable configuration [Witkin and Baraff, 2001].

Manifold projection A third way of maintaining constrains is via manifold projection methods [Hairer et al., 2006; Goldenthal et al., 2007]. These methods perform constraint enforcement as a post-integration step, alternating between a time-integration step and a constraint-enforcement step. The first step results in an unconstrained position, $\widetilde{\mathbf{q}}^{n+1}$, for the system, while the second step seeks to find the closest point, $\mathbf{q}^{n+1}$, that is on the constraint manifold, $\mathbf{c}\left(\mathbf{q}^{n+1}\right) = 0$.

6.4 Building a tracking solver

In this section, we begin by describing how to design and implement the weak-form constraints when using meshes to represent the guide and tracked surfaces, then show a numerical integration scheme for the equations of motion that maintain those constraints, and conclude by pointing the reader to thin shell simulation models that can be used within this framework.

6.4.1 Designing test functions

Recall that in order to create the tracking constraints given by (6.2), we need to define a matching between the guide and tracked configurations by establishing *corresponding* test functions over both surfaces. We provide one manual and two automatic methods to design such a correspondence for meshes.

On-mesh painting In the most hands-on approach, the user paints pairs of corresponding regions on the undeformed (reference) guide and tracked meshes. To each painted region on the guide mesh, we associate a test function $\overline{\phi}_i$ defined to have value 1 for all vertices inside that region and value 0 otherwise. To determine the constant value of ϕ_i within the corresponding region on the tracked mesh, we use the normalization condition given by (6.3).

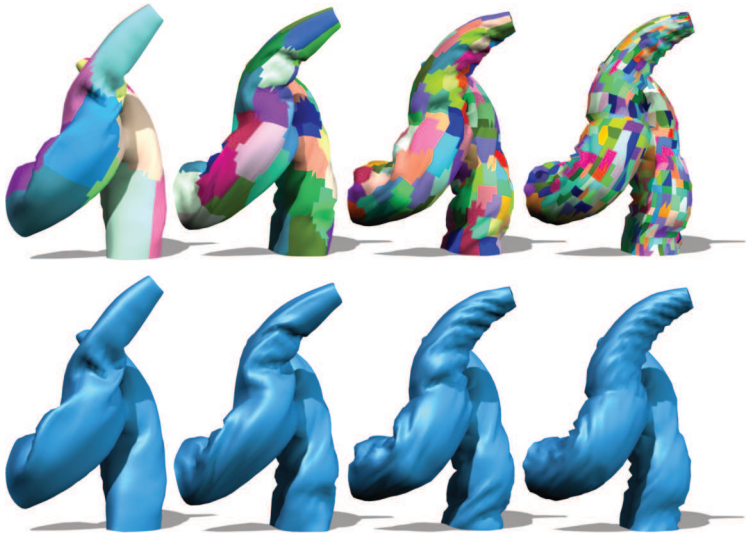


Figure 6.6: Increasing the number of partitions (*left to right*: 16, 64, 256, 1024) narrows the support of the test functions, forcing closer tracking and finer wrinkles.

Mesh subdivision For some examples, (see Figs. 6.4, 6.8, and 6.9) we create the undeformed tracked mesh by linearly subdividing the undeformed guide mesh; such subdivision automatically induces a hierarchical basis [Zorin and Schröder, 1998]. We use the coarsest level of the resulting hierarchical basis as the test functions on both the guide and tracked meshes, so $\phi_i = \overline{\phi}_i$. These test functions correspond to the Lagrange basis functions on the guide mesh: each function $\overline{\phi}_i$ is linear and defined to be 1 at vertex i on the guide mesh and 0 at all others.

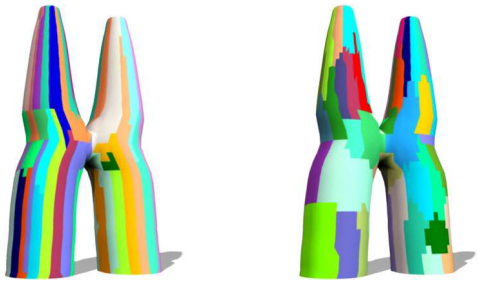


Figure 6.7: Results shown using original *(left)* and animation-aware *(right)* VSA clustering algorithm.

Variational clustering We propose an animation-aware extension to Lloyd's algorithm based on Variational Shape Approximation (VSA) [Cohen-Steiner et al., 2004], which we use to partition the guide mesh into r disjoint regions (see Fig. 6.6) and assign a test function to each region. To initialize the algorithm, we randomly choose r triangles as *proxies* $\{P_1, \ldots, P_r\}$. The algorithm then proceeds in two steps: (i) growing regions and (ii) updating the proxies.

Step i: We grow regions around the proxies using the algorithm described in §3.3 of Cohen-Steiner et al. [2004], which assigns each mesh triangle to the *closest* proxy while ensuring connectedness of regions. To compute the distortion distance between a guide mesh triangle, T_j, and a proxy, P_k, we use the error metric

$$E(T_j, P_k) = \sup_{t \in [t_0, t_1]} A_j(t) \left(\frac{\mu}{A(t)} \|\mathbf{x}_j(t) - \mathbf{x}_k(t)\|^2 + (1-\mu)\|\mathbf{n}_j(t) - \mathbf{n}_k(t)\|^2 \right) ,$$

where $\mathbf{x}_j(t)$ and $\mathbf{x}_k(t)$ are centroids of T_j and P_k respectively, $\mathbf{n}_j(t)$ and $\mathbf{n}_k(t)$ are unit normals associated to T_j and P_k respectively, $A_j(t)$ is the area of T_j, $A(t)$ is the total surface area of the guide mesh, and all the aforementioned variables are functions of time. The user-prescribed coefficient $\mu \in [0, 1]$ controls the bias of the regions toward compactness ($\mu \to 1$) or flatness ($\mu \to 0$).

Our error metric differs from that proposed in VSA due to two considerations. First, whereas VSA's focus on *remeshing* prefers large, flat regions, our focus on *tracking* prefers evenly sized, flat regions; consequently, we use the Euclidean metric in place of VSA's distance-to-plane metric. Second, whereas VSA partitions a static mesh, we seek a fixed partition that is reasonable at any instant in time, $t \in [t_0, t_1]$, for the given animation of the guide mesh; therefore, we include a supremum over time. In practice, we approximate the supremum by considering the maximum over a fixed set of randomly chosen frames. Notice that our animation-aware algorithm tends to align seams between regions to boundaries between rigidly deforming patches, e.g., to the character's joints (see Fig. 6.7).

Step ii: Next, we recompute the proxy, P_k, for each region as an area-weighted average of its constituent triangles' position and normal; since geometry is a function of time, so are the proxies.

These two steps are repeated until convergence, resulting in a clustering of the guide mesh into regions (see Fig. 6.7). We use this partitioning algorithm only when the undeformed tracked and guide meshes are identical, so the regions on the tracked mesh are the same as those on the guide. Once these regions are created, we again assign a test function ϕ_i to each region on the mesh such that $\phi_i = 1$ for vertices within the region and $\phi_i = 0$ otherwise (since the two meshes are identical, we have $\overline{\phi}_i = \phi_i$).

For the animation shown in Fig. 6.2, consisting of a 3038 vertex mesh, our animation-aware partitioning required 32 iterations (less than five minutes) to generate a 200 region partition. The choice of cluster-count effectively controls the cut-off point between the scale of the details added through simulation and those kept intact from the input guide.

6.4.2 Implementing constraints

In our implementation, we express the test functions on the guide mesh as linear combinations of the Lagrange basis functions, $\{\overline{\psi}_j\}$:

$$\overline{\phi}_i(\overline{x}) = \sum_{j=1}^{\overline{n}} \overline{S}_{ij}\overline{\psi}_j(\overline{x}) \ , \tag{6.5}$$

where $\overline{n}$ represents the number of vertices in the guide mesh. To obtain the entries of the matrix $\overline{S}$, we use one of the methods from §6.4.1 to evaluate $\overline{\phi}_i$ at mesh vertices and set $\overline{S}_{ij} = \overline{\phi}_i(\overline{x}_j)$, where $\overline{x}_j$ is the material coordinate of vertex j of the guide mesh. We write the test functions, $\{\phi_i\}$, on the tracked mesh in an analogous manner to (6.5).

We can now express the $3m$ constraints from (6.2) and the associated gradient at time-step k as

$$\mathbf{c}_k(\mathbf{q}_k) = \underbrace{\overline{S\mathcal{M}}}_{\overline{C}}\overline{\mathbf{q}}_k - \underbrace{S\mathcal{M}}_{C}\mathbf{q}_k \quad \text{and} \tag{6.6}$$

$$\nabla\mathbf{c}_k(\mathbf{q}_k) = -C \ , \tag{6.7}$$

where we use the abbreviation $\mathbf{c}_k(\mathbf{q}_k) = \mathbf{c}(\mathbf{q}_k, t_k)$, and the entries of the Petrov-Galerkin mass matrix, $\mathcal{M}$, are given by

$$\mathcal{M}_{jj} = \frac{A_j}{6} \qquad \text{and} \qquad \mathcal{M}_{jl} = \frac{A_{jl}}{12} \ .$$

Here, A_j denotes the total area of the triangles incident to vertex j, and A_{jl} denotes the total area of the triangles incident to edge jl (and analogously for $\overline{\mathcal{M}}$).

The normalization condition from (6.3) can be satisfied by scaling the entries in each row of C at the beginning of the simulation:

$$C_i \leftarrow C_i \frac{\sum_{j=1}^{\overline{n}} \overline{C}_{ij}}{\sum_{l=1}^{n} C_{il}} \ . \tag{6.8}$$

Since the matrices C and $\overline{C}$ depend only on the undeformed meshes and the test functions, they can be precomputed for efficiency.

6.4.3 Numerical integration

We describe our implementation of the constrained implicit Euler method (for alternatives consult [Hauth et al., 2003; Boxerman and Ascher, 2004; Hairer et al., 2006]):

$$\begin{aligned} M\frac{\dot{\mathbf{q}}_{k+1} - \dot{\mathbf{q}}_k}{h} - \mathbf{F}(\mathbf{q}_{k+1}) + \boldsymbol{\lambda}_k^T \nabla \mathbf{c}_k(\mathbf{q}_k) &= 0 \ , \\ \frac{\mathbf{q}_{k+1} - \mathbf{q}_k}{h} - \dot{\mathbf{q}}_{k+1} &= 0 \ , \\ \mathbf{c}_{k+1}(\mathbf{q}_{k+1}) &= 0 \ . \end{aligned}$$

Here $(\mathbf{q}_1, \mathbf{q}_2, \dots)$ denotes the time-discrete trajectory of the fine mesh, $(\dot{\mathbf{q}}_1, \dot{\mathbf{q}}_2, \dots)$ denotes the velocities along the trajectory, M is the physical mass matrix, and $\mathbf{F}$ represents the forces present in the system (see §6.4.4). The above system can be rearranged as an implicit equation in the unknown variables $(\dot{\mathbf{q}}_{k+1}, \boldsymbol{\lambda}_k)$, with the update rule for $\mathbf{q}_{k+1}$ given in the second line. We compute a single iteration of Newton's method per time-step, as in Baraff and Witkin [1998]. Taking $(\dot{\mathbf{q}}_k, 0)$ as the initial guess for $(\dot{\mathbf{q}}_{k+1}, \boldsymbol{\lambda}_k)$ yields:

$$\begin{bmatrix} M - h^2 J_F(\mathbf{q}_k + h\dot{\mathbf{q}}_k) & h\left[\nabla g_k(\mathbf{q}_k)\right]^T \\ h\nabla g_k(\mathbf{q}_k) & 0 \end{bmatrix} \begin{bmatrix} \Delta\dot{\mathbf{q}} \\ \boldsymbol{\lambda}_k \end{bmatrix} = \begin{bmatrix} h\mathbf{F}(\mathbf{q}_k + h\dot{\mathbf{q}}_k) \\ -g_{k+1}(\mathbf{q}_k + h\dot{\mathbf{q}}_k) \end{bmatrix} ,$$

where $J_F(\mathbf{q}_k + h\dot{\mathbf{q}}_k)$ is the Jacobian of forces evaluated at $\mathbf{q}_k + h\dot{\mathbf{q}}_k$, and the constraint and constraint gradient are evaluated using (6.6) and (6.7), respectively. We solve this system for $(\Delta\dot{\mathbf{q}}, \boldsymbol{\lambda}_k)$ using the PARDISO [Schenk and Gärtner, 2006] sparse linear solver, then set

$$\dot{\mathbf{q}}_{k+1} = \dot{\mathbf{q}}_k + \Delta\dot{\mathbf{q}} \qquad \text{and} \qquad \mathbf{q}_{k+1} = \mathbf{q}_k + h\dot{\mathbf{q}}_{k+1} \ .$$

6.4.4 Implementing thin shell simulations

Our approach is not tied to a particular way of discretizing shells or cloth and easily incorporates into an existing solver with its constituent implementation of bending and stretching forces. We chose to work with triangle meshes and employed the model of *Discrete Shells* [Grinspun et al., 2003] for bending potential energy and *Constant Strain Triangle* [Zienkiewicz and Taylor, 2000] for in-plane stretching potential energy. As these models (and several variants surveyed by Ng and Grimsdale [1996]; Thomaszewski and Wacker [2006]) are by now well known and widely employed by the graphics community, we refer the reader to the above references for details.

The true power of tracking is most evident when the underlying (non-tracking) simulator is able to capture the look and feel of many different materials. In this light, we briefly describe a simple plasticity model that broadened our simulator's expressive power.

Simple model for plasticity Plasticity has been widely studied within computer graphics (see, e.g., [Terzopoulos and Fleischer, 1988; O'Brien et al., 2002; Irving et al., 2004; Gingold et al., 2004; Wicke et al., 2005] and references therein; for a broader overview, see [Zienkiewicz and Taylor, 2000]). We briefly outline a model for plasticity developed primarily with consideration for (a) simplicity of implementation and (b) proper scaling (material response should not depend on mesh- or time-resolution).

We start with a traditional bending model [Baraff and Witkin, 1998; Bridson et al., 2003; Grinspun et al., 2003] and add two additional parameters: (i) maximal curvature deviation, $\Delta\kappa_{max} \in [0, \infty]$, which dictates the maximal strain for which response is purely elastic and (ii) hardening factor, $\eta \in \mathbb{R}$, which governs the hardening ($\eta > 0$) or softening ($\eta < 0$) of creases due to plastic work. For each mesh edge, we store the undeformed configuration's bend angle, $\overline{\theta}_i$, as well as a per-edge elastic bending stiffness, k_i. At the start of each timestep, we test for a *plastic update* on each edge (the subscript i is implied). We compute the *curvature deviation*,

$$\Delta\kappa = \kappa - \overline{\kappa} = \frac{\overline{e}}{2\overline{A}}(\theta - \overline{\theta}) \ ,$$

where $\overline{A}$ is the combined area of the undeformed triangles incident to edge $\overline{e}$, and θ (resp. $\overline{\theta}$) is the angle formed between incident deformed (resp. undeformed) triangle normals. Unlike the change in bend angle $(\theta - \overline{\theta})$, the curvature deviation accounts for spatial scaling (mesh sampling density) [Grinspun and Wardetzky, 2008]. If $|\Delta\kappa| > \Delta\kappa_{max}$, we perform a *plastic strain update*[2] with *plastic hardening*:

$$\overline{\theta} \leftarrow \overline{\theta} + \text{sign}(\Delta\kappa)\frac{2\overline{A}}{\overline{e}}\left(|\Delta\kappa| - \Delta\kappa_{max}\right) \ ,$$

$$k \leftarrow k \exp\left(\eta\left(|\Delta\kappa| - \Delta\kappa_{max}\right)\right) \ .$$

We note that the exponential provides proper temporal scaling.

External forces and collisions While we focus mainly on material properties, our method accommodates wind and other external forces. As it is single-pass and involves only forward-dynamics, any existing collision-response code should be compatible with our approach; by contrast, spacetime optimization has difficulties under complex collisions [Wojtan et al., 2006]. Our experiments use the robust treatment of collisions and self-collisions described in Bridson et al. [2002].

6.5 Applications

In the following application examples, we demonstrate *TRACKS* by directing discrete shell simulations. The computation times for the tracking solver are very practical, consuming for the most complex examples comfortably below four hours on a 2GHz processor. Indeed, we find that, consistent to our goal, the bulk of our time and effort is allocated to the artistic process. We consider scenarios spanning four axes:

1. To **create the input trajectory**, we use a rapid-preview physical simulation in §6.5.1, procedurally generate motion in §6.5.2, build on a motion-capture (mocap) sequence in §6.5.3, and ask an artist to animate expressive character gestures in §6.5.4.

[2]Note that the original publication [Bergou et al., 2007] has a sign error in formula for the plastic strain update.

2. In regards to **input mesh resolutions**, we test a coarsest possible input in §6.5.2, a more refined input in §6.5.1 and §6.5.3, and a fine input in §6.5.4.

3. In setting up **input→output combinatorics**: the output corresponds to a combinatorially subdivided input in §6.5.1, §6.5.2, and §6.5.3, whereas input and output are combinatorially identical in §6.5.4.

4. We consider three techniques for **creating test functions**: in §6.5.2–§6.5.3 we use the Lagrange basis of the guide mesh, prolongated onto the fine tracked mesh via linear subdivision; in §6.5.4, we use a novel extension to Lloyd's algorithm to automatically construct a piecewise constant basis; but first, in §6.5.1, we put aside the automatic methods, and manually paint a pair of test functions.

6.5.1 Simulation design with fast, coarse previews

Fleshing out a coarsely run simulation is a prototypical use of our solver. As shown in the accompanying video and Fig. 6.1, the director intends for the flying cloth to get stuck on a branch. We search for initial conditions and material coefficients using a *rapid preview* design cycle: working with a coarse mesh (35 vertices), after twenty design iterations, or about 100min working on a 2GHz notebook, we establish our desired parameters (see Fig. 6.1-*left*). If instead, the full-resolution mesh (1625 vertices) were used during design, the consequent simulation cost (30+ hours) would require a batch process, instead of a rapid and thus more intuitive design cycle.

Unfortunately, the preview's parameters do not translate well to the high-resolution simulation as shown in Fig. 6.1-*middle*. Indeed, despite great care in designing a system of meshing-independent simulation parameters (i.e., a system that incorporates well-reasoned scaling factors), the bifurcations and complex configuration landscape of quasi-inextensible surfaces that buckle and fold make it very unlikely that a coarse and fine simulation will agree.

Using our tracking method ensures that the final, expensive, high-resolution computation produces the expected results on the first run. We use four hand-painted test functions on

the flag because our goal is to guarantee a global trajectory for the cloth, not to control its fine shape. The fine mesh is a combinatorially subdivided version of the input one.

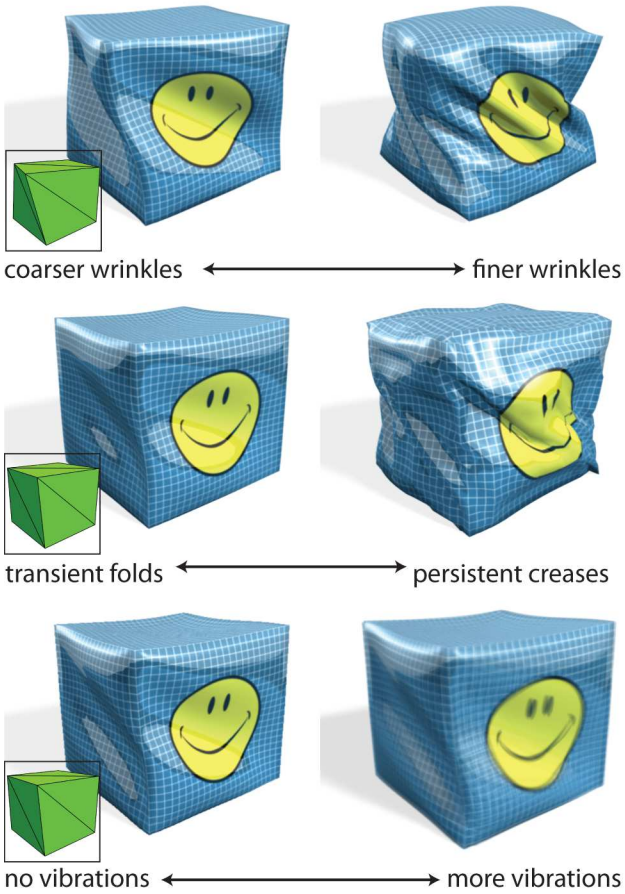


Figure 6.8: Traditionally, after an animation is completed, the lighting and rendering crew adjust and compare various rendering alternatives. With *TRACKS*, an artist enjoys the *post facto* freedom to also adjust material and dynamical properties: *(top)* stretching and bending stiffness control the shape of wrinkles during a twist; *(middle)* plasticity controls whether the material rebounds or the wrinkles persist; *(bottom)* damping controls oscillations such as vibrations and wobbliness.

6.5.2 The simple box

A more challenging scenario is one in which the coarse motion is not physically based. Our simplest such experiment begins with a coarse box modeled with eight vertices (see Fig. 6.8), which is animated with procedurally generated rigid motion of the top face while the bottom

face is fixed to the ground. We produce the corresponding fine mesh by linearly subdividing four times (triangle quadrisection and a linear geometric stencil), and we choose as our (piecewise linear) test functions the coarsest subdivision basis.

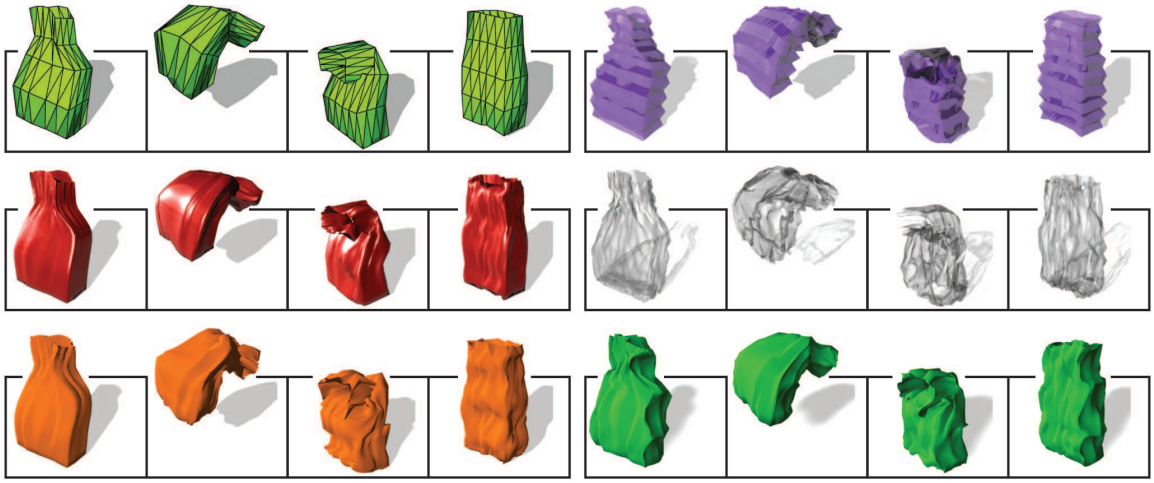


Figure 6.9: We naïvely rig a coarse mesh and use off-the-shelf motion-capture data to obtain the animation shown *(top-left)*, which, while riddled with self-collisions, serves as input to the automatic tracking process. The following sequences depict snapshots of a collision-free 1297-vertex mesh tracking the coarse motion under a variety of dynamical and material parameters.

The viewer witnesses an inanimate object come to life, as the fine box simultaneously achieves two objectives: (i) it clearly tracks the non-physical coarse motion while (ii) exhibiting dynamical material properties. By *dynamical*, we mean that similar effects could *not* be achieved by post-processing each frame of the coarse animation in isolation. For example, at the end of the sequence, the coarse mesh suddenly stops moving, yet the tracked mesh continues to move under inertia.

6.5.3 Motion-captured backflip

By combining multiple constitutive laws and dynamical controls, *TRACKS* enables an artist to apply a broad range of material behaviors to a predefined coarse motion. Since *TRACKS* is a framework incorporated over existing simulation code, it inherits the expressive power and material realism existent in the underlying simulation software.

In this example, we begin with mocap data of a breakdancer's backflip. As depicted in Fig. 6.9 and the accompanying video, we apply the mocap sequence to a coarse (88 vertices) triangle mesh using an industry-standard rigging tool [Autodesk]. To create the initial fine mesh (1297 vertices), we twice subdivide the initial coarse mesh. Once again, we use the subdivision basis functions of the coarse mesh as the test functions for the weak constraints.

Holding the input trajectory fixed, we run the tracking algorithm using various dynamical and material parameters, obtaining a diverse range of results (refer to video and Fig. 6.9). The coarse mocap-induced motion is riddled with surface self-intersections, which are fully resolved during tracking.

Elasticity The first animation depicts a red vinyl material. Notice the fine vertically oriented wrinkles and folds formed as the top of the bag contracts in the first snapshot, the crinkly top and boxy backside corners in the third snapshot, and the vertically rippled appearance due to slight lateral contraction in the final snapshot. We achieve this look using a highly damped elastic shell model. Observe that damping affects only the material, *not* the coarse motion.

Damping The second animation depicts a flowy, undamped material reminiscent of an orange silk satchel. Because we use identical elastic parameters, in the first snapshot the red vinyl and orange flowy materials appear to be similar. However, since the second material is relatively undamped, its appearance diverges from the first as time advances. The impact of the backflip's hard landing (third snapshot) causes flowy, larger-wavelength horizontal waves (subtly, the smaller-wavelength vertical ripples due to the bag's lateral compression remain discernible).

Undeformed configurations The next pair of animations use our ability to track the coarse mesh using differing undeformed configurations. Applying coarse horizontal folds effects an accordion-like geometry on a purple bag; likewise, we obtain the appearance typical of a woven red-ribbon gift bag with a set of fine vertical ridges (refer to video). Altering

the undeformed shape affects the objects' silhouettes and influences the formation and positioning of additional wrinkles, buckles, and folds (compare snapshot four to materials with a flat rest state)—these effects exceed what is achievable purely during rendering.

Complex collisions Building on this, we perturb the undeformed configuration with noisy offsets in the normal direction, achieving the appearance of a crinkled cellophane bag. The rough surface features did not noticeably alter runtime: tracking is affected by collisions much like an ordinary simulation; by contrast, many optimization methods rely on smoothness properties that are annihilated by rough collision landscapes.

External forces The final example incorporates a strong external wind force. This force excites the material without disturbing the gross motion, creating a psychedelic effect.

Unlike the coarse input, the tracked output is free of self-intersections. While one could conceivably attempt these material effects by subdividing the coarse motion and procedurally superimposing crinkly geometry or flowy motion, a simulation-based approach excels at producing temporally evolving wrinkles and folds while steering clear of collisions present in the input data.

6.5.4 Physically detailed characters

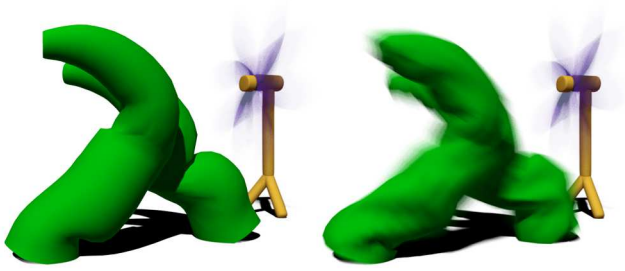


Figure 6.10: We underscore the animator's broad gestures *(left)* with a simulated headwind *(right)*. For equal comparison, both are rendered using 16-frame motion blur.

Our final example explores the role of *TRACKS* as a catalyst for synergy between art and

	# verts (guide)	# verts (tracked)	# constraints	increase in DOFs	time per frame (s)	increase in time
§6.5.1	35	1625	4	0.2%	17.7 (17.0)	4.1%
§6.5.2	8	1538	8	0.5%	15.8 (14.9)	6.0%
§6.5.3	88	1297	88	6.8%	16.1 (15.7)	2.5%
§6.5.4	3038	3038	280	9.2%	43.2 (39.0)	10.8%

Table 6.1: We report the runtimes on a 2GHz CPU for the various simulations throughout this chapter using our tracking solver. For comparison, we have also listed in parentheses the runtimes obtained by simply running a standard simulation using the same parameters but with no tracking. We see a typical increase in simulation time of 5%–10% for the problems considered.

physics. The artist models and animates Xavier, an X-shaped biped (see Figs. 6.2,6.6,6.10) made of thin, flexible material. In doing so, she disregards physical details, such as wrinkling and buckling, and focuses on expressive motion, such as shifting body weight, sudden movement, and keeping time to the music. The tracking process enables us to outfit Xavier's motion with different materials and physical effects. As Xavier walks past his friends, observe the perfect roll-through of his feet—painting fine test functions on the feet enables us to precisely echo the artist's intent. Next, Xavier encounters a strong headwind (see Fig. 6.10). We underscore Xavier's struggle by introducing wind forces that make his body flutter rapidly. This would have been painstaking to achieve by hand. In the next scene, Xavier steps on a live wire, and by altering the undeformed configuration between a crinkly and smooth state, we (regrettably) simulate the consequent electrocution (see Fig. 6.2). Fortunately, Xavier "shakes off" the remaining charge and returns to his unruffled state.

6.5.5 Timing comparisons

The exact runtimes we report for the examples in this chapter are very specific to our implementation; however, our experience for all the example applications is that solving the constrained system incurs typically a 5%–10% overhead over an unconstrained simulation. Unlike collision detection for example, tracking is not done as a separate pass but as part

of the simulation itself, so optimizing the simulation code would not change the percentage increase in runtime due to tracking.

6.6 Discussion

We presented *TRACKS*, a framework which enriches given input motion of thin flexible surfaces with physically simulated detail. Our approach may be viewed as a method inspired by the "line of action" metaphor [Thomas and Johnston, 1981]—adding detail to a preview in a coarse-to-fine workflow. Our solver, in a single pass, provides such detail by integrating the equations of motion subject to constraints dictated by the input guide trajectory; we perform this integration using the framework of constrained Lagrangian mechanics. The scale at which our solver adds detail is controlled by the size and shape of Petrov-Galerkin test functions. We have shown the influence of the support and spatial distribution of test functions in controlling the characteristic shape of wrinkles and folds of thin shell materials. One particular avenue we plan to pursue in the future is to explore the effect of temporally varying test functions for input meshes that exhibit subtly changing detail over time.

In our current formulation of tracking, constraint forces always overpower the material's internal forces, without discrimination as to whether they arise from stretching versus bending response. Consequently, if the artist animates the mesh in a stretching mode, the underlying material will also stretch. While this is often desirable, there are cases where it is preferred that the material remain inextensible, even if the consequence is violation of the averaged constraints. Therefore, we are interested in extending the tracking formulation with a form of constraint regulation that effectively bounds constraint forces so that, when desired, they affect bending but not stretching modes. Furthermore, in the future, we would like to explore methods for constraining the *silhouettes* of animated characters, since art-directing the silhouettes is commonplace in traditional animation [Thomas and Johnston, 1981].

Tracking offers the opportunity to experiment with a diverse range of thin shell appearances that defy emulation via geometric subdivision or per-frame post-processing. While our examples focused on thin plates and shells, the theory behind *TRACKS* requires only the

existence of a computationally tractable Lagrangian formulation, e.g., such as those known for fluids and solid elastica. Therefore, we expect that this general framework also find applications in animation of smoke, liquids, biological tissue, and solid elastica.

Chapter 7

Conclusion

7.1 Discussion

Our main contributions are: (i) a model for the bending energy of cloth based on expressing the mean curvature of the surface as a *linear* function of the surface's position surface, offering good agreement with and superior performance over the standard nonlinear models (§2); (ii) a unified, reduced coordinate model for simulating elastic rods and viscous threads (§4 and §5); and (iii) a framework for directable simulation that *guarantees* that the high resolution, physically simulated output matches a given input, which can be simulated or hand-animated (§6).

In all of our work, we approach the problem from a mathematical point of view—in particular, we focus on the *geometry* describing the physical system and its dynamics. In our work on directable simulation, we view the requirement of "matching" the input animation as restricting the motion of the physical system to *permissible* configurations defined by a set of matching constraints—that is, requiring the motion of the system to lie on some constraint manifold. This leads naturally to considering the framework of constrained Lagrangian mechanics, and the goal of the work becomes defining these matching constraints. We provide one possibility using the notion of weak-form, Petrov-Galerkin constraints that allows the user to easily set the scale at which to introduce physically simulated detail.

The core of our work on the bending energy of cloth and the simulation of elastic rods and viscous threads is built on the ideas from Discrete Differential Geometry (DDG). The main objective of this is to identify the "right" discrete models for a continuous system; i.e., those that preserve the geometric structures from the continuous system. While many discrete models converge to the smooth model, the main challenge is to build the discrete model in such a way that these structures are preserved *exactly*, not just in the limit of refinement.

While building these types of discrete geometric models for simulation of dynamics can be important from a theoretical point of view, our aim has been to show that building the models in this way also has a *practical* benefit. In addition to the principled, mathematical foundation that we strive to build our work on, one of our goals is to ensure the *efficiency* of the resulting methods. In the case of the bending energy of cloth, the computational efficiency resulting from the linear forces and constant energy Hessian provide one such quantifiable benefit to preserving the structure afforded by considering isometric deformations of the surface. Likewise, exposing the role that parallel transport plays in the geometric quantities associated with and coordinates describing an adapted framed curves allows us to make computationally favorable choices without sacrificing on the elegance and rigor of the theory.

7.2 Future work

We applied the ideas presented in our work on directable simulation to the case of simulating thin shells. However, the underlying framework of constrained Lagrangian mechanics applies to a much broader range of physical systems, so the ideas could potentially be applied to many different systems. For example, fluids also exhibit a variety of behaviors at different scales, so the idea of using weak-form constraints to guide the overall motion while allowing for small-scale details to be added could find application in this area, as well.

In our work on elastic rods and viscous threads, we have exposed the role of space-time symmetry in choosing an efficient representation for the purpose of simulation. While this symmetry was key at arriving at the representation, our final algorithm takes the standard

route of expressing the equations of motion as a system of ODEs that are discrete in space but continuous in time and then applying a standard time-stepping scheme to this system of ODEs. However, it would be interesting to develop the discrete theory such that space and time are both on the same footing, arriving at a discrete system that has the same symmetry as the continuous one.

In addition to this more theoretical work, it would be of interest from an engineering perspective to validate the models presented using different constitutive laws, involving different relations between the stresses and strains. While implementation-wise, the requisite changes would not be drastic, the benefit would be the ability to use the models presented for a wider variety of materials and applications.

Bibliography

E. Andreassen, E. Gundersen, E. L. Hinrichsen, and H. P. Langtangen. *Numerical Methods and Software Tools in Industrial Mathematics*, chapter 9: A Mathematical Model for the Melt Spinning of Polymer Fibers, pages 195–212. Birkhäuser, Boston, 1997.

A. Angelidis, F. Neyret, K. Singh, and D. Nowrouzezahrai. A Controllable, Fast and Stable Basis for Vortex Based Smoke Simulation. In *2006 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 25–32, sep 2006.

S. S. Antman. *Nonlinear Problems of Elasticity*, volume 107 of *Applied Mathematical Sciences*. Springer, 2nd edition, 2005.

U. M. Ascher and E. Boxerman. On the modified conjugate gradient method in cloth simulation. *The Visual Computer*, 19(7-8):526–531, 2003.

B. Audoly and Y. Pomeau. *Elasticity and Geometry: From Hair Curls to the Non-linear Response of Shells*. Oxford University Press, 2010.

B. Audoly, N. Clauvelin, and S. Neukirch. Elastic Knots. *Phys. Rev. Lett.*, 99(16):164301, 2007.

Autodesk. Autodesk Maya Unlimited 8.0.

S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc Web page, 2009. http://www.mcs.anl.gov/petsc.

D. Baraff and A. P. Witkin. Large Steps in Cloth Simulation. In *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, pages 43–54, jul 1998.

D. Baraff, A. Witkin, and M. Kass. Untangling Cloth. *ACM Transactions on Graphics*, 22 (3):862–870, jul 2003.

C. Batty and R. Bridson. Accurate Viscous Free Surfaces for Buckling, Coiling, and Rotating Liquids. In *2008 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 219–226, jul 2008.

M. Bergou, M. Wardetzky, D. Harmon, D. Zorin, and E. Grinspun. A Quadratic Bending Model for Inextensible Surfaces. In *Fourth Eurographics Symposium on Geometry Processing*, pages 227–230, jun 2006.

M. Bergou, S. Mathur, M. Wardetzky, and E. Grinspun. TRACKS: Toward Directable Thin Shells. *ACM Transactions on Graphics*, 26(3):50:1–50:10, jul 2007.

M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, and E. Grinspun. Discrete Elastic Rods. *ACM Transactions on Graphics*, 27(3):63:1–63:12, aug 2008.

M. Bergou, B. Audoly, E. Vouga, M. Wardetzky, and E. Grinspun. Discrete Viscous Threads. *ACM Transactions on Graphics*, 29(4), Jul 2010.

F. Bertails, B. Audoly, M.-P. Cani, B. Querleux, F. Leroy, and J.-L. Lévêque. Super-Helices for Predicting the Dynamics of Natural Hair. *ACM Transactions on Graphics*, 25 (3):1180–1187, jul 2006.

R. L. Bishop. There is More than One Way to Frame a Curve. *The American Mathematical Monthly*, 82(3):246–251, 1975.

W. Blascke. *Vorlesungen über Differentialgeometrie III*. Springer, 1929.

A. I. Bobenko and P. Schröder. Discrete Willmore Flow. In *Third Eurographics Symposium on Geometry Processing*, pages 101–110, jul 2005.

A. I. Bobenko and Y. B. Suris. Discrete Time Lagrangian Mechanics on Lie Groups, with an Application to the Lagrange Top. *Communications in Mathematical Physics*, 204(1): 147–188, 1999.

A. I. Bobenko, P. Schröder, J. M. Sullivan, and G. M. Ziegler, editors. *Discrete Differential Geometry*, volume 38 of *Oberwolfach Seminars*. Birkhäuser Verlag AG, Basel, Switzerland, 2008. ISBN 978-3-7643-8620-7.

A. Bonito, M. Picasso, and M. Laso. Numerical simulation of 3D viscoelastic flows with free surfaces. *Journal of Computational Physics*, 215(2):691–716, jul 2006.

E. Boxerman and U. Ascher. Decomposing cloth. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 153–161, jul 2004.

F. Boyer and D. Primault. Finite element of slender beams in finite transformations: a geometrically exact approach. *International Journal for Numerical Methods in Engineering*, 59(5):669–702, 2004.

D. E. Breen, D. H. House, and M. J. Wozny. Predicting the Drape of Woven Cloth Using Interacting Particles. In *Proceedings of SIGGRAPH 94*, Computer Graphics Proceedings, Annual Conference Series, pages 365–372, jul 1994.

R. Bridson, R. P. Fedkiw, and J. Anderson. Robust Treatment of Collisions, Contact, and Friction for Cloth Animation. *ACM Transactions on Graphics*, 21(3):594–603, jul 2002.

R. Bridson, S. Marino, and R. Fedkiw. Simulation of clothing with folds and wrinkles. In *2003 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 28–36, aug 2003.

R. Bridson, R. Fedkiw, and M. Muller-Fischer. Fluid simulation course notes. *SIGGRAPH '06 Courses*, Jul 2006.

J. Brown, J.-C. Latombe, and K. Montgomery. Real-time knot-tying simulation. *The Visual Computer*, 20(2):165–179, 2004.

P. B. Canham. The minimum energy of bending as a possible explanation of the biconcave shape of the human red blood cell. *Journal of theoretical biology*, 26(1):61, 1970.

S. Capell, S. Green, B. Curless, T. Duchamp, and Z. Popović. Interactive Skeleton-Driven Dynamic Deformations. *ACM Transactions on Graphics*, 21(3):586–593, jul 2002.

S. Capell, M. Burkhart, B. Curless, T. Duchamp, and Z. Popović. Physically Based Rigging for Deformable Characters. In *SCA '05*, pages 301–310, jul 2005.

M. Carignan, Y. Yang, N. M. Thalmann, and D. Thalmann. Dressing Animated Synthetic Actors with Complex Deformable Clothes. In *Proceedings of SIGGRAPH*, pages 99–104, 1992.

J. Chang, D. X. Shepherd, and J. J. Zhang. Cosserat-beam-based dynamic response modelling. *Computer Animation and Virtual Worlds*, 18(4–5):429–436, 2007.

Y. Chang, K. Bao, Y. Liu, J. Zhu, and E. Wu. A particle-based method for viscoelastic fluids animation. *VRST '09*, pages 111–117, Nov 2009.

N. Chentanez, R. Alterovitz, D. Ritchie, L. Cho, K. K. Hauser, K. Goldberg, J. R. Shewchuk, and J. F. O'Brien. Interactive Simulation of Surgical Needle Insertion and Steering. *ACM Transactions on Graphics*, 28(3):88:1–88:10, jul 2009.

S. Chiu-Webster and J. R. Lister. The fall of a viscous thread onto a moving surface: a 'fluid-mechanical sewing machine'. *J. Fluid Mech.*, 569:89–111, 2006.

B. Choe, M. G. Choi, and H.-S. Ko. Simulating complex hair with robust collision handling. In *SCA '05*, pages 153–160, 2005.

K.-J. Choi and H.-S. Ko. Stable but Responsive Cloth. *ACM Transactions on Graphics*, 21(3):604–611, jul 2002.

K.-J. Choi and H.-S. Ko. Research problems in clothing simulation. *Computer-Aided Design*, 37(6):585–592, 2005.

P. G. Ciarlet. *Mathematical Elasticity - Volume III: Theory of Shells*. North Holland, 2000. ISBN 0444828915.

F. Cirak, M. Ortiz, and P. Schröder. Subdivision Surfaces: A New Paradigm for Thin-Shell Finite-Element Analysis. *Internat. J. Numer. Methods Engrg.*, 47(12):2039–2072, 2000.

U. Clarenz, U. Diewald, G. Dziuk, M. Rumpf, and R. Rusu. A finite element method for surface restoration with smooth boundary conditions. *Computer Aided Geometric Design*, 21(5):427–445, 2004.

S. Clavet, P. Beaudoin, and P. Poulin. Particle-based viscoelastic fluid simulation. *SCA '05*, Jul 2005.

M. F. Cohen. Interactive spacetime control for animation. In *Proc. of SIGGRAPH '92*, pages 293–302, jul 1992.

D. Cohen-Steiner and J. M. Morvan. Restricted Delaunay triangulations and normal cycle. In *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*, pages 312–321. ACM, 2003.

D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM Transactions on Graphics*, 23(3):905–914, aug 2004.

F. Cordier and N. Magnenat-Thalmann. A Data-driven Approach for Real-Time Clothes Simulation. *CGF*, 24(2):173–183, jun 2005.

M. Crouzeix and P. A. Raviart. Conforming and non-conforming finite elements for solving stationary Stokes equations. *RAIRO Anal. Numer.*, 7:33–76, 1973.

L. D. Cutler, R. Gershbein, X. C. Wang, C. Curtis, E. Maigret, L. Prasso, and P. Farson. An Art-Directed Wrinkle System for CG Character Clothing. In *SCA '05*, pages 117–126, jul 2005.

R. de Vries. Evaluating changes of writhe in computer simulations of supercoiled DNA. *J. Chem. Phys.*, 122:064905, 2005.

K. Deckelnick, G. Dziuk, and C. M. Elliott. Fully Discrete Semi-Implicit Second order Splitting for Anisotropic Surface Diffusion of Graphs. *SINUM*, 43:1112–1138, 2005.

M. Desbrun and M. Gascuel. Smoothed particles: A new paradigm for animating highly deformable bodies. *Computer Animation and Simulation*, pages 61–76, Jan 1996.

M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit Fairing of Irregular Meshes Using Diffusion and Curvature Flow. In *Proceedings of SIGGRAPH 99*, Computer Graphics Proceedings, Annual Conference Series, pages 317–324, aug 1999a.

M. Desbrun, P. Schröder, and A. Barr. Interactive Animation of Structured Deformable Objects. In *Graphics Interface '99*, pages 1–8, jun 1999b.

J. N. Dewynne, J. R. Ockendon, and P. Wilmott. A systematic derivation of the leading-order equations for extensional flows in slender geometries. *J. Fluid Mech.*, 24:323–338, 1992.

M. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976.

M. do Carmo. *Riemannian Geometry*. Mathematics: Theory and Applications. Birkhäuser, 1992.

J. Eggers and T. F. Dupont. Drop formation in a one-dimensional approximation of the Navier–Stokes equation. *J. Fluid Mech.*, 262:205–221, 1994.

V. M. Entov and A. L. Yarin. The dynamics of thin liquid jets in air. *J. Fluid Mech.*, 140: 91–111, 1984.

O. Etzmuss, B. Eberhardt, and M. Hauth. Implicit-Explicit Schemes for Fast Animation with Particle Systems. In *Computer Animation and Simulation 2000*, pages 138–151, aug 2000.

O. Etzmuss, M. Keckeisen, and W. Strasser. A fast finite element solution for cloth modelling. *Proceedings 11th Pacific Conference on Computer Graphics and Applications*, pages 244–251, 2003.

R. S. Falk and J.-M. Xu. Convergence of a Second-Order Scheme for the Nonlinear Dynamical Equations of Elastic Rods. *SJNA*, 32(4):1185–1209, 1995.

R. Fattal and D. Lischinski. Target-driven smoke animation. *ACM Transactions on Graphics*, 23(3):441–448, aug 2004.

R. Fedkiw, J. Stam, and H. Jensen. Visual simulation of smoke. *SIGGRAPH '01*, pages 15–22, Aug 2001.

C. R. Feynman. Modeling the Appearance of Cloth. Master's thesis, Massachusetts Institute of Technology, May 1986.

N. Foster and R. Fedkiw. Practical animation of liquids. *SIGGRAPH '01*, pages 23–30, Aug 2001.

N. Foster and D. Metaxas. Modeling the motion of a hot, turbulent gas. *SIGGRAPH '97*, pages 181–188, Aug 1997.

F. Frenet. *Sur les courbes à double courbure*. PhD thesis, Toulouse, 1847.

F. B. Fuller. The Writhing Number of a Space Curve. *PNAS*, 68(4):815–819, 1971.

F. B. Fuller. Decomposition of the linking number of a closed ribbon: a problem from molecular biology. *PNAS*, 75(8):3557–3561, 1978.

N. Galoppo, M. A. Otaduy, P. Mecklenburg, M. Gross, and M. C. Lin. Fast Simulation of Deformable Models in Contact Using Dynamic Deformation Textures. In *SCA '06*, pages 73–82, sep 2006.

D. Gerszewski, H. Bhattacharya, and A. Bargteil. A point-based method for animating elastoplastic solids. *SCA '09*, Aug 2009.

Y. Gingold, A. Secord, J. Han, E. Grinspun, and D. Zorin. Simulating fracture and tearing of thin shells. Technical report, NYU, 2004.

M. Gleicher, H. J. Shin, L. Kovar, and A. Jepsen. Snap-together motion: assembling run-time animations. In *2003 ACM Symposium on Interactive 3D Graphics*, pages 181–188, apr 2003.

T. G. Goktekin, A. W. Bargteil, and J. F. O'Brien. A method for animating viscoelastic fluids. *ACM Transactions on Graphics*, 23(3):463–468, aug 2004.

R. Goldenthal, D. Harmon, R. Fattal, M. Bercovier, and E. Grinspun. Efficient Simulation of Inextensible Cloth. *ACM Transactions on Graphics*, 26(3):49:1–49:7, jul 2007.

R. E. Goldstein and S. A. Langer. Nonlinear Dynamics of Stiff Polymers. *Phys. Rev. Lett.*, 75(6):1094–1097, Aug 1995.

A. Goriely. Twisted Elastic Rings and the Rediscoveries of Michell's Instability. *Journal of Elasticity*, 84:281–299, 2006.

A. Goriely and M. Tabor. Spontaneous Helix Hand Reversal and Tendril Perversion in Climbing Plants. *Phys. Rev. Lett.*, 80(7):1564–1567, Feb 1998.

N. K. Govindaraju, D. Knott, N. Jain, I. Kabul, R. Tamstorf, R. Gayle, M. C. Lin, and D. Manocha. Interactive collision detection between deformable models using chromatic decomposition. *ACM Transactions on Graphics*, 24(3):991–999, aug 2005.

S. Goyal, N. C. Perkins, and C. L. Lee. Non-linear dynamic intertwining of rods with self-contact. *International Journal of Non-Linear Mechanics*, 43(1):65–73, 2008.

M. Grégoire and E. Schömer. Interactive simulation of one-dimensional flexible parts. *Computer-Aided Design*, 39(8):694–707, 2007. Solid and Physical Modeling 2006.

E. Grinspun and M. Wardetzky, editors. *Discrete Differential Geometry: An Applied Introduction*. Course Notes. SIGGRAPH ASIA, 2008.

E. Grinspun, P. Krysl, and P. Schröder. CHARMS: A Simple Framework for Adaptive Simulation. *ACM Transactions on Graphics*, 21(3):281–290, July 2002.

E. Grinspun, A. N. Hirani, M. Desbrun, and P. Schröder. Discrete shells. In *SCA '03*, pages 62–67, aug 2003.

E. Grinspun, Y. Gingold, J. Reisman, and D. Zorin. Computing discrete shape operators on general meshes. *Computer Graphics Forum*, 25(3), 2006.

S. Hadap. Oriented Strands - Dynamics of Stiff Multi-Body System. In *2006 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 91–100, sep 2006.

S. Hadap and N. Magnenat-Thalmann. Modeling Dynamic Hair as a Continuum. *Computer Graphics Forum*, 20(3):329–338, 2001.

S. Hadap, E. Bangarter, P. Volino, and N. Magnenat-Thalmann. Animating Wrinkles on Clothes. In *IEEE Viz. '99*, pages 175–182, oct 1999.

S. Hadap, M.-P. Cani, M. Lin, T.-Y. Kim, F. Bertails, S. Marschner, K. Ward, and Z. Kačić-Alesić. *Strands and hair: modeling, animation, and rendering*, pages 1–150. ACM SIGGRAPH 2007 Courses, 2007.

E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer Series in Computational Mathematics. Springer, 2006.

A. J. Hanson. *Visualizing Quaternions*. Morgan Kaufmann/Elsevier, San Francisco, CA, 2006.

H. Hasimoto. Motion of a Vortex Filament and its Relation to Elastica. *Journal of the Physical Society of Japan*, 31(1):293–294, 1971.

R. Haumann. 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*. 1987.

M. Hauth. *Visual Simulation of Deformable Models*. PhD thesis, University of Tübingen, 2004.

M. Hauth and O. Etzmuss. A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods. *Computer Graphics Forum*, 20(3):319–328, 2001.

M. Hauth, O. Etzmuss, and W. Straßer. Analysis of numerical methods for the simulation of deformable models. *The Visual Computer*, 19(7-8):581–600, 2003.

W. Helfrich. Elastic properties of lipid bilayers: theory and possible experiments. *Zeitschrift für Naturforschung. Teil C: Biochemie, Biophysik, Biologie, Virologie*, 28(11):693, 1973.

K. Hildebrandt and K. Polthier. Anisotropic Filtering of Non-Linear Surface Features. *Computer Graphics Forum*, 23(3):391–400, sep 2004.

K. Hildebrandt, K. Polthier, and M. Wardetzky. On the convergence of metric and geometric properties of polyhedral surfaces. ZIB-Report ZR-05-24, 2005.

J.-M. Hong and C.-H. Kim. Discontinuous fluids. *SIGGRAPH '05*, pages 915–920, Jul 2005.

D. H. House and D. E. Breen, editors. *Cloth modeling and animation*. A. K. Peters, Ltd., Natick, MA, USA, 2000. ISBN 1-56881-090-3.

L. Hsu, R. Kusner, and J. Sullivan. Minimizing the squared mean curvature integral for surfaces in space forms. *Experiment. Math.*, 1(3):191–207, 1992. ISSN 1058-6458.

T. J. R. Hughes. *Finite Element Method - Linear Static and Dynamic Finite Element Analysis*. Prentice-Hall, Englewood Cliffs, 1987.

G. Irving. *Methods for the physically based simulation of solids and fluids*. PhD thesis, Stanford University, 2007.

G. Irving, J. Teran, and R. Fedkiw. Invertible finite elements for robust simulation of large deformation. In *SCA '04*, pages 131–140, jul 2004.

J. M. Kaldor, D. L. James, and S. Marschner. Efficient Yarn-based Cloth with Adaptive Contact Linearization. *ACM Transactions on Graphics*, 29(4), Jul 2010.

M. Keckeisen, S. Kimmerle, B. Thomaszewski, and M. Wacker. Modelling Effects of Wind Fields in Cloth Animations. In *Journal of WSCG*, volume Vol. 12, pages 205–212, 2004.

A. Kecskeméthy and M. Tändl. *Advances in Robot Kinematics*, chapter A robust model for 3D tracking in object-oriented multibody systems based on singularity-free Frenet framing, pages 255–264. Springer Netherlands, 2006.

L. Kharevych, W. Yang, Y. Tong, E. Kanso, J. E. Marsden, P. Schröder, and M. Desbrun. Geometric, Variational Integrators for Computer Animation. In *2006 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 43–52, sep 2006.

D. Kim, O.-Y. Song, and H.-S. Ko. Stretching and wiggling liquids. *SIGGRAPH Asia '09*, page 120, Dec 2009.

S. Kircher and M. Garland. Editing arbitrarily deforming surface animations. *ACM Transactions on Graphics*, 25(3):1098–1107, jul 2006.

G. Kirchhoff. Über das Gleichgewicht und die Bewegung eines unendlich dünnen elastischen Stabes. *Journal für die reine und angewandte Mathematik*, 56:285–313, 1859.

I. Klapper. Biological Applications of the Dynamics of Twisted Elastic Rods. *J. Comp. Phys.*, 125(2):325–337, 1996.

J. T. Klosowski, M. Held, J. S. B. Mitchell, H. Sowizral, and K. Zikan. Efficient Collision Detection Using Bounding Volume Hierarchies of k-DOPs. *IEEE TVCG*, 4(1):21–36, January-March 1998.

R. Kondo, T. Kanai, and K. Anjyo. Directable Animation of Elastic Objects. In *SCA '05*, pages 127–134, jul 2005.

C. Lanczos. *The Variational Principles of Mechanics*. Dover, 4$^{\text{th}}$ edition, 1986.

L. D. Landau and E. M. Lifshitz. *Theory of Elasticity (Course of Theoretical Physics)*. Pergamon Press, 2$^{\text{nd}}$ edition, 1981.

M. Le Bret. Catastrophic Variation of Twist and Writhing of Circular DNAs with Constraint? *Biopolymers*, 18(7):1709–1725, 1979.

M. Le Merrer, J. Seiwert, D. Quéré, and C. Clanet. Shapes of hanging viscous filaments. *EPL*, 84:56004, 2008.

S. Lee, S. Olsen, and B. Gooch. Interactive 3D fluid jet painting. *NPAR '06*, pages 97–104, Jun 2006.

J. Lenoir, S. Cotin, C. Duriez, and P. Neumann. Interactive physically-based simulation of catheter and guidewire. *Computer & Graphics*, 30(3):417–423, 2006.

C.-C. Lin and H. R. Schwetlick. On the Geometric Flow of Kirchhoff Elastic Rods. *SJAM*, 65(2):720–736, 2004.

S. P. Lloyd. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 28:129–137, 1982.

F. Losasso, F. Gibou, and R. Fedkiw. Simulating water and smoke with an octree data structure. *ACM Transactions on Graphics*, 23(3):457–462, Aug 2004.

J. Loviscach. Wrinkling Coarse Meshes on the GPU. *CGF*, 25(3):467–476, sep 2006.

L. Ma, J. Hu, and G. Baciu. Generating seams and wrinkles for virtual clothing. In *ACM VRCIA '06*, pages 205–211, 2006.

J. H. Maddocks. Stability of nonlinearly elastic rods. *Archive for Rational Mechanics and Analysis*, 85(4):311–354, 1984.

J. H. Maddocks and D. J. Dichmann. Conservation laws in the dynamics of rods. *Journal of elasticity*, 34:83–96, 1994.

N. Magnenat-Thalmann and P. Volino. From early draping to haute couture models: 20 years of research. *The Visual Computer*, 21(8-10):506–519, 2005.

L. Mahadevan, W. S. Ryu, and A. D. T. Samuel. Fluid 'rope trick' investigated. *Nature*, 392(140):502, 1998.

L. E. Malvern. *Introduction to the Mechanics of a Continuous Medium*. Prentice-Hall, Englewood Cliffs, NJ, 1969.

J. E. Marsden and T. Ratiu. *Introduction to Mechanics and Symmetry*, volume 17 of *Texts in Applied Mathematics*. Springer-Verlag, 2nd edition, 1994.

A. McNamara, A. Treuille, Z. Popović, and J. Stam. Fluid control using the adjoint method. *ACM Transactions on Graphics*, 23(3):449–456, aug 2004.

C. Mercat. Discrete Riemann Surfaces and the Ising Model. *Communications in Mathematical Physics*, 218(1):177–216, 2001.

M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In H.-C. Hege and K. Polthier, editors, *Visualization and Mathematics III*, pages 113–134. Springer-Verlag, Heidelberg, 2003.

G. Miller and A. Pearce. Globular dynamics: A connected particle system for animating viscous fluids. *COMP. GRAPH.*, pages 305–309, Jan 1989. ar.

S. W. Morris, J. H. P. Dawes, N. M. Ribe, and J. R. Lister. Meandering instability of a viscous thread. *Phys. Rev. E (Statistical, Nonlinear, and Soft Matter Physics)*, 77(6): 066218, 2008.

M. Müller, D. Charypar, and M. Gross. Particle-based fluid simulation for interactive applications. *SCA '03*, pages 154–159, Jan 2003.

M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa. Point based animation of elastic, plastic and melting objects. *SCA '04*, pages 141–151, Aug 2004.

A. Nealen, M. Muller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. *Computer Graphics Forum*, 25(4):1–24, 2006.

H. N. Ng and R. L. Grimsdale. Computer Graphics Techniques for Modeling Cloth. *IEEE Comput. Graph. & Appl.*, 16(5):28–41, 1996.

J. F. O'Brien, A. W. Bargteil, and J. K. Hodgins. Graphical Modeling and Animation of Ductile Fracture. *ACM Transactions on Graphics*, 21(3):291–294, jul 2002.

C. M. Oishi, M. F. Tomé, J. A. Cuminato, and S. McKee. An implicit technique for solving 3D low Reynolds number moving free surface flows. *J. Comput. Phys.*, 227(16):7446–7468, 2008.

D. K. Pai. STRANDS: Interactive Simulation of Thin Solids using Cosserat Models. *Computer Graphics Forum*, 21(3):347–352, 2002.

S. Panda, N. Marheineke, and R. Wegener. Systematic derivation of an asymptotic model for the dynamics of curved viscous fibers. *Math. Meth. Appl. Sci.*, 31(10):1153–1173, 2008.

S. I. Park and J. K. Hodgins. Capturing and animating skin deformation in human motion. *ACM Transactions on Graphics*, 25(3):881–889, jul 2006.

J. Phillips, A. Ladd, and L. E. Kavraki. Simulated knot tying. In *Proceedings of ICRA 2002*, volume 1, pages 841–846, 2002.

U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experim. Math.*, 2:15–36, 1993.

J. C. Platt and A. H. Barr. Constraint Methods for Flexible Models. In *Computer Graphics (Proceedings of SIGGRAPH 88)*, pages 279–288, aug 1988.

K. Polthier. Polyhedral Surfaces of Constant Mean Curvature, 2002. Habilitationsschrift, TU Berlin.

J. Popović, S. Seitz, M. Erdmann, Z. Popović, and A. Witkin. Interactive Manipulation of Rigid Body Simulations. In *Proc. of SIGGRAPH '00*, pages 209–218, July 2000.

J. Popović, S. M. Seitz, and M. Erdmann. Motion sketching for control of rigid-body simulations. *ACM Transactions on Graphics*, 22(4):1034–1054, oct 2003.

X. Provot. Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior. In *Graphics Interface '95*, pages 147–154, may 1995.

S. Radev, B. Chavdarov, and A. L. Yarin. Buckling of thin fluid jets and threads. *Fluid Dynamics*, 22(4):525–532, 1987.

R. Radovitzky and M. Ortiz. Error estimation and adaptive meshing in strongly nonlinear dynamic problems. *Comput. Methods Appl. Mech. Eng*, 172(1-4):203–240, 1999.

A. Rafiee, M. T. Manzari, and M. Hosseini. An incompressible SPH method for simulation of unsteady viscoelastic free-surface flows. *Int. J. Non Linear Mech.*, 42(10):1210–1223, 2007.

N. Rasmussen, D. Enright, D. Nguyen, S. Marino, N. Sumner, W. Geiger, S. Hoon, and R. Fedkiw. Directable photorealistic liquids. In *SCA '04*, pages 193–202, jul 2004.

N. M. Ribe. Coiling of viscous jets. *Proc. Math., Phys. and Eng. Sci.*, pages 3223–3239, 2004.

N. M. Ribe, H. E. Huppert, M. A. Hallworth, M. Habibi, and D. Bonn. Multiple coexisting states of liquid rope coiling. *J. Fluid Mech.*, 555(1):275–297, 2006.

M. B. Rubin. *Cosserat Theories: Shells, Rods and Points*. Kluwer Academic Publishers, 2000.

O. Schenk and K. Gärtner. On fast factorization pivoting methods for symmetric indefinite systems. *Elec. Trans. Numer. Anal.*, pages 158–179, 2006.

R. Schneider and L. Kobbelt. Geometric fairing of irregular meshes for free-form surface design. *Computer Aided Geometric Design*, 18(4):359–379, may 2001.

J. A. Serret. Sur quelques formules relatives à la théorie des courbes à double courbure. *J. de Math.*, 16, 1851.

A. Shabana. *Computational Dynamics*. John Wiley & Sons, New York, 2nd edition, 2001.

L. Shi and Y. Yu. Controllable smoke animation with guiding objects. *ACM Transactions on Graphics*, 24(1):140–164, jan 2005.

E. Sifakis, I. Neverov, and R. Fedkiw. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM Transactions on Graphics*, 24(3):417–425, aug 2005.

K. Singh and E. Kokkevis. Skinning Characters using Surface Oriented Free-Form Deformations. In *GI '00*, pages 35–42, 2000.

M. Skorobogatiy and L. Mahadevan. Folding of viscous sheets and filaments. *EPL*, 52(5): 532–538, 2000.

J. Smith, A. Witkin, and D. Baraff. Fast and Controllable Simulation of the Shattering of Brittle Objects. *CGF*, 20(2):81–91, 2001.

R. Smith. Open Dynamics Engine, 2008. URL `http://www.ode.org/`.

A. Sommerfeld. *Mechanics of deformable bodies*, volume 2 of *Lectures on theoretical physics*. Academic Press, 1964.

J. Spillmann and M. Teschner. CORDE: Cosserat Rod Elements for the Dynamic Simulation of One-Dimensional Elastic Objects. In *2007 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 63–72, aug 2007.

J. Spillmann and M. Teschner. An Adaptive Contact Model for the Robust Simulation of Knots. *CGF*, 27(2):497–506, 2008. ISSN 1467-8659.

J. Stam. Stable fluids. *SIGGRAPH '99*, pages 121–128, Jul 1999.

K. Steele, D. Cline, P. Egbert, and J. Dinerstein. Modeling and rendering viscous liquids. *CAVW*, pages 183–192, Jan 2004.

D. Stora, P.-O. Agliati, M.-P. Cani, F. Neyret, and J.-D. Gascuel. Animating lava flows. *GI '99*, pages 203–210, Jan 1999.

G. Strang and G. Fix. *An Analysis of the Finite Element Method*. Wellesley-Cambridge Press, 1973.

J. W. Strutt. *Theory of Sound*, volume 2. Dover Publications, 1945.

R. W. Sumner and J. Popović. Deformation transfer for triangle meshes. *ACM Transactions on Graphics*, 23(3):399–405, aug 2004.

G. I. Taylor. Instability of jets, threads, and sheets of viscous fluid. In Springer, editor, *Proc. 12th Intl Congr. Appl. Mech., Stanford*, page 382, 1968.

J. Teichman and L. Mahadevan. The viscous catenary. *Journal of Fluid Mechanics*, 478: 71–80, 2003.

J. Teran, E. Sifakis, G. Irving, and R. Fedkiw. Robust Quasistatic Finite Elements and Flesh Simulation. In *2005 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 181–190, jul 2005.

D. Terzopoulos and K. Fleischer. Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture. In *Computer Graphics (Proceedings of SIGGRAPH 88)*, pages 269–278, aug 1988.

D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically Deformable Models. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, pages 205–214, jul 1987.

D. Terzopoulos, J. Platt, and K. Fleischer. Heating and melting deformable models. *J. Visual. Comp. Animat.*, 2(2):68–73, 1991.

A. Theetten, L. Grisoni, C. Duriez, and X. Merlhiot. Quasi-Dynamic Splines. In *SPM '07: Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pages 409–414, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-666-0.

A. Theetten, L. Grisoni, C. Andriot, and B. Barsky. Geometrically exact dynamic splines. *Computer-Aided Design*, 40:35–48, 2008.

F. Thomas and O. Johnston. *The Illusion of Life: Disney Animation*. Hyperion Books, New York, 1981.

B. Thomaszewski and M. Wacker. Bending Models for Thin Flexible Objects. In *WSCG Short Communication Proceedings*, 2006.

N. Thürey, R. Keiser, M. Pauly, and U. Rüde. Detail-Preserving Fluid Control. In *SCA '06*, pages 7–15, sep 2006.

A. Treuille, A. McNamara, Z. Popović, and J. Stam. Keyframe Control of Smoke Simulations. *ACM Transactions on Graphics*, 22(3):716–723, jul 2003.

F. R. S. Trouton. On the coefficient of viscous traction and its relation to that of viscosity. *Proc. Royal Soc. London, A*, 77:426–440, 1906.

C. D. Twigg and D. L. James. Many-Worlds Browsing for Control of Multibody Dynamics. *ACM Transactions on Graphics*, 26(3):14–1, jul 2007.

C. D. Twigg and D. L. James. Backward Steps in Rigid Body Simulation. *ACM Transactions on Graphics*, 27(3):25–1, aug 2008.

G. H. M. van der Heijden and J. M. T. Thompson. Helical and Localised Buckling in Twisted Rods: A Unified Analysis of the Symmetric Case. *Nonlinear Dynamics*, 21(1): 71–99, 2000.

G. H. M. van der Heijden, S. Neukirch, V. G. A. Goss, and J. M. T. Thompson. Instability and self-contact phenomena in the writhing of clamped rods. *Int. J. Mech. Sci.*, 45: 161–196, 2003.

P. Volino, M. Courchesne, and N. M. Thalmann. Versatile and efficient techniques for simulating cloth and other deformable objects. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 137–144, New York, NY, USA, 1995. ACM Press. ISBN 0-89791-701-4.

K. Ward, F. Bertails, T.-Y. Kim, S. R. Marschner, M. P. Cani, and M. C. Lin. A Survey on Hair Modeling: Styling, Simulation, and Rendering. *IEEE TVCG*, 13(2):213–234, mar/apr 2007.

M. Wardetzky, M. Bergou, D. Harmon, D. Zorin, and E. Grinspun. Discrete Quadratic Curvature Energies. *Computer Aided Geometric Design*, 24(8-9):499–518, 2007.

J. H. White. A global invariant of conformal mappings in space. *Proceedings of the American Mathematical Society*, 38:162–164, 2000.

M. Wicke, D. Steinemann, and M. Gross. Efficient Animation of Point-Sampled Thin Shells. *CGF*, 24(3):667–676, sep 2005.

T. J. Willmore. Surfaces in Conformal Geometry. *Annals of Global Analysis and Geometry*, 18(3):255–264, 2000.

A. Witkin and D. Baraff, editors. *Physically Based Modeling: Principles and Practice*. Course Notes. ACM SIGGRAPH '01, 2001.

A. Witkin and M. Kass. Spacetime Constraints. In *Proc. of SIGGRAPH '88*, pages 159–168, aug 1988.

A. Witkin and W. Welch. Fast Animation and Control of Nonrigid Structures. In *Proc. of SIGGRAPH '90*, pages 243–252, aug 1990.

C. Wojtan and G. Turk. Fast viscoelastic behavior with thin features. *SIGGRAPH '08*, Aug 2008.

C. Wojtan, P. J. Mucha, and G. Turk. Keyframe Control of Complex Particle Systems Using the Adjoint Method. In *SCA '06*, pages 15–24, sep 2006.

Y. Yang, I. Tobias, and W. K. Olson. Finite element analysis of DNA supercoiling. *J. Chem. Phys.*, 98(2):1673–1686, 1993.

S. Yoshizawa and A. Belyaev. Fair Triangle Mesh Generation via Discrete Elastica. In *GMP2002*, pages 119–123. IEEE, July 2002.

H. Zhu, X. Jin, J. Feng, and Q. Peng. Survey on cloth animation. *Journal of Computer Aided Design & Computer Graphics*, 16(5):613–618, 2004.

O. C. Zienkiewicz and R. L. Taylor. *The finite element method*, volume 1 and 2. Butterworth-Heinemann, 5^{th} edition, 2000.

V. B. Zordan and J. K. Hodgins. Motion Capture-Driven Simulations that Hit and React. In *SCA '02*, pages 89–96, jul 2002.

D. Zorin and P. Schröder, editors. *Subdivision for Modeling and Animation*. Course Notes. SIGGRAPH, 1998.