The Dissertation Committee for Hsiao-yu Chen
certifies that this is the approved version of the following dissertation:

Discrete Representation of Elastic Bodies for Physical Simulation

Committee:

Mary Wheeler, Supervisor

Etienne Vouga, Co-Supervisor

George Biros

Leszek F. Demkowicz

Alex Huth

Richard Tsai

Discrete Representation of Elastic Bodies for Physical Simulation

by

Hsiao-yu Chen

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2022

Discrete Representation of Elastic Bodies for Physical Simulation

Publication No. ____________________

Hsiao-yu Chen, Ph.D.
The University of Texas at Austin, 2022

Supervisors: Mary Wheeler
Etienne Vouga

Simulation of elastic objects has received a lot of attention in the past decade in the computer graphics community, due to their ubiquity and importance in our everyday life; some examples include muscles, squishy balls, cloth, and many more. When approaching physical simulation, the computer graphics community has focused on the questions on the representation of 3D data, for example, by building a theory of discrete differential geometry to represent nonlinear deformation, and by inventing algorithms to reconstruct and simulate digital twins of real-world elastic objects. Despite the extensive research, there is not a unified solution that integrates the discrete geometric understanding in the graphics research and combines it with the sophisticated physical modelling in scientific computing. This thesis explores possibilities to bridge the gap between graphics and computational physics by taking the state-of-the-art computer graphics algorithms for representing and discretizing

3D geometry and deformations and equipping these discrete geometric models with physics.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Recent achievements in computer graphics and computer vision have provided us with powerful tools to understand and visualize the world around us. For example, the long-standing Plateau problem of solving for the minimal surface given complex boundary conditions, which is intractable for any analytical method, can be easily solved by representing the curves and surfaces using their discrete differential forms [149]. Surface and volumetric reconstruction is another great example: with a few pictures taken from a consumer-grade camera, we can build a 3D reconstruction of the environment around us with incredible amounts of detail. The rendering techniques have become so sophisticated that we often find it challenging to distinguish an actual picture from the rendering of virtual reconstruction. All these methods provide great tools for us to build a virtual equivalent of a real-world object, but at the same time lack the description of the physics of the object.

On the other end of the spectrum, physicists have derived mathematical models that correctly predict the motion of an object using only a handful of physical parameters, and the same models generalize across different elastic materials and interactions. Elastic simulation has been extensively studied in scientific computing, and algorithms such as the finite element methods [93, 39, 60, 126] and peridynamics [125] have successfully predicted the nonlinear motion of elastic objects.

The work of this thesis explores the recent trend in computer graphics to formulate dynamics based on the discrete geometric model obtained using state-of-the-art computer graphics techniques. We use the language of discrete differential geometry, such as the discrete analog of first and second fundamental forms, and seek a principled method of discretization that preserves some fundamental geometric and topological structures of the underlying continuous model. We then formulate our simulation based on these discrete geometric properties which enable us to apply insight from the smooth setting directly into our new models.

The following chapters of this thesis explore the idea. Starting from a new energy formulation for thin shells that encodes the change due to environmental stimulus in chapter 2, a constraint-based thin shell simulator that achieves faster computation speed than energy-based solvers in chapter 3, and a complete pipeline of learning a real-world deformation of elastic objects in chapter 4.

1.1 Publication

The content of this dissertation has appeared in the following publications:

- Chapter 2: Hsiao-Yu Chen, Arnav Sastry, Wim M. van Rees, Etienne Vouga. "Physical simulation of environmentally induced thin shell deformation", *ACM Transactions on Graphics/Siggraph*, 2018

- Chapter 4: Hsiao-yu Chen, Edith Tretschk, Tuur Stuyck, Petr Kadlecek, Ladislav Kavan, Etienne Vouga, Christoph Lassner. "Virtual Elastic Objects", *Computer Vision and Pattern Recognition Conference(CVPR)*, 2022

The dataset of the Virtual Elastic Objects can be downloaded from the corresponding project site: https://hsiaoyu.github.io/VEO/

Chapter 2

Environmentally-Induced Thin Shell Deformation

2.1 Introduction

Consider the wrinkling and curling of a drying leaf. The drying process corresponds to water evaporating from the internal cells, so that the tissue contracts in volume. This process typically happens differentially, due to having one side exposed to the sun, or having boundaries farther away from the veins than interior, leading to non-uniform curvature and curling of the leaf [157, 63]. Related phenomena are the swelling and wrinkling of paper when exposed to water, such as when coffee is spilled on a notebook page, or the wrinkling and shrinking of plastic when heated. Although the physical mechanisms in these examples are different (shrinking of cells vs swelling of fibers vs contraction of polymers) the *effect* on the objects is the same: the intrinsic geometry of the thin objects change over time in response to dynamic changes in the environment.

