Unified Simulation of Elastic Rods, Shells, and Solids

Sebastian Martin, ETH Zurich
Peter Kaufmann, ETH Zurich
Mario Botsch, Bielefeld University
Eitan Grinspun, Columbia University
Markus Gross, ETH Zurich

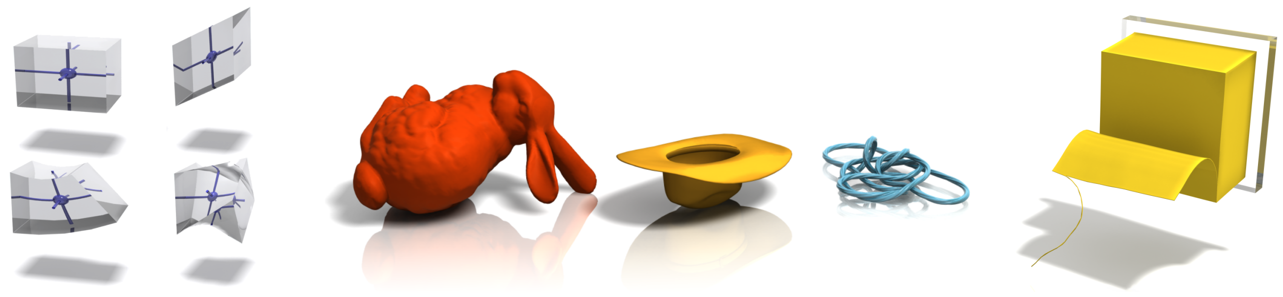


Figure 1: *An elaston measures stretching, shearing, bending, and twisting along any axis (left). An assembly of elastons accurately captures the behavior of elastic materials of any dimension (center), manifold or not, such as this rod cut out of a shell cut out of a cube (right).*

Abstract

We develop an accurate, unified treatment of elastica. Following the method of resultant-based formulation to its logical extreme, we derive a higher-order integration rule, or *elaston*, measuring stretching, shearing, bending, and twisting along any axis. The theory and accompanying implementation do not distinguish between forms of different dimension (solids, shells, rods), nor between manifold regions and non-manifold junctions. Consequently, a single code accurately models a diverse range of elastoplastic behaviors, including buckling, writhing, cutting and merging. Emphasis on convergence to the continuum sets us apart from early unification efforts.

1 Introduction

Elastic bodies take many forms, from long and slender *rods*, to flat and wide *shells*, to thick and bulky *solids*. Over the past decades specialized methods have emerged for the efficient and compelling simulation of each of these forms. But this specialization has opened a Pandora's box: we must debug, extend, and interface between multiple specialized codes. We struggle both with the software interface as well as with the mathematical, or physical model of the interface. And how do we model the physics of objects that do not neatly fit into one of the categories? Junctions, for example, are outside the scope of most specialized models, and are treated as an afterthought. Some shapes transition smoothly (either along their spatial dimension, or as they evolve temporally) between one form and another—must we make a binary decision in categorizing them? If two specialized methods use a different geometric representation (points, triangles, tetrahedra), the question of how to transition is doubly complicated.

Is there an alternative to the specialized models? Can we efficiently and simply simulate a spectrum of forms with a unified computational model and still "get the physics right?"

We argue that a simple, unified treatment spanning rods, shells, and solids is possible and desirable. By *unified,* we mean that the code does not distinguish between forms. We draw motivation from previous unification efforts, but are set apart by our emphasis on physical correctness, specifically convergence to the continuum model. Beyond theoretical appeal, convergence keeps downstream simulation results consistent under upstream resampling or refinement of geometry; convergence is also a prerequisite to smooth adaptive simulations free of popping artifacts.

Contributions We derive a simple quadrature rule for volumetric deformation fields that stably and accurately resolves the stored deformation energy regardless of the (local) form. By evaluating this quadrature rule at points we call *elastons*, we obtain elastoplastic forces acting on the simulation degrees of freedom (DOFs).

Our use of elastons is independent of the representation of the volumetric deformation field, or choice of DOFs. In our work, we choose *generalized moving least squares* (GMLS)—a meshless generalization of Hermite interpolation to three dimensions. We demonstrate efficient, accurate simulations which converge to the smooth underlying continuum formulation, ensuring that simulations are consistent under resampling or refinement of the geometry. We observe excellent agreement with established benchmarks for rods, shells, and solids, a consequence of the theoretically-grounded development of elastons.

But the approach extends beyond the scope of standard benchmarks: without any modification to the implementation, we demonstrate compelling examples on non-manifold geometry (where classical rod or shell models break down) and on hybrid forms that cannot be discretely classified as rods, shells, or solids (where classical rod or shell models do not apply, and naïve implementations of volumetric elastica suffer from poor numerics).

2 Related Work

Terzopoulos et al. [1987] argue that differential geometry provides a natural language for describing the physics of curves, surfaces, and volumetric bodies, naturally establishing a trichotomy of geometric forms whose strains are given by torsion, curvature, and the metric tensor. The exposition mirrors the mechanics literature, where the three forms are each analyzed individually [Malvern 1969]. This sets the stage for principled, specialized, efficient graphical models of rods [Pai 2002; Spillmann and Teschner 2007; Hadap et al.

2007], shells [Cirak et al. 2000; Grinspun et al. 2003], and volumes [O'Brien and Hodgins 1999; Irving et al. 2007; Bargteil et al. 2007], and a more recent interest in mesh-free shells [Wicke et al. 2005; Guo et al. 2006] and volumes [Müller et al. 2004; Gerszewski et al. 2009].

Challenges arising from specialization spur researchers to explore techniques that tie together multiple models. Finite element (FE) packages such as ABAQUS and SOFEA encapsulate a plethora of diverse elements and techniques, often using dynamic method dispatch to provide a unified application interface to distinct underlying codes [Krysl 2005]; additional specialized code is required to capture the physics of interactions between differing models. Sifakis et al. [2007] formulate an "all-purpose glue" to pass forces and constraints between various physical models using *soft* and *hard bindings*; this method models the physics of interactions between two black-box codebases and also easily extends existing codes to handle non-manifold geometry, however it can require the computation of a non-physical mass associated to the binding particles. Glue techniques are attractive when there is a need to quickly combine a number of distinct existing codebases.

However, a gluing strategy does not lighten the burden of maintaining multiple codes (rather it introduces additional code). To keep code manageable and extendable, various researchers advocate sacrificing some benefits of specialization for the simplicity and ultimately scalability of a unified treatment of elastica. Many works in graphics use networks of point masses because they can be connected by any combination of stretching, bending [Baraff and Witkin 1998; Bridson et al. 2003], and altitude springs [Selle et al. 2008]; constraints can replace stiff springs for stabler integration and can also allow for unilateral action [Provot 1995; Müller et al. 2007]. Stam's *Nucleus* [2009] efficiently enforces competing constraints on arbitrary simplicial complexes capturing a diverse range of materials. We are inspired by the goals of generality and simplicity, and depart by additionally asking for a convergent approximation of a continuum formulation. Cosserat points [Rubin 1985] model a small elastic volume equipped with its own mass, DOFs, and elastic energy. Cosserat points must be glued together explicitly by kinematic constraints; since elastons are quadrature points embedded in a separately-defined deformation field, no formulation of glue is necessary.

Some works attempt to approach a continuum result by careful choice of masses and spring coefficients [Etzmuss et al. 2003; Zerbato et al. 2007] using prestressed configurations [Wang and Devarajan 2005; Lloyd et al. 2007] or biquadratic springs [Delingette 2008], or by factoring and approximating FEs [Kikuuwe et al. 2009], with applications to cloth simulation [Volino et al. 2009], however researchers agree with Van Gelder's claim [1998] that it is impossible to expect mass-spring networks to converge to continuum models for the general setting of arbitrary material parameters and network topology.

While mass-spring networks attempt to resolve thin features, other methods compute the dynamics of a volumetric deformation field, embedding arbitrary (thin, degenerate, non-manifold) geometries in the field. The works of Wojtan et al. [2008], and Nesme et al. [2009] show that embedding a high resolution surface in a volumetric field yields a single, unified method that can efficiently and plausibly simulate a diverse range of viscous and plastic materials. Elastic materials might be well-captured by incorporating our proposed deformation field represented by GMLS and integrated using elastons.

3 Elastic Solids

Notation Our derivation begins with the classical theory of solid elastica. Consider a material whose undeformed positions $\bar{\mathbf{x}}(\boldsymbol{\theta})$ are parameterized by curvilinear coordinates $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)^T$ over the material domain Ω. Let the comma denote partial differentiation, e.g., $\mathbf{x}_{,i} \equiv \partial\mathbf{x}/\partial\theta_i$, $\mathbf{u}_{,ik} \equiv \frac{\partial^2 \mathbf{u}}{\partial\theta_i\,\partial\theta_k}$, while the dot $(\cdot)$, cross $(\times)$, and colon $(:)$ denote the vector dot product, vector cross product, and tensor contraction, respectively.

Strain When the material undergoes a deformation, an undeformed point $\bar{\mathbf{x}}(\boldsymbol{\theta})$ is displaced to the new position

$$\mathbf{x}(\boldsymbol{\theta}) \;=\; \bar{\mathbf{x}}(\boldsymbol{\theta}) + \mathbf{u}(\boldsymbol{\theta})\,, \tag{1}$$

The deformation is measured via the linear 3×3 Cauchy strain $\boldsymbol{\epsilon}$

$$\boldsymbol{\epsilon}_{ij} \;=\; \frac{1}{2}\left(\mathbf{u}_{,i}\cdot\bar{\mathbf{x}}_{,j} + \bar{\mathbf{x}}_{,i}\cdot\mathbf{u}_{,j}\right)\,, \tag{2}$$

where $\bar{\mathbf{x}}_{,i}$ and $\mathbf{x}_{,i}$ are the local frames of the undeformed and deformed configuration, respectively. The Cauchy strain is valid only for small displacements; our implementation uses a corotational approach to handle large displacements [Veubeke 1976; Müller et al. 2002; Hauth and Strasser 2004].

Forces In an elastic material, strain leads to restoring forces represented by a 3×3 stress tensor $\boldsymbol{\sigma}$. Assuming a Hookean material yields a linear relation between stress and strain

$$\boldsymbol{\sigma} \;=\; \mathbf{C} : \boldsymbol{\epsilon}\,,$$

where the 4-tensor $\mathbf{C}$ contains the elastic coefficients (Young's modulus, Poisson's ratio) of the material. The stored elastic energy W is the integral of *volumetric energy density*, $\boldsymbol{\epsilon} : \boldsymbol{\sigma}$, over the material domain:

$$W(\mathbf{u}) \;=\; \frac{1}{2}\int_{\Omega} \boldsymbol{\epsilon} : \mathbf{C} : \boldsymbol{\epsilon}\,\mathrm{d}\Omega\,. \tag{3}$$

The internal elastic forces correspond to the variational derivative $\partial W/\partial\mathbf{u}$ of the elastic energy. The dynamic behavior of the material is then described by the governing equation

$$\rho\,\ddot{\mathbf{u}} + \mathbf{f}_d(\dot{\mathbf{u}}) - \frac{\partial W}{\partial\mathbf{u}} \;=\; \mathbf{f}\,,$$

where $\ddot{\mathbf{u}}$ denotes second order time derivative (acceleration), ρ the density of the material, $\mathbf{f}_d$ a damping force, and $\mathbf{f}$ the external forces.

These governing equations are typically discretized using volumetric (e.g., linear or quadratic tetrahedral) FEs [Hughes 2000], or using meshfree Galerkin formulations [Fries and Matthies 2004].

4 Volumetric Thin Shells

If we consider a volumetric surface-like solid whose extent along θ_1 and θ_2 (the "tangent directions") is much greater than along θ_3 (the "normal direction"), we arrive at the special case of *thin shells*. In this case, a naïve discretization using linear or quadratic tetrahedral FEs leads to arbitrarily poor numerical conditioning and slow convergence to the smooth limit (*locking*) and wastefully allocates DOFs along the normal direction [Yang et al. 2000].

These difficulties motivate the development of *resultant-based formulations*, where thin bodies are treated by a reduced set of equations governing a lower-dimensional object—the thin shell *middle surface*—and specifying the displacement field both on the surface and in its (normal) vicinity. The reduced representation offers superior numerical conditioning, convergence, and more efficient allocation of DOFs [Naghdi 1972].

Strain about middle surface For notational convenience, let the middle surface $\mathcal{S}$ be parameterized by the material-domain surface $\boldsymbol{\theta}_0 = (\theta_1, \theta_2, 0)$. Assuming the shell to be sufficiently thin in normal direction, we expand to first-order the positions and displacements:

$$\begin{aligned}\bar{\mathbf{x}}(\boldsymbol{\theta}) &\approx \bar{\mathbf{x}}(\boldsymbol{\theta}_0) + \theta_3\, \bar{\mathbf{x}}_{,3}(\boldsymbol{\theta}_0)\,,\\ \mathbf{u}(\boldsymbol{\theta}) &\approx \mathbf{u}(\boldsymbol{\theta}_0) + \theta_3\, \mathbf{u}_{,3}(\boldsymbol{\theta}_0)\,.\end{aligned}$$

Substituting into (2) yields the strain about the mid-surface

$$\boldsymbol{\epsilon}(\boldsymbol{\theta}) \approx \boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \theta_3\, \boldsymbol{\beta}^3(\boldsymbol{\theta}_0)\,, \tag{4}$$

expressed in terms of the *membrane strain*

$$\boldsymbol{\alpha}_{ij} = \frac{1}{2}\left(\mathbf{u}_{,i}\cdot\bar{\mathbf{x}}_{,j} + \bar{\mathbf{x}}_{,i}\cdot\mathbf{u}_{,j}\right) \tag{5}$$

and the *bending strain* related to direction θ_k

$$\boldsymbol{\beta}^k_{ij} = \frac{1}{2}\left(\mathbf{u}_{,ik}\cdot\bar{\mathbf{x}}_{,j} + \bar{\mathbf{x}}_{,i}\cdot\mathbf{u}_{,jk} + \mathbf{u}_{,i}\cdot\bar{\mathbf{x}}_{,jk} + \bar{\mathbf{x}}_{,ik}\cdot\mathbf{u}_{,j}\right)\,. \tag{6}$$

Equation (4) is an approximation since higher order terms in the thin direction θ_3 have been discarded from (6).

Departure from Kirchhoff-Love models Note that $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}^k$ are volumetric 3×3 tensors. At this point in the derivation, a typical resultant-based formulation would assume that a normal vector to the undeformed mid-surface is deformed such that it retains its orthogonality to the mid-surface. This *Kirchhoff-Love* assumption yields vanishing normal components for the strain, reducing the 3×3 volumetric to 2×2 surface strain tensors. Although not our primary concern at this point, we note this assumption is by nature restrictive, prohibiting the tangential shearing of a shell, an effect visually important for thick shells.

We do not invoke the Kirchhoff-Love assumption. The transition from a volumetric to a surface-based representation would be advantageous for an exclusive treatment of shells, but it is also the primary source of aches in the effort to consistently unite specialized models. By following the classical resultant-based derivation—but only halfway—we gain the notions of measuring stretching separately from bending, while keeping the unified *volumetric* displacement required for unification of the models. With that important remark, we are ready to resume the derivation.

Energy Integration The elastic energy of our volumetric shell model can be derived straightforwardly from (3) by replacing the small strain $\boldsymbol{\epsilon}$ by our approximation (4):

$$W = \frac{1}{2}\int_\Omega \big(\boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \theta_3\,\boldsymbol{\beta}^3(\boldsymbol{\theta}_0)\big) : \mathbf{C} : \big(\boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \theta_3\,\boldsymbol{\beta}^3(\boldsymbol{\theta}_0)\big)\mathrm{d}\Omega.$$

The integration in normal direction can be performed analytically. In doing so, we observe that the cross-terms vanish, leading to an integral of the *surface energy density* over the mid-surface $\mathcal{S}$:

$$W = \frac{h_3}{2}\int_{\mathcal{S}} \boldsymbol{\alpha} : \mathbf{C} : \boldsymbol{\alpha} + \frac{h_3^2}{12}\,\boldsymbol{\beta}^3 : \mathbf{C} : \boldsymbol{\beta}^3\,\mathrm{d}\mathcal{S}\,. \tag{7}$$

Here, h_3 denotes the shell's thickness and $\mathcal{S}$ the mid-surface.

5 Volumetric Thin Rods

Consider next a volumetric curve-like solid whose extent along θ_1 (the "tangent direction") is much greater than along θ_2 and θ_3 (the "normal directions" spanning the "cross-sectional plane"). We derive this special case of thin rods analogously, identifying the reduced geometry (the *centerline* curve), linearizing strain about the centerline (this time along the *two* directions spanning the cross-sectional neighborhood), and again omitting the collapse of the strain tensor into a lower dimension.

Strain about centerline For notational convenience, let the centerline curve Γ be parameterized by the material-domain curve $\boldsymbol{\theta}_0 = (\theta_1, 0, 0)$. For small extents along both normals θ_2 and θ_3, we linearly approximate positions and displacements by

$$\begin{aligned}\bar{\mathbf{x}}(\boldsymbol{\theta}) &\approx \bar{\mathbf{x}}(\boldsymbol{\theta}_0) + \theta_2\, \bar{\mathbf{x}}_{,2}(\boldsymbol{\theta}_0) + \theta_3\, \bar{\mathbf{x}}_{,3}(\boldsymbol{\theta}_0)\,,\\ \mathbf{u}(\boldsymbol{\theta}) &\approx \mathbf{u}(\boldsymbol{\theta}_0) + \theta_2\, \mathbf{u}_{,2}(\boldsymbol{\theta}_0) + \theta_3\, \mathbf{u}_{,3}(\boldsymbol{\theta}_0)\,.\end{aligned}$$

Performing the same steps for the derivation of the small strain yields its linearized version

$$\boldsymbol{\epsilon}(\boldsymbol{\theta}) \approx \boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \theta_2\,\boldsymbol{\beta}^2(\boldsymbol{\theta}_0) + \theta_3\,\boldsymbol{\beta}^3(\boldsymbol{\theta}_0)\,, \tag{8}$$

where the bending strains $\boldsymbol{\beta}^2$ and $\boldsymbol{\beta}^3$ are defined as in (6). Assuming cross-sections to stay normal would lead to the Kirchhoff rod model. However, we again do not further simplify our model, but instead keep the strains volumetric.

Remark on Twist The twist of a rod can be computed as $\mathbf{x}_{,12}\cdot\mathbf{x}_{,3}$ or $\mathbf{x}_{,13}\cdot\mathbf{x}_{,2}$, where the two values are identical (up to their sign) under the Kirchhoff assumptions. The difference in twist between the deformed and undeformed configuration can be measured as $\mathbf{x}_{,12}\cdot\mathbf{x}_{,3} - \bar{\mathbf{x}}_{,12}\cdot\bar{\mathbf{x}}_{,3}$. Using (1) and linearizing in $\mathbf{u}$ this becomes $\mathbf{u}_{,12}\cdot\bar{\mathbf{x}}_{,3} + \mathbf{u}_{,3}\cdot\bar{\mathbf{x}}_{,12}$. From (6) we see that these quantities are part of the strain entries $\boldsymbol{\beta}^2_{13}$, $\boldsymbol{\beta}^2_{31}$, $\boldsymbol{\beta}^3_{12}$, and $\boldsymbol{\beta}^3_{21}$, showing that our bending strain also incorporates a measurement for twisting deformations.

Energy Integration Analytic integration in the normal directions θ_2 and θ_3 yields the remaining one-dimensional integral of *axial energy density* over the rod's centerline Γ:

$$\begin{aligned}W = \frac{h_2 h_3}{2}\int_\Gamma &\boldsymbol{\alpha} : \mathbf{C} : \boldsymbol{\alpha} + \\ &+ \frac{h_2^2}{12}\boldsymbol{\beta}^2 : \mathbf{C} : \boldsymbol{\beta}^2 + \frac{h_3^2}{12}\boldsymbol{\beta}^3 : \mathbf{C} : \boldsymbol{\beta}^3\,\mathrm{d}\Gamma\,,\end{aligned} \tag{9}$$

where h_2 and h_3 denote the thickness of the rod in the two normal directions θ_2 and θ_3, respectively. Again, higher order terms in the thin directions θ_2 and θ_3 are discarded.

6 Elastons

We obtained the stored deformation energy of solids, shells, and rods deforming under the volumetric displacement field $\mathbf{u}$. Since the energy expressions (3), (7), and (9) are distinct, they do not "agree" on a single unified implementation. Hypothetically, we could try to classify each region of the material domain as solid, shell, or rod. This is both complicated and unlikely to be effective: manual classification would fail for materials that drastically deform under cutting or plastic deformation; automatic classification is not a well-posed problem, e.g., not all shapes can be divided into pieces that are unambiguously solids, shells, or rods. Perhaps more fundamental is the observation that because classifications are discrete decisions, a change in classification (e.g., due to cutting or extreme deformation) could result in a jump in elastic forces and consequently popping artifacts. For these reasons, we avoid classification altogether, by following the pattern of derivations of solids (zero normal directions), shells (one normal), and rods (two normals) to its logical conclusion, *elastons* (three normals).

Consider a volumetric point-like solid whose extent along *all three* directions is small. This time the reduced geometry is a point, or elaston, and our notion of strain in the vicinity of this point will measure the linear deformations of stretch and shear at the center, as well as the quadratic deformations of bending and twist along all three "normal" directions. Figure 1, left, depicts these four essential deformation modes.

Linearizing Strain For the elaston centered at $\boldsymbol{\theta}_0 = (0,0,0)$, we perform a first-order Taylor approximation of positions and displacements, this time in all *three* normal directions θ_1, θ_2, θ_3:

$$\begin{aligned}\bar{\mathbf{x}}(\boldsymbol{\theta}) &\approx \bar{\mathbf{x}}(\boldsymbol{\theta}_0) + \sum_{k=1}^{3} \theta_k \, \bar{\mathbf{x}}_{,k}(\boldsymbol{\theta}_0)\,, \\ \mathbf{u}(\boldsymbol{\theta}) &\approx \mathbf{u}(\boldsymbol{\theta}_0) + \sum_{k=1}^{3} \theta_k \, \mathbf{u}_{,k}(\boldsymbol{\theta}_0)\,.\end{aligned}$$

Substituting into (2) yields the strain centered about the elaston

$$\boldsymbol{\epsilon}(\boldsymbol{\theta}) \approx \boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \sum_{k=1}^{3} \theta_k \, \boldsymbol{\beta}^k(\boldsymbol{\theta}_0)\,. \tag{10}$$

Observe that this expression naturally generalizes its shell (4) and rod (8) analogues; in particular, it captures stretching, shearing, bending, and twisting along all three axes.

Energy Integration Recall the two steps we have taken to compute the total energy from the strain. First, we analytically integrate over the zero, one, and two normal directions of a solid, shell, or rod, respectively, to obtain the *tangential energy density*. Finally, we integrate the energy density over the remaining three, two, and one tangent directions of a solid, shell, or rod, respectively.

Correspondingly, we compute the elaston's energy by substituting (10) into (3) to obtain an integral over the elaston's volume Ω_e,

$$\begin{aligned}W = \frac{1}{2}\int_{\Omega_e} \Big(\boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \sum_{k=1}^{3} \theta_k \, \boldsymbol{\beta}^k(\boldsymbol{\theta}_0)\Big) : \\ : \mathbf{C} : \Big(\boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \sum_{k=1}^{3} \theta_k \, \boldsymbol{\beta}^k(\boldsymbol{\theta}_0)\Big)\, \mathrm{d}\Omega\,,\end{aligned}$$

which—since all three directions have thin extent—we can analytically integrate, obtaining

$$W = \frac{V}{2}\Big(\boldsymbol{\alpha}(\boldsymbol{\theta}_0) : \mathbf{C} : \boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \sum_{k=1}^{3} \frac{h_k^2}{12}\, \boldsymbol{\beta}^k(\boldsymbol{\theta}_0) : \mathbf{C} : \boldsymbol{\beta}^k(\boldsymbol{\theta}_0)\Big)\,. \tag{11}$$

Here h_k denotes the thickness of the elaston along direction θ_k and $V = h_1 h_2 h_3$ the volume of the elaston.

As we are about to see, elastons serve as basic building blocks for assembling the elastic energy of any deformable object, independent of its form.

7 Summing Up: A New Integration Rule

The classical goal of resultant-based thin shell models is to reduce the dimensionality of the model from three to two dimensions, thereby simplifying its numerical treatment. The reduction simplifies the formulation because the energy integration can be performed analytically in normal direction.

We adopt a rather unorthodox, alternative perspective: consider a volumetric solid discretized into a family of shells, like the layers of an onion. The deformation energy of each sheet can be measured using the thin shell model. Considering all sheets, such an integration scheme approximates the volumetric elastic energy. The accuracy of the approximation is governed by the resolution of the slicing and converges to the exact energy with increasing number of slices.

Likewise, a rod model serves as an integration rule for shell energies, and by transitivity for volumetric solids.

Therefore, elastons offer the most general integration rule. By placing the elastons along a rod's centerline, or on a shell's mid-surface, or throughout a solid's volume, we can approximate the stored elastic energy of rods, shells, or solids, respectively.

Formally, a set of elastons $e \in \mathcal{E}$ form an integration rule approximating the elastic energy (3) as

$$W = \sum_{e\in\mathcal{E}} \frac{V^e}{2}\Big(\boldsymbol{\alpha}^e : \mathbf{C} : \boldsymbol{\alpha}^e + \sum_{k=1}^{3} \frac{(h_k^e)^2}{12}\, \boldsymbol{\beta}^{ke} : \mathbf{C} : \boldsymbol{\beta}^{ke}\Big)\,, \tag{12}$$

where V^e denotes the volume associated with elaston e and $\boldsymbol{\alpha}^e$ and $\boldsymbol{\beta}^{ke}$ are the membrane and bending strains of elaston e evaluated at its center $\boldsymbol{\theta}_0^e$. This equation describes a convenient approximation of the elastic energy, which allows the treatment of solids, shells, and rods in a unified manner. However, in order to use this integration rule in simulations, there are two further requirements:

- The elastons have to sample the material Ω sufficiently densely, such that all relevant deformations are measured and no undesired modes can appear because they would not be captured by the integration rule. A dense sampling avoids these problems and at the same time guarantees an accurate integration of the elastic energy (see Section 9).
- We also have requirements for an admissible basis of the solution space. Basis functions must be twice differentiable in order to be able to measure bending strains. They need to reproduce constant and linear functions for accurate preservation of linear and angular momenta.

We first discuss the second point.

8 Displacement Discretization

We consider the elaston to be a general-purpose *integration rule* for solids, shells, and rods. The remaining theoretical component is a discretization of the displacement field $\mathbf{u}(\mathbf{x})$; any discretization that is twice weakly differentiable (i.e., has square-integrable second derivatives) is sufficient. We could therefore discretize $\mathbf{u}$ using standard tetrahedral or hexahedral finite elements with quadratic shape functions. This volumetric tessellation, however, would be less suitable for shells and rods. A point-based discretization is not only philosophically compatible with our thinking of elastons as "elastic points", it also allows us to easily take advantage of the agnosticism of elastons to local form, topology (non-manifold junctions), and so forth.

Most previous meshless simulations in graphics [Müller et al. 2004; Pauly et al. 2005; Gerszewski et al. 2009] use a linear *moving least squares* (MLS) discretization due to its high flexibility and good approximation properties (see [Fries and Matthies 2004] for an overview). The undeformed object Ω is sampled by points $\mathbf{x}_1, \ldots, \mathbf{x}_n$, each of which is associated with a displacement DOF $\mathbf{u}_i \in \mathbb{R}^3$. The displacement field $\mathbf{u}(\mathbf{x})$ is represented by a polynomial $\mathbf{a}^T\mathbf{p}(\mathbf{x})$ with a vector of monomials $\mathbf{p}(x,y,z) = (1,x,y,z)^T$

and coefficients $\mathbf{a} = \mathbf{a}(\mathbf{x})$. The latter are determined by a *local* least squares fit to the displacements $\mathbf{u}_i$, i.e., by minimizing

$$J(\mathbf{a}) \;=\; \sum_{i=1}^{n} w(\mathbf{x}-\mathbf{x}_i)\,\left\|\mathbf{a}^T\mathbf{p}(\mathbf{x}_i)-\mathbf{u}_i\right\|^2\,, \qquad (13)$$

where $w(\mathbf{x}-\mathbf{x}_i)$ denotes a classic MLS weighting kernel. We apply the commonly used weighting kernel $w(\mathbf{d}) = (1-\|\mathbf{d}\|^2)^3$ in our implementation [Fries and Matthies 2004]. Analytically minimizing $J(\mathbf{a})$ by setting $\partial J/\partial\mathbf{a} = 0$ yields

$$\begin{aligned}\mathbf{u}(\mathbf{x}) &= \sum_{i=1}^{n}\mathbf{u}_i\,N_i(\mathbf{x})\,,\\ N_i(\mathbf{x}) &= \mathbf{p}(\mathbf{x})^T\,\mathbf{G}^{-1}(\mathbf{x})\,\mathbf{p}(\mathbf{x}_i)\,w(\mathbf{x}-\mathbf{x}_i)\,, \qquad (14)\\ \mathbf{G}(\mathbf{x}) &= \sum_{i=1}^{n} w(\mathbf{x}-\mathbf{x}_i)\,\mathbf{p}(\mathbf{x}_i)\,\mathbf{p}(\mathbf{x}_i)^T\,.\end{aligned}$$

Motivation for GMLS The above equation reveals an important limitation of the classical MLS shape functions $N_i(\mathbf{x})$: To guarantee an invertible matrix $\mathbf{G}(\mathbf{x})$, there have to be sufficiently many samples $\mathbf{x}_i$ supporting the point of evaluation $\mathbf{x}$ (determined by $w(\mathbf{x}-\mathbf{x}_i)$), and *these samples* $\mathbf{x}_i$ *must not be coplanar.*

This condition is unacceptable when simulating shells and rods, which are naturally represented by (locally nearly) coplanar and colinear samplings. We could perhaps artificially construct a volumetric sampling (*away* from the mid-surface or centerline) by adding points along the normal direction(s), but this too is littered with perils: too many DOFs in normal direction lead to rank-deficient systems, since they allow more deformation modes than can actually be measured by the (coplanar or colinear) elastons.

Pursuing a simpler approach, we discretize the displacement field using *generalized moving least squares* (GMLS), an extension of classical MLS approximation to Hermite data [Atluri et al. 1999; Fries and Matthies 2004]. With GMLS we concentrate all displacement information for the normal direction(s) on point $\mathbf{x}_i$ located on the mid-surface or centerline, sidestepping volumetric sampling.

Linear GMLS In addition to its displacement DOF $\mathbf{u}_i$, each sample point $\mathbf{x}_i$ is associated with derivative DOFs $\mathbf{u}_{i,j} \in \mathbb{R}^3$ $(1 \le j \le 3)$ and the coefficients $\mathbf{a}(\mathbf{x})$ are determined by fitting the polynomial $\mathbf{a}^T\mathbf{p}(\mathbf{x})$ to both value and derivative DOFs. This simply amounts to adding to $J(\mathbf{a})$ in (13) the error term $\sum_{i=1}^{n}\sum_{j=1}^{3} w(\mathbf{x}-\mathbf{x}_i)\,\left\|\mathbf{a}^T\mathbf{p}_{,j}(\mathbf{x}_i)-\mathbf{u}_{i,j}\right\|^2$. Minimizing the resulting H^p-like norm leads to

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{n}\left[\mathbf{u}_i N_i(\mathbf{x}) + \sum_{j=1}^{3}\mathbf{u}_{i,j}N_i^j(\mathbf{x})\right]\,, \qquad (15)$$

with additional basis functions for derivative information

$$N_i^j(\mathbf{x}) \;=\; \mathbf{p}(\mathbf{x})^T\,\mathbf{G}^{-1}(\mathbf{x})\,\mathbf{p}_{,j}(\mathbf{x}_i)\,w(\mathbf{x}-\mathbf{x}_i) \qquad (16)$$

and an augmented, generalized matrix $\mathbf{G}$ used in the construction of $N_{i,j}(\mathbf{x})$ in (16) *and* $N_i(\mathbf{x})$ in (14):

$$\mathbf{G}(\mathbf{x}) = \sum_{i=1}^{n} w(\mathbf{x}-\mathbf{x}_i)\left[\mathbf{p}(\mathbf{x}_i)\,\mathbf{p}(\mathbf{x}_i)^T + \sum_{j=1}^{3}\mathbf{p}_{,j}(\mathbf{x}_i)\,\mathbf{p}_{,j}(\mathbf{x}_i)^T\right].$$

Compared to (14), the outer products of monomial derivatives $\mathbf{p}_{,j}$ guarantee a regular matrix $\mathbf{G}$ in any case—for coplanar and colinear samplings, even for a single sample point. This independence of the sampling makes GMLS the ideal choice for discretizing deformation fields of solids, shells, and rods, where each point $\mathbf{x}_i$ now has the 12 DOFs $\mathbf{u}_i,\ \mathbf{u}_{i,1},\ \mathbf{u}_{i,2},\ \mathbf{u}_{i,3} \in \mathbb{R}^3$.

Quadratic GMLS If higher accuracy and faster convergence are desired, we can alternatively use a quadratic polynomial $\mathbf{p}(\mathbf{x}) = (1, x, y, z, xx, xy, xz, yy, yz, zz)^T$ and additionally consider second order derivative DOFs $\mathbf{u}_{i,jk}$ $(1 \le j,k \le 3)$, providing 30 DOFs per sample point $\mathbf{x}_i$. This adds to $J(\mathbf{a})$ a third error term $\sum_{i=1}^{n}\sum_{j,k=1}^{3} w(\mathbf{x}-\mathbf{x}_i)\,\left\|\mathbf{a}^T\mathbf{p}_{,jk}(\mathbf{x}_i)-\mathbf{u}_{i,jk}\right\|^2$ and leads to

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{n}\left[\mathbf{u}_i N_i(\mathbf{x}) + \sum_{j=1}^{3}\mathbf{u}_{i,j}N_i^j(\mathbf{x}) + \sum_{j,k=1}^{3}\mathbf{u}_{i,jk}N_i^{jk}(\mathbf{x})\right],$$

with additional second-order derivative shape functions

$$N_i^{jk}(\mathbf{x}) \;=\; \mathbf{p}(\mathbf{x})^T\,\mathbf{G}^{-1}(\mathbf{x})\,\mathbf{p}_{,jk}(\mathbf{x}_i)\,w(\mathbf{x}-\mathbf{x}_i) \qquad (17)$$

and a matrix $\mathbf{G}(\mathbf{x})$ (for N_i, $N_{i,j}$, and $N_{i,jk}$) that is augmented a second time by $\sum_{i=1}^{n}\sum_{j,k=1}^{3} w(\mathbf{x}-\mathbf{x}_i)\,\mathbf{p}_{,jk}(\mathbf{x}_i)\,\mathbf{p}_{,jk}(\mathbf{x}_i)^T$.

For notational convenience, we denote in the following all shape functions N_i, $N_{i,j}$, $N_{i,jk}$ and DOFs $\mathbf{u}_i$, $\mathbf{u}_{i,j}$, $\mathbf{u}_{i,jk}$ simply by N_i and $\mathbf{u}_i$, respectively. The discretization of the deformation field hence has the form $\mathbf{u}(\mathbf{x}) \approx \sum_i \mathbf{u}_i\,N_i(\mathbf{x})$.

In our simulation framework we integrated both first and second order GMLS. While the first order scheme allows for faster simulations (12 DOFs/sample), the second order discretization (30 DOFs/sample) provides more accurate results and converges to the physically correct solution (see Section 10). The first and second order methods have the same structure, so that a single code can offer an easy trade-off between speed and accuracy.

9 Implementation

The combination of a GMLS-based deformation field and elaston-based integration opens the door to simple and extendable point-based simulation. Our implementation is outlined below.

```
   Input: Point-based material representation $\mathcal{M}$
1  Precomputation
2      Generate GMLS points $\mathbf{x}_i$ by sampling $\mathcal{M}$ (Sec. 9.1)
3      Generate elastons $e$ by sampling $\mathcal{M}$ (Sec. 9.1)
4      Compute elaston stiffness matrices $\mathbf{K}^e$ (Sec. 9.2)
5      Compute mass matrix $\mathbf{M}$ (Sec. 9.2)
6  Simulation loop
7      Assemble global stiffness matrix $\mathbf{K}$ (Sec. 9.2)
8      Boundary conditions assembly (Sec. 9.3)
9      Collision detection and handling (Sec. 9.3)
10     Time integration (Sec. 9.3)
```

We accept as input a high-resolution point cloud $\mathcal{M} = \{\mathbf{m}_1, \mathbf{m}_2, \dots\}$, with associated radii $\{r_1, r_2, \dots\}$, i.e., we represent the material by a set of spheres $B(\mathbf{m}_i, r_i)$. Devoid of connectivity, this format is a flexible intermediary between data that could exist in the form of a surface or volumetric mesh, implicit surface, triangle soup, range scan, or just points.

9.1 Sampling

Given an input cloud, we generate the positions $\{\mathbf{x}_1, \dots \mathbf{x}_n\}$ of GMLS sample points and elaston centers $\{\mathbf{e}_1, \dots, \mathbf{e}_m\}$ by subsampling the dense material point set $\mathcal{M}$. For DOF positions $\mathbf{x}_i$, we use farthest point sampling of the material $\mathcal{M}$, similar to [Adams et al. 2008]. Starting from $\mathbf{x}_1 = \mathbf{m}_1$, subsequent samples $\mathbf{x}_{i+1}$ are

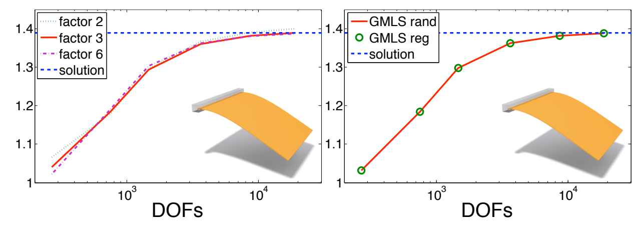


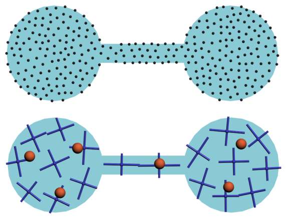
Figure 2: *Left: Convergence behavior for a clamped shell under gravity, for several ratios of elaston density to GMLS sample density. Moving from a double to triple ratio improves convergence, but higher ratios do not lead to further improvements. Right: Randomly rotating the elastons' tangent axes around the normal vector does not change the result; the two curves are coincident. Both plots show the displacement at the rightmost point. The reference solution is computed using a high-resolution Kirchhoff-Love shell.*

picked in a greedy manner to maximize the distance to the points $\{\mathbf{x}_1, \ldots, \mathbf{x}_i\}$ already selected. This sampling strategy results in a uniform distribution, but favors samples at the boundary of the material. In our context, however, samples should ideally be located at the mid-surface or centerline.

We therefore improve the sampling by Lloyd relaxation [Lloyd 1957]: The material $\mathcal{M}$ is partitioned into $\mathcal{M}_1 \cup \cdots \cup \mathcal{M}_n$ by associating each material point $\mathbf{m}_j \in \mathcal{M}$ to its closest sample $\mathbf{x}_i$, yielding a discrete Voronoi diagram. In a second step the samples $\mathbf{x}_i$ are repositioned to the centroids of their Voronoi cells $\mathcal{M}_i$, and these two steps are iterated until convergence. For regions with few material samples in a certain direction, i.e., that locally are shell- or rod-like, this approach leads to centered sample positions. For solid regions the samples are distributed regularly in the volume. Adaptive discretizations with varying sampling density can be achieved through a user-defined grading field.

Simulation accuracy depends on the ratio of densities of elastons $\mathbf{e}_i$ and of GMLS samples $\mathbf{x}_i$ (see Figure 2). To capture all relevant deformation modes, elastons should be sampled at 2–3 times the axial density of $\mathbf{x}_i$. We begin with elaston positions $\mathbf{e}_i = \mathbf{x}_i$ $(1 \le i \le n)$ and add more elastons $\{\mathbf{e}_{n+1}, \ldots, \mathbf{e}_m\}$ by farthest point sampling, followed by Lloyd relaxation to improve the sampling.

For theoretical completeness, our derivation of elastons in Sections 4–7 assumed curvilinear coordinates $\bar{\mathbf{x}}(\boldsymbol{\theta})$ for strain computations. In implementation we find it simpler to construct a per-elaston local flat parameterization, so that each undeformed elaston is a small cuboid. Given the elaston center $\mathbf{e}_i$ and Voronoi region $\mathcal{M}_i$, covariance analysis of the spheres $B(\mathbf{m}_i, r_i)$ yields an orthogonal local frame and eigenvalues $\{\lambda_j\}$. These axes serve as the undeformed first derivatives $\bar{\mathbf{x}}_{,j}$ ("tangent vectors") for the computation of the strains in (5) and (6). Note that rotating two tangents of equal length around their common normal does not change the simulation accuracy (Fig. 2,right).



The incident figure shows material points $\mathbf{m}_i$ (top), GMLS samples $\mathbf{x}_i$ (bottom, red spheres), and elastons $\mathbf{e}_i$ with their local frames (bottom, blue crosses). The relative lengths of the cuboid sides are given by $\sqrt{\lambda_1} : \sqrt{\lambda_2} : \sqrt{\lambda_3}$ and their absolute value is chosen such that the elaston's volume matches that of the Voronoi region $\mathcal{M}_i$, i.e., the exact volume to be approximated by the elaston, ensuring exact representation of total mass independent of the sampling pattern.

As a consequence of our assumption of a locally flat parameterization the second derivatives $\bar{\mathbf{x}}_{,jk}$ used in the computation of the bending strain (6) vanish. The error introduced by this depends on the curvature of the undeformed state, and therefore can be reduced by a curvature-adaptive sampling of elastons, i.e., by adjusting the grading field of the Lloyd clustering. Our experiments have shown that the remaining errors of our discrete elaston integration scheme are insignificant compared to the model discretization error (see Figs. 2, 3). Furthermore, in the limit case the energy of a single elaston approaches the exact continuous energy density at that point, guaranteeing vanishing integration error under refinement.

9.2 System Matrices

After setting up the discretization of the deformation field $\mathbf{u}(\mathbf{x})$ (samples $\mathbf{x}_i$, DOFs $\mathbf{u}_i$) and of the elastic energy W (elaston centers $\mathbf{e}_i$ and axes $\bar{\mathbf{x}}_{,j}$), we proceed to compute the system matrices, i.e., the stiffness and mass matrix. Note that we can precompute the mass matrix and the local elaston stiffness matrices, since they remain constant throughout an elastic simulation. Using corotational strain requires global stiffness reassembly in each time step.

Stiffness Matrix To integrate membrane and bending strains (see (5) and (6)), we need first and second order derivatives $\bar{\mathbf{x}}_{,j}$, $\bar{\mathbf{x}}_{,jk}$ and $\mathbf{u}_{,j}$, $\mathbf{u}_{,jk}$. By construction of our local elaston parameterization, $\bar{\mathbf{x}}_{,j}$ are the elaston's axes and $\bar{\mathbf{x}}_{,jk}$ vanish. Writing the deformation field as $\mathbf{u}(\bar{\mathbf{x}}(\boldsymbol{\theta}))$, we have the first derivatives

$$\mathbf{u}_{,i} \;=\; \frac{\partial \mathbf{u}(\bar{\mathbf{x}}(\boldsymbol{\theta}))}{\partial \theta_i} \;=\; \nabla \mathbf{u}\, \frac{\partial \bar{\mathbf{x}}(\boldsymbol{\theta})}{\partial \theta_i} \;=\; \nabla \mathbf{u}\, \bar{\mathbf{x}}_{,i}\,, \qquad (18)$$

where $\nabla \mathbf{u}$ is the deformation's 3×3 Jacobian matrix with respect to Cartesian coordinates. For the second order derivatives of $\mathbf{u}$ we apply the same projection procedure and exploit the vanishing second order derivatives of $\bar{\mathbf{x}}$, leading to

$$\mathbf{u}_{,ij} \;=\; \bar{\mathbf{x}}_{,i} \cdot \mathbf{H}_{\mathbf{u}}\, \bar{\mathbf{x}}_{,j}\,, \qquad (19)$$

where $\mathbf{H}_{\mathbf{u}}$ is the Hessian of $\mathbf{u}$ with respect to Cartesian coordinates. This allows us to compute both strains easily without constructing a global parameterization.

We perform a classic Galerkin discretization and replace the continuous solution $\mathbf{u}(\mathbf{x})$ by our GMLS approximation $\sum_i \mathbf{u}_i\, N_i(\mathbf{x})$ of Section 8. For each elaston $e \in \mathcal{E}$, this leads to a local stiffness matrix $\mathbf{K}^e$, which we construct using Voigt notation [Hughes 2000]. We identify all basis functions N_i that have support at the elaston's position; for each of these basis functions we compute 6×3 matrices $\mathbf{A}_i$ and $\mathbf{B}_i^k$ corresponding to the membrane strain $\boldsymbol{\alpha}$ and the bending strains $\boldsymbol{\beta}^k$ $(k = 1, 2, 3)$, given in terms of their rows

$$\begin{aligned}
\Big[\mathbf{A}_i\Big]_a &= (\nabla N_i \cdot \bar{\mathbf{x}}_{,a})\, \bar{\mathbf{x}}_{,a}^T\,, \\
\Big[\mathbf{A}_i\Big]_{3+a} &= (\nabla N_i \cdot \bar{\mathbf{x}}_{,b})\, \bar{\mathbf{x}}_{,c}^T + (\nabla N_i \cdot \bar{\mathbf{x}}_{,c})\, \bar{\mathbf{x}}_{,b}^T\,, \\
\Big[\mathbf{B}_i^k\Big]_a &= (\bar{\mathbf{x}}_{,a} \cdot \mathbf{H}_{N_i} \bar{\mathbf{x}}_{,k})\, \bar{\mathbf{x}}_{,a}^T\,, \\
\Big[\mathbf{B}_i^k\Big]_{3+a} &= (\bar{\mathbf{x}}_{,c} \cdot \mathbf{H}_{N_i} \bar{\mathbf{x}}_{,k})\, \bar{\mathbf{x}}_{,b}^T + (\bar{\mathbf{x}}_{,b} \cdot \mathbf{H}_{N_i} \bar{\mathbf{x}}_{,k})\, \bar{\mathbf{x}}_{,c}^T\,,
\end{aligned}$$

with $1 \le a \le 3$, $b = ((a+1) \bmod 3)$, and $c = ((a+2) \bmod 3)$.

Using the 6×6 constitutive tensor $\mathbf{C}$ and considering (11), we compute 3×3 blocks $\mathbf{K}_{ij}^e$ of the elaston's stiffness matrix $\mathbf{K}^e$ as

$$\mathbf{K}_{ij}^e \;=\; V^e \left[\mathbf{A}_i^T \mathbf{C} \mathbf{A}_j \;+\; \sum_{k=1}^{3} \frac{(h_k^e)^2}{12} \mathbf{B}_i^{k\,T} \mathbf{C}\, \mathbf{B}_j^k \right]. \qquad (20)$$

We build such a 3×3 block for all pairs of basis functions N_i and N_j having support at the elaston's position, and assemble the elaston stiffness matrix $\mathbf{K}^e$ from these blocks.

As mentioned earlier, we employ linear strain measures, such that the Hessian of the energy becomes constant, which enables fast and stable simulations by means of corotation [Müller et al. 2002; Hauth and Strasser 2004]. However, for the meshless case the corotation method is not directly applicable. Mezger et al. [2008] present an alternative approach where corotation is performed at each integration point. Similarly, we estimate the local rotation matrix $\mathbf{R}^e$ at each elaston by polar decomposition of the deformation gradient $(\mathbf{I} + \nabla \mathbf{u})$ and replace the blocks $\mathbf{K}^e_{ij}$ by their rotated versions $\mathbf{R}^e \mathbf{K}^e_{ij} \mathbf{R}^{eT}$. The global stiffness matrix $\mathbf{K}$ can then be assembled from the rotated elaston stiffness matrices $\mathbf{K}^e$. Moreover, the vector $\mathbf{R}^e \mathbf{K}^e_{ij}(\mathbf{I} - \mathbf{R}^{eT})\mathbf{x}^0_j$ has to be added to the i-th vector component of the external force $\mathbf{f}$. The $\mathbf{x}^0_j$ are the coefficients of the basis functions N_j when representing the undeformed configuration. They are equal to DOF locations for positional DOFs, equal to the unit vectors $\mathbf{e}_k$ for the first derivative DOFs in direction $\mathbf{e}_k$, and vanish for second order derivative DOFs.

Mass Matrix The mass matrix can also be precomputed, since it does not change as long as the undeformed shape of the material remains unchanged. Because we have non-nodal basis functions we avoid simplifications such as mass lumping and instead compute the mass matrix classically by assembling 3×3 blocks

$$\mathbf{M}_{ij} \; = \; \mathbf{I} \cdot \int_\Omega \rho(\mathbf{x})\, N_i(\mathbf{x})\, N_j(\mathbf{x})\, \mathrm{d}\Omega \,. \tag{21}$$

When numerically performing this integration, we have to pay attention to correctly measure the object's inertia. Consider for instance a straight rod. Using simple Monte-Carlo integration points placed only on the rod's axis leads to wrong dynamics, since no inertia can be measured with respect to rotations about the centerline.

Our higher order elaston integration allows us to compute moments of inertia more accurately, thereby avoiding rotation artifacts. Linearizing the basis functions N_i around each elaston's center $\boldsymbol{\theta}_0 = (0,0,0)$ yields

$$N_i(\boldsymbol{\theta}) \;\approx\; N_i(\boldsymbol{\theta}_0) + \sum_{k=1}^{3} \theta_k\, N_{i,k}(\boldsymbol{\theta}_0)\,.$$

Computing the integral in (21) as a sum of elaston integrals of the linearized shape functions—which can again be evaluated analytically—results in the following approximation:

$$\mathbf{M}_{ij} \; = \; \mathbf{I} \cdot \sum_{e \in \mathcal{E}} V^e \rho^e \left[N_i N_j + \sum_{k=1}^{3} \frac{(h^e_k)^2}{12} N_{i,k} N_{j,k} \right] ,$$

where all functions are evaluated at the respective elaston centers. Since the mass matrix has the same sparsity structure as the stiffness matrix, the performance of the linear system solve during the dynamic simulation is not influenced by computing the mass matrix in this manner. Moreover, the basis function derivatives $N_{i,k}$ are already known from the stiffness matrix integration.

9.3 Implementation Details

Time Integration We employ a simple semi-implicit Euler integration scheme to advance the simulation in time. The linear systems resulting from our discretizations are sparse, symmetric, and positive definite, which makes them ideally suited for efficient linear solvers. We choose a direct sparse Cholesky solver [Chen et al. 2008] due to its robustness and its good scaling properties.

Boundary Conditions For simulations employing nodal basis functions (i.e., $\mathbf{u}(\mathbf{x}_i) = \mathbf{u}_i$) prescribing Dirichlet constraints simply corresponds to fixing vertices. However, GMLS basis functions are not interpolating, requiring a different approach for imposing boundary conditions. In our framework we employ a penalty method, since it is simple to implement, gives satisfactory results, and does not introduce additional DOFs into the system. A target displacement $\mathbf{u}^c$ can be imposed on a subdomain $\Omega^c \subset \Omega$ by adding a term to the elastic energy that penalizes the deviation from the prescribed displacement, weighted by a coefficient γ:

$$W^c(\mathbf{u}) \; = \; \frac{\gamma}{2} \int_{\Omega^c} \|\mathbf{u}(\mathbf{x}) - \mathbf{u}^c(\mathbf{x})\|^2 \, \mathrm{d}\Omega \,.$$

This leads to matrix blocks $\mathbf{K}^c_{ij} = \gamma\, \mathbf{I} \int_{\Omega^c} N_i(\mathbf{x}) N_j(\mathbf{x})\, \mathrm{d}\Omega$ and vectors $\mathbf{f}^c_i = \gamma \int_{\Omega^c} N_i(\mathbf{x})\, \mathbf{u}^c(\mathbf{x})\, \mathrm{d}\Omega$ to be assembled into the stiffness matrix $\mathbf{K}$ and the force vector $\mathbf{f}$, respectively.

Collisions Collision detection and handling is point-based. Since the object is represented by a dense set of spheres $B(\mathbf{m}_j, r_j)$, we detect collisions between material spheres $B(\mathbf{m}_j + \mathbf{u}(\mathbf{m}_j), r_j)$ in the deformed configuration. We also detect collisions with analytically defined objects such as planes and cylinders. We respond to collisions using penalty forces.

10 Results

We start with a quantitative convergence analysis and qualitative evaluations of our approach, followed by experiments that demonstrate the generality of the method. The full animations for these examples are shown in the accompanying video. For visualization we embed a high-resolution triangle mesh into the deformation field, which could, however, be replaced by any sample-based geometry representation (triangle soups, point clouds).

Convergence We performed a series of numerical evaluations to verify the accuracy of our method. Figure 3 shows representative plots for solids, shells, and rods, which are subjected to a gravitational force (and twist for the lower-right rod). The cube is constrained at its top, the shell and rod are clamped at their left-hand side as depicted in the insets of Fig. 3. The thickness of the shell and rod is equal to one percent of the object's side length.

The limit solutions of quadratic GMLS show good correspondence with our reference solutions. We compare to hexahedral FEM for solids, Kirchhoff-Love shells, and analytic solutions for rod bending and twisting. Linear GMLS appears to suffer from locking (artificial stiffness for the rod). Convergence is faster than simple finite elements but slower than highly-specialized methods. Those, however, are only valid in their specific application domain and do not cover the same range of different geometries as our method.

Qualitative Verification We also verified the qualitative behavior of our model on a couple of well-known test cases for shells and rods. Figure 4 shows a thin cylinder that develops the expected buckling patterns as it is compressed. Figure 5 demonstrates that we are able to reproduce the characteristic dynamic behavior of rods, building plectonemes and helical perversions as shown previously by Spillmann and Teschner [2007] and Bergou et al. [2008].

Non-manifold Connections Real-world objects often consist of complex assemblies of different forms of geometry. Our method is able to handle these mixed cases in a unified manner as demonstrated in Fig. 6. These examples would be difficult to realize by combining several specialized methods.

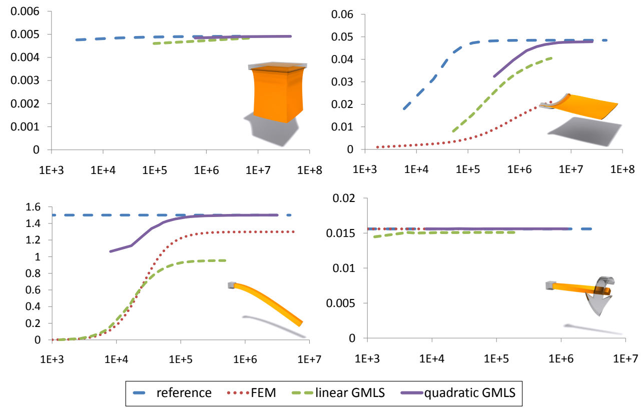


Figure 3: *Convergence of our method compared to standard approaches or analytic solutions for a clamped solid, shell, and rod, subject to gravity and twist. The plots show displacement values of the furthest point for increasing (computational) complexity, measured as the number of non-zeros of the stiffness matrix.*

Plasticity Objects can not only transition between solid, shell, and rod in space (Fig. 6 and Fig. 8), regime changes can also develop over time [Terzopoulos and Fleischer 1988]. In plastic and viscous deformations, for instance, material can be stretched into thin sheets or strands (Fig. 7), whose elastic behavior can correctly be captured by our approach. We adapted the additive plasticity model of [O'Brien et al. 2002; Müller et al. 2004] to our approach by defining *plastic* membrane and bending strains at the elastons. Incorporating these plastic strains into our representation of the elastic energy (12) yields additional plastic forces. Moreover, in order to allow for large plastic deformations, we perform a periodic resampling similar to [Wojtan and Turk 2008], which can easily be integrated thanks to our meshless framework. The multiplicative model of Gerszewski et al. [2009] would be a natural next step.

Cutting Thin structures can also show up dynamically due to cutting or fracturing. We handle cutting by adapting the idea of a connectivity graph of [Steinemann et al. 2006] and the virtual node approach of [Molino et al. 2004]. The connectivity graph for material points $\mathbf{m}_j$ is implicitly defined by connecting two points if their two corresponding spheres $B(\mathbf{m}_j, r_j)$ overlap. The mapping of material points $\mathbf{m}_j$ to elastons $\mathbf{e}_i$ computed during Lloyd clustering then defines the connectivity of the elastons. We cut by disconnecting material points, consequently updating elaston connectivity. When a material region $\mathcal{M}_i$ is split through cutting, we generate new elastons for each of the resulting connected components. A GMLS basis function $N_i(\mathbf{x})$, resp. its DOF position $\mathbf{x}_i$ and value $\mathbf{u}_i$, is then duplicated if its region of influence contains multiple disconnected material components. Figure 1 shows a block of material being cut twice into a very thin slice and rod. Our renderer likewise cuts the embedded triangle mesh.

Virtual nodes also enable efficient simulation of spatially close features. By introducing new elastons and splitting GMLS samples, we keep distinct the motion of close features while avoiding high sampling density (e.g., the fish's spikes in Fig. 8). More details on the implementation of plasticity and cutting can be found in the accompanying supplementary document.

Merging To merge objects we simply resample as per Section 9.1. As soon as elastons of an object fall within influence of another object's GMLS basis, resampling merges the objects. When merging is not desired, we use the virtual node approach described above. Fig. 8 depicts four-bunny fusion.

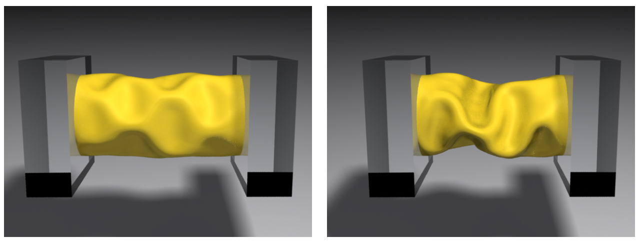


Figure 4: *A cylinder shows the typical buckling patterns as it is getting more and more compressed.*

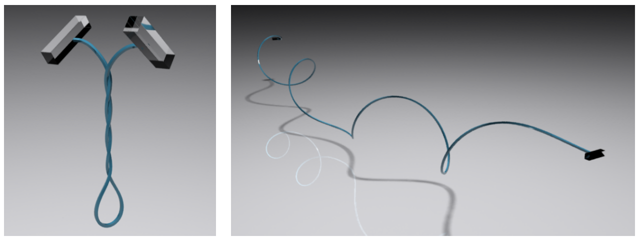


Figure 5: *Twisting a thin rod at both ends generates a* plectoneme *(left). Straightening a helical rod and moving its two ends back together results in a* helical perversion.

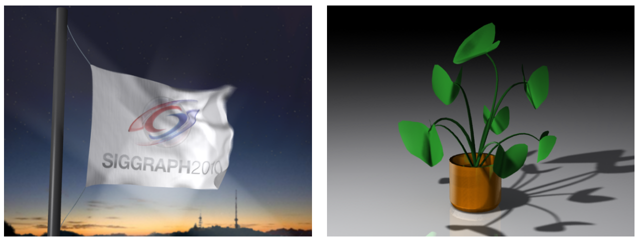


Figure 6: *These models show complex interaction between different types of geometry, handled in a unified way by our approach.*

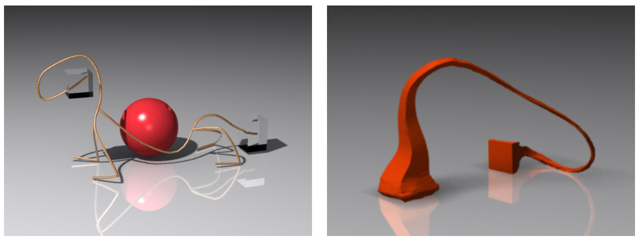


Figure 7: *A ball drops on a wire-dog and deforms it plastically (left). An elastoplastic cuboid is stretched under gravity (right).*

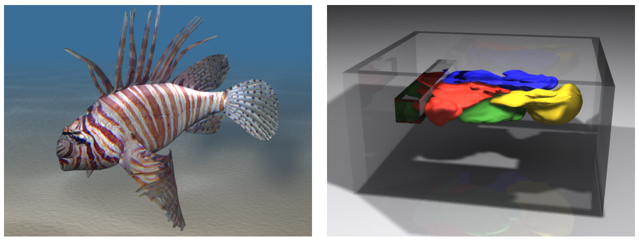


Figure 8: *The fish model consists of solid-, shell-, and rod-like regions (left). Four elastoplastic bunnies are compressed and merged into a solid block (right).*

Timings For a representative selection of examples, Table 1 shows average timings for matrix precomputation, per-frame matrix assembly, and per-frame solution of the involved linear system. It can be observed that the time required for solving the linear system depends not only on the number of DOFs, but also on the density of the stiffness matrix, which decreases from solids over shells to rods. While the measured timings are comparably high, we should not expect a very general method to outperform specialized methods in the sense of accuracy vs. computational costs. We believe that when a unified code is desirable, the simplicity and generality of our approach outweigh this limitation. Moreover, note that the matrix assembly could easily be sped up by parallelizing the strain corotation over all elastons.

Model	#elaston	#samp	#DOFs	t_{pre}	t_{asm}	t_{sol}
Cube cut	2688	50	1.5k	41.96	2.17	0.32
Buckling	2805	528	16k	74.59	5.06	19.62
Plectoneme	100	30	900	0.56	0.021	0.014
Perversion	1200	400	12k	3.86	0.05	0.063
Flag	5670	1490	45k	68.06	2.92	20.14
Flag low-res	4020	434	13k	20.75	0.42	0.52
Plant	1390	501	15k	30.7	4.21	3.79
Wire-dog	630	88	3k	3.35	0.24	0.027
Plastic cuboid	4000	117	1.4k	1.51	0.59	0.054
Fish	5000	540	6.5k	4.48	2.45	0.67
Squeeze bunnies	800	160	2k	0.65	0.34	0.29

Table 1: *Timings (in seconds) for stiffness matrix precomputation (t_{pre}), assembly of the global stiffness matrix (t_{asm}), and linear system solve (t_{sol}), taken on an Intel Core2 Duo 2.4 GHz. The first three columns denote the number of elastons, number of GMLS samples and the number of DOFs, respectively.*

11 Discussion

The synthesis of elaston-based integration with GMLS-based displacement discretization opens exciting avenues for further exploration. While interactive for basic examples, our code will benefit from optimization and concrete improvements: First, our rudimentary collision handling framework should incorporate hierarchical detection accelerations and stabler collision response. Second, our implementation uniformly samples the integration domain. Deformation modes vary widely in space and time, suggesting the profitable use of adaptive placement of integration points guided by geometric or data-driven criteria [An et al. 2008]. Online adaptive refinement of the simulation DOFs would help detailed features such as creases to form at lower cost.

The expressive range of our system could be further explored. We used a simple embedded triangle mesh to visualize a detailed surface, which complicates topological changes such as cutting. Recent advanced embedding strategies, surface tracking or point-based strategies could address this challenge. If symplectic or reversible integration is desired (e.g., to eliminate numerical dissipation), a geometrically nonlinear strain should be introduced into our model. Given the promising results thus far, we plan to further refine our error analysis for the method.

Acknowledgments

The authors are grateful to Petr Krysl and the anonymous reviewers for their helpful comments and suggestions. Sebastian Martin is supported by the Swiss National Science Foundation (grant No. 200021-117756). Mario Botsch is supported by the Deutsche Forschungsgemeinschaft (Center of Excellence in "Cognitive Interaction Technology", CITEC). Eitan Grinspun is supported by the Sloan Foundation, NSF (MSPA Award No. IIS-05-28402, CSR Award No. CNS-06-14770, CAREER Award No. CCF-06-43268), Adobe, ATI, Autodesk, mental images, NVIDIA, Side Effects Software, the Walt Disney Company, and Weta Digital.

References

ADAMS, B., OVSJANIKOV, M., WAND, M., SEIDEL, H.-P., AND GUIBAS, L. J. 2008. Meshless modeling of deformable shapes and their motion. In *Proc. of Symp. on Computer Animation*, 77–86.

AN, S. S., KIM, T., AND JAMES, D. L. 2008. Optimizing cubature for efficient integration of subspace deformations. *ACM Trans. on Graphics 27*, 5, 164:1–164:11.

ATLURI, S. N., CHO, J. Y., AND KIM, H.-G. 1999. Analysis of thin beams, using the meshless local Petrov-Galerkin method, with generalized moving least squares interpolations. *Computational Mechanics*, 24, 334–347.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proc. of ACM SIGGRAPH*, 43–54.

BARGTEIL, A. W., WOJTAN, C., HODGINS, J. K., AND TURK, G. 2007. A finite element method for animating large viscoplastic flow. *ACM Trans. on Graphics 26*, 3, 16.1–16.8.

BERGOU, M., WARDETZKY, M., ROBINSON, S., AUDOLY, B., AND GRINSPUN, E. 2008. Discrete elastic rods. *ACM Trans. on Graphics 27*, 3, 63:1–63:12.

BRIDSON, R., MARINO, S., AND FEDKIW, R. 2003. Simulation of clothing with folds and wrinkles. In *Proc. of Symp. on Computer Animation*, 28–36.

CHEN, Y., DAVIS, T. A., HAGER, W. W., AND RAJAMANICKAM, S. 2008. Algorithm 887: CHOLMOD, supernodal sparse cholesky factorization and update/downdate. *ACM Trans. Math. Softw. 35*, 3, 1–14.

CIRAK, F., ORTIZ, M., AND SCHRÖDER, P. 2000. Subdivision surfaces: A new paradigm for thin-shell finite-element analysis. *Int. J. Numer. Methods Eng. 47*, 12, 2039–2072.

DELINGETTE, H. 2008. Triangular springs for modeling nonlinear membranes. *IEEE Trans. on Visualization and Computer Graphics 14*, 2, 329–341.

ETZMUSS, O., GROSS, J., AND STRASSER, W. 2003. Deriving a particle system from continuum mechanics for the animation of deformable objects. *IEEE Trans. on Visualization and Computer Graphics 9*, 538–550.

FRIES, T. P., AND MATTHIES, H. G. 2004. Classification and overview of meshfree methods. Informatikbericht 2003-03, revised 2004, Institute of Scientific Computing, Technical University Braunschweig.

GERSZEWSKI, D., BHATTACHARYA, H., AND BARGTEIL, A. W. 2009. A point-based method for animating elastoplastic solids. In *Proc. of Symp. on Computer Animation*, 133–138.

GRINSPUN, E., HIRANI, A. N., DESBRUN, M., AND SCHRÖDER, P. 2003. Discrete shells. In *Proc. of Symp. on Computer Animation*, 62–67.

GUO, X., LI, X., BAO, Y., GU, X., AND QIN, H. 2006. Meshless thin-shell simulation based on global conformal parameterization. *IEEE Trans. on Visualization and Computer Graphics 12*, 3, 375–385.

HADAP, S., CANI, M.-P., LIN, M., KIM, T.-Y., BERTAILS, F., MARSCHNER, S., WARD, K., AND KAČIĆ-ALESIĆ, Z. 2007. *Strands and hair: modeling, animation, and rendering*. ACM SIGGRAPH 2007 Courses, 1–150.

HAUTH, M., AND STRASSER, W. 2004. Corotational simulation of deformable solids. In *Proc. of WSCG*, 137–145.

HUGHES, T. J. R. 2000. *The Finite Element Method. Linear Static and Dynamic Finite Element Analysis*. Dover Publications.

IRVING, G., SCHROEDER, C., AND FEDKIW, R. 2007. Volume conserving finite element simulations of deformable models. *ACM Trans. on Graphics 26*, 3, 13.1–13.6.

KIKUUWE, R., TABUCHI, H., AND YAMAMOTO, M. 2009. An edge-based computationally efficient formulation of Saint Venant-Kirchhoff tetrahedral finite elements. *ACM Trans. on Graphics 28*, 1, 1–13.

KRYSL, P. 2005. *A Pragmatic Introduction to the Finite Element Method for Thermal and Stress Analysis*. Pressure Cooker Press, San Diego.

LLOYD, B., SZÉKELY, G., AND HARDERS, M. 2007. Identification of spring parameters for deformable object simulation. *IEEE Trans. on Visualization and Computer Graphics 13*, 1081–1094.

LLOYD, S. 1957. Least squares quantization in PCM. Tech. rep., Tech. rep., Bell Telephone Laboratories, Murray Hill, NJ.

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

MEZGER, J., THOMASZEWSKI, B., PABST, S., AND STRASSER, W. 2008. Interactive physically-based shape editing. In *Proc. of ACM Symp. on Solid and Physical Modeling*, 79–89.

MOLINO, N., BAO, Z., AND FEDKIW, R. 2004. A virtual node algorithm for changing mesh topology during simulation. *ACM Trans. on Graphics 23*, 3, 385–392.

MÜLLER, M., DORSEY, J., MCMILLAN, L., JAGNOW, R., AND CUTLER, B. 2002. Stable real-time deformations. In *Proc. of Symp. on Computer Animation*, 163–170.

MÜLLER, M., KEISER, R., NEALEN, A., PAULY, M., GROSS, M., AND ALEXA, M. 2004. Point-based animation of elastic, plastic and melting objects. In *Proc. of Symp. on Computer Animation*, 141–151.

MÜLLER, M., HEIDELBERGER, B., HENNIX, M., AND RATCLIFF, J. 2007. Position based dynamics. *Journal of Visual Communication and Image Representation 18*, 2, 109 – 118.

NAGHDI, P. 1972. *Handbuch der Physik, Mechanics of Solids II*, vol. VI a/2. Springer, Berlin.

NESME, M., KRY, P. G., JEŘÁBKOVÁ, L., AND FAURE, F. 2009. Preserving topology and elasticity for embedded deformable models. *ACM Trans. on Graphics 28*, 3, 52:1–52:9.

O'BRIEN, J. F., AND HODGINS, J. K. 1999. Graphical modeling and animation of brittle fracture. In *Proc. of ACM SIGGRAPH*, 137–146.

O'BRIEN, J. F., BARGTEIL, A. W., AND HODGINS, J. K. 2002. Graphical modeling and animation of ductile fracture. *ACM Trans. on Graphics 21*, 3, 291–294.

PAI, D. K. 2002. STRANDS: interactive simulation of thin solids using Cosserat models. *Comput. Graphics Forum 21*, 3, 347–352.

PAULY, M., KEISER, R., ADAMS, B., DUTRE, P., GROSS, M., AND GUIBAS, L. J. 2005. Meshless animation of fracturing solids. *ACM Trans. on Graphics 24*, 3, 957–964.

PROVOT, X. 1995. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Graphics Interface '95*, 147–154.

RUBIN, M. B. 1985. On the theory of a cosserat point and its application to the numerical solution of continuum problems. *Journal of Applied Mechanics 52*, 2, 368–372.

SELLE, A., LENTINE, M., AND FEDKIW, R. 2008. A mass spring model for hair simulation. *ACM Trans. on Graphics 27*, 3, 64.1–64.11.

SIFAKIS, E., SHINAR, T., IRVING, G., AND FEDKIW, R. 2007. Hybrid simulation of deformable solids. In *Proc. of Symp. on Computer Animation*, 81–90.

SPILLMANN, J., AND TESCHNER, M. 2007. CoRdE: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *Proc. of Symp. on Computer Animation*, 63–72.

STAM, J. 2009. Nucleus: Towards a unified dynamics solver for computer graphics. In *IEEE International Conference on Computer-Aided Design and Computer Graphics*, 1–11.

STEINEMANN, D., OTADUY, M. A., AND GROSS, M. 2006. Fast arbitrary splitting of deforming objects. In *Proc. of Symp. on Computer Animation*, 63–72.

TERZOPOULOS, D., AND FLEISCHER, K. 1988. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. In *Proc. of ACM SIGGRAPH*, 269–278.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically deformable models. In *Proc. of ACM SIGGRAPH*, 205–214.

VAN GELDER, A. 1998. Approximate simulation of elastic membranes by triangulated spring meshes. *Journal of Graphics Tools 3*, 2, 21–42.

VEUBEKE, B. F. D. 1976. The dynamics of flexible bodies. *International Journal of Engineering Science 14*, 895–913.

VOLINO, P., MAGNENAT-THALMANN, N., AND FAURE, F. 2009. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Trans. on Graphics 28*, 4, 1–16.

WANG, X., AND DEVARAJAN, V. 2005. 1D and 2D structured mass-spring models with preload. *Visual Computer 21*, 7, 429–448.

WICKE, M., STEINEMANN, D., AND GROSS, M. 2005. Efficient animation of point-sampled thin shells. *Comput. Graphics Forum 24*, 667–676.

WOJTAN, C., AND TURK, G. 2008. Fast viscoelastic behavior with thin features. *ACM Trans. on Graphics 27*, 3, 47:1–47:8.

YANG, H., SAIGAL, S., MASUD, A., AND KAPANIA, R. 2000. A survey of recent shell finite elements. *Int. J. Numer. Methods Eng.*, 47, 101–127.

ZERBATO, D., GALVAN, S., AND FIORINI, P. 2007. Calibration of mass spring models for organ simulations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 370–375.