In addition to growth, wrinkling, and swelling, other phenomena involving the same physics include burning of thin objects, especially those made of curved or composite materials; wilting of flowers and dynamic response of plants to light or humidity; changes and wrinkling in skin due to aging, moisture, creams; shrinking and subsequent wrinkling of clothes; warping of wood due to one-sided heating; etc. There is also increasing excitement about manufacturing processes based on differential or inhomogeneous growth, that require precise control over the material's rest geometry. Examples include water printers that induces paper bending [53], two-layer structures consisting of plastic printed on canvas under tension [112], the various works of Nervous Systems [121], and fabrication using networks of filaments that stretch anisotropically when moistened. Exploiting these technologies requires solving challenging inverse problems, with reliable methods for solving *forward* problems an essential first step.

Of course, one could capture all of these phenomena by modeling elastic volumes explicitly using tetrahedral or hexahedral elements, and tracking the change in moisture or heat within each element. However, such a volumetric model is computationally expensive, and unnecessary: structures that are thin relative to their surface diameter ought to be able to benefit from a reduced elastic shell model, augmented to track and account for dynamic changes in the environmental stimulus throughout the shell volume. Yet previous methods studying phenomena such as burning of paper [80], drying of leaves [63], and cooking pasta [150], have used either ad-hoc application-specific shell models, or volumetric finite elements, to account for intrinsic geometric changes. Our goal instead is to provide a principled low-order simulation methodology for such systems. This goal can be broken into two connected challenges: to simulate the object's elastic response to changes in its intrinsic geometry, and to model its intrinsic geometry changes in response to environmental stimuli. In this work we address both, resulting in a numerical method that allows researchers studying growth and related phenomena to plug in any realistic model and parameter set, and reap a working simulation.

Contribution We present a unified low-order discrete shell model tailored to simulating non-uniform, anisotropic, differential growing and shrinking of thin shells. This model is needed for simulating real-world thin materials whose geometry changes in response to stimuli such as heat, moisture, and growth. In contrast to previous methods for simulating such phenomena, our formulation builds on discrete geometric shell theory and supports arbitrary rest curvature and strain, and physical settings such as thickness and Lamé parameters. We couple our shell model to a simple formulation of moisture and temperature diffusion in both the lateral and thickness directions, which takes into account anisotropy of the material grain. In a series of experiments, we show that our model successfully predicts the qualitative behavior of thin shells undergoing complex, dynamic deformations due to material expansion or contraction, such as occurs when paper is moistened or thin plastic melts.

2.1.1 Related Work

Simulating Burning/Melting/Swelling Several papers look at related problems, such as evolving the boundary of a burning or melting solid, without incorporating curling/wrinkling and other elastic deformations of the solid. Melek and Keyser [88, 89] simulate pyrolysis and heat diffusion of burning objects, but do not consider their elastic deformation. Losasso et al [83] proposed tracking of the burning boundary of thin shells using an adaptive level set on the shell. Some of the deformation can be qualitatively approximated by mapping physical quantities like heat and moisture to cells of a coarse grid around the object, deforming the cage, and mapping the deformation back onto the shell (as in Free Form Deformation); this approach was proposed by Melek and Keyser [90] and adopted by Liu et al [80].

Steps towards a more principled elastic model include the use of a mass-spring network to represent the shell, with update rules for how spring rest lengths should change due to physical processes in the shell. Such rules are simplest to formulate in the case where growth or shrinkage is uniform through the shell thickness, and the shell can be represented using a single spring layer; Larboulette et al [73] present such a rule, which includes handling of the *machine direction* of paper: a bias in the orientation of the fibers composing the paper which causes the paper to swell anisotropically. We adopt this parameter in our material model.

Most similar to our work is the method of Jeong et al [62, 63], which uses a *bilayer* of springs (a triangle mesh and its circumcentric dual, offset a distance from the primal mesh) to represent the shell. The bilayer allows the method to capture *differential* growth due to gradients in moisture concentration across the thickness of leaves, leading to visually impressive simulations of leaves curling as they dry. Our work is based on the same fundamental idea (representing the shell using a rest strain that varies linearly through the thickness) but couched in the machinery of differential geometry; our formulation allows us to easily incorporate non-zero rest curvature, machine direction, and a physical material model. Also somewhat related are the *CurveUps* of Guseinov et al [54], which induce rest curvature in shells by embedding rigid pieces within a pre-stressed substrate.

Mechanics of Shells The mathematics underpinning the physics of thin shells is a venerable topic: Ciarlet's book [28] on elasticity as applied to shells offers a thorough overview. Our work is based on the common *Kirchhoff-Love* assumption that the shell does not undergo any transverse shear; i.e., that the shell volume is foliated by normal offsets of the shell's *midsurface*. The problem of studying deformation of the 3D shell volume then reduces to that of deformation of a 2D surface, and tools from Riemannian geometry can be applied [126].[1] One key property of the shells we want to simulate is that they are *non-Euclidean*: they do not have a rest (strain-free) state that is realizable in three-dimensional space. Non-Euclidean shells have received substantial attention recently in the physics community [70, 68], thanks to their potential applications in fabrication and robotics, and their connection to biological

[1] We adopt the so-called "intrinsic" view [101] that shells can be understood in terms of Kirchhoff-Love and geometric principles, as this view allows us to easily discretize shell physics by leveraging discrete differential geometry, but we note in passing that the validity of the Kirchhoff-Love assumption, and of reduced shell models in general, remains unsettled, and the literature documents numerous alternative shell theories.

growth; physicists such as Sharon, Efrati, and Ben Amar [49, 31, 35, 124] pioneered the study of shell mechanics in this setting.

For the sake of being self-contained, we briefly review the geometric foundations of shell mechanics in Section 2.2.

Computational Modeling of Thin Shells Thin shells first caught the interest of the graphics community in the context of simulating cloth [6, 20]. These early methods tended to focus on thin *plates*, i.e. shells that have a flat rest configuration, and formulate shell dynamics in terms of either hinge-based bending energies [134, 135] or the insight that the bending energy can be written in terms of the intrinsic Laplace-Beltrami operator applied to the shell's embedding function [15, 12, 152].

Grinspun et al [51] introduced to graphics the simulation of shells with non-zero rest curvature. Their formulation is based on *differences of squared mean curvature*, leading to a simple and easy-to-discretize bending energy. This model is physically suspect, however: consider a half-cylinder at rest when curled around the x-axis. Unbend the shell and re-bend it around the y-axis; the deformed configuration's strain cannot be captured by looking at mean curvature alone, as it is pointwise identical to the mean curvature of the rest configuration. Complete support for rest curvature therefore requires a bending energy that incorporates full information about the extrinsic deformation of the shell [50]. One such discrete energy is described in Weischedel's work on discrete Cosserat shells [154]; our exposition is modeled closely on hers, though we make different modeling choices (we use an intrinsic rather than Cosserat shell model, and require more flexible handling of the shell rest geometry).

A popular alternative to Grinspun et al.'s bending formulation based on the mid-edge shape operator is to use a triangle-averaged shape operator proposed by Gingold et al. [46], which expresses bending energy of a triangle in terms of the hinge angles of each of its three edges. Gingold et al. demonstrate how plasticity can be implemented by maintaining and dynamically updating "rest" values of the hinge angles, and also propose a simple method for simulating thin shell fracture. Although Grinspun et al. [50] raised concerns about the consistency of this averaged shape operator, it is simple to implement and has been adopted in thin shell simulation frameworks like ArcSim [99]. We briefly discuss how Gingold et al.'s operator might be adapted for non-Euclidean rest geometry in Section 2.6.

Higher-order methods for simulating shells (including with NURBS or subdivision elements) are common in computational mechanics and isogeometric analysis [9, 8, 29, 67, 11, 5] and have also been proposed for computer graphics [153] and growing shells [144]. High-order methods have some obvious advantages (better convergence behavior in the thin limit, continuous surface normals) at the cost of additional computation and complexity, especially when handling contact.

In this paper, we ignore the problem of mesh tessellation, or of adapting the mesh in response to either large deflections or large amounts of growth;

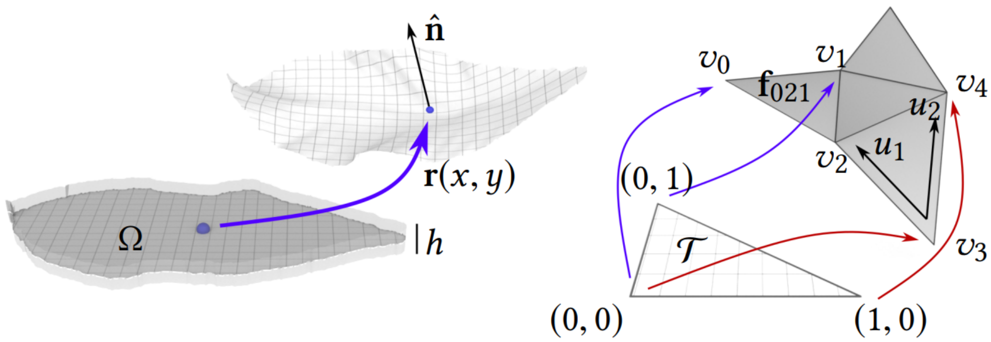

Figure 2.1: Left: the volumetric shell is parameterized by a slab $\Omega \times [-h/2, h/2]$ around a region Ω in the plane. $\mathbf{r}$ maps Ω to the shell midsurface. Right: we parameterize all triangles of discrete shells by a single canonical triangle $\mathcal{T}$. We express all face-based quantities in the face's local barycentric coordinate system (u_1, u_2), which is not consistent across faces.

such remeshing is an important component of a practical shell simulation but orthogonal to our focus on shell dynamics. An existing tool such as ArcSim [100], which incorporates a method of adaptive remeshing while avoiding significant popping artifacts, could easily be adopted in our framework if desired.

2.2 Continuous Formulation

Before describing our discretization of shells, we briefly review the formulation in the continuous setting, as this formulation will guide our discretization.

Shell Geometry We can represent shells $S \subset \mathbb{R}^3$ of thickness h by a parameter domain Ω in the plane and an embedding $\phi\colon \Omega \times [-h/2, h/2] \to \mathbb{R}^3$ with S the image of ϕ (see Figure 2.1). The Kirchhoff-Love assumption allows us to represent the entire shell volume only in terms of the shell's *midsurface* $\mathbf{r}\colon \Omega \to \mathbb{R}^3$. In other words,

$$\phi(x, y, z) = \mathbf{r}(x, y) + z\hat{\mathbf{n}}(x, y)$$

where $\hat{\mathbf{n}} = (\mathbf{r}_x \times \mathbf{r}_y)/\|\mathbf{r}_x \times \mathbf{r}_y\|$ is the midsurface normal. The metric $\mathbf{g}$ on the slab $\Omega \times [-h/2, h/2]$, pulled back from $\mathbb{R}^3$, can be expressed in terms of the geometry of the midsurface:

$$\mathbf{g} = \begin{bmatrix} \mathbf{a} - 2z\mathbf{b} + z^2\mathbf{c} & 0 \\ 0 & 1 \end{bmatrix}, \tag{2.1}$$

where

$$\mathbf{a} = d\mathbf{r}^T d\mathbf{r} \quad \mathbf{b} = -d\mathbf{r}^T d\hat{\mathbf{n}} \quad \mathbf{c} = d\hat{\mathbf{n}}^T d\hat{\mathbf{n}}$$

are the classical first, second, and third fundamental forms of the surface $\mathbf{r}$.

Oftentimes, the parameterization domain of a thin shell is assumed to be also the rest state of the shell, so that the strain in the material of the shell can be determined directly from looking at $\mathbf{g}$. We cannot assume this: consider for instance a piece of paper whose center has been moistened by spilled coffee. The fibers in the coffee stain stretch; since they are confined by the surrounding non-wet region of the paper, the paper cannot globally stretch in such a way that both the wet and dry regions of the paper are simultaneously at rest. Instead, the paper will *buckle* out of plane, into a shape that compromises between relaxing the in-plane (stretching) strain and the introduced bending strain. At this point the paper's rest state is *non-Euclidean*—it is impossible to find any embedding of the paper into $\mathbb{R}^3$ that is entirely strain-free.

We therefore record the rest state of the shell using a *rest metric* $\bar{\mathbf{g}}(x, y, z)$ [35].[2] Since our model is tailored to simulating differential in-plane swelling or shrinking across the thickness of the shell, we make the simplifying assumption that this rest metric is linear in the thickness direction:

$$\bar{\mathbf{g}}(x, y, z) = \begin{bmatrix} \bar{\mathbf{a}}(x, y) - 2z\bar{\mathbf{b}}(x, y) & 0 \\ 0 & 1 \end{bmatrix}.$$

A shell that begins a simulation at rest will simply have $\bar{\mathbf{a}} = \mathbf{a}^0$ and $\bar{\mathbf{b}} = \mathbf{b}^0$, where $\mathbf{a}^0$ and $\mathbf{b}^0$ are the values of $\mathbf{a}$ and $\mathbf{b}$ at the start of the simulation, respectively; this setup is a special case of a shell which has a rest state specified by a "rest surface" $\bar{\mathbf{r}}$ that is isometrically embeddable in $\mathbb{R}^3$, in which case $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ are the first and second fundamental forms of that rest surface. Therefore $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ can be thought of as representing the "rest metric" and "rest curvature" of the shell's midsurface, respectively.[3]

To summarize, our parameterization of thin shells involves the following kinematic elements:

- a thickness h and parameterization domain $\Omega \subset \mathbb{R}^2$, both of which are fixed over the course of the simulation;

[2]Here and throughout the paper, we use an overbar to denote quantities associated to the shell rest state.

[3]We stress, though, that these labels are to provide intuition only—$\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ must not, and generally will not, satisfy usual relationships from differential geometry such as the Gauss-Codazzi-Mainardi equations.

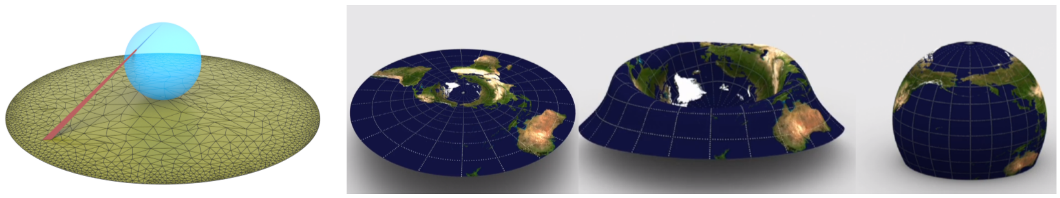

Figure 2.2: Stereographic projection of the disk into the plane yields a conformal parameterization of one by the other (left); we simulate the disk dynamics as it relaxes to equilibrium by adopting a spherical shape (right).

- an embedding $\mathbf{r} : \Omega \to \mathbb{R}^3$ representing the shell midsurface's "current" or "deformed" geometry, and which evolves over time. From this embedding, the current midsurface normals $\hat{\mathbf{n}}$ can be calculated, and thus $\mathbf{r}$ provides the embedding of the full shell volume ϕ, as well as the midsurface fundamental forms;

- a rest metric $\bar{\mathbf{g}}$, parameterized by the pair of tensor fields $\bar{\mathbf{a}}, \bar{\mathbf{b}}$ over Ω, respectively. These might also evolve over time, due to changes in the shell rest state via expansion or contraction.

2.2.1 Shell Dynamics

Motivated by the common observation that a sufficiently thin shell bends much more readily than it will stretch, we assume that the shell's deformation involves *large rotations* but only small in-plane strain of the midsurface: $\|\bar{\mathbf{a}}^{-1}\mathbf{a} - \mathbf{I}\|_\infty < h$. We also assume that the shell's material is uniform and isotropic. The simplest constitutive law consistent with these assumptions is the St. Venant-Kirchhoff material model[4] together with Green strain; it can be shown (see e.g. Weischedel [154]) that these choices yield an elastic energy density (the *Koiter shell model*) that can be approximated up to $O(h^4)$ by

$$W(x,y) = \left(\frac{h}{4}\|\bar{\mathbf{a}}^{-1}\mathbf{a} - \mathbf{I}\|_{SV}^2 + \frac{h^3}{12}\|\bar{\mathbf{a}}^{-1}(\mathbf{b} - \bar{\mathbf{b}})\|_{SV}^2\right)\sqrt{\det \bar{\mathbf{a}}} \qquad (2.2)$$

where $\|\|_{SV}$ is the "St. Venant-Kirchhoff norm" [154]

$$\|M\|_{SV} = \frac{\alpha}{2}\operatorname{tr}^2 M + \beta \operatorname{tr}\left(M^2\right),$$

for material parameters α, β. In terms of the Young's modulus E and Poisson's ratio ν,

$$\alpha = \frac{E\nu}{1-\nu^2}, \quad \beta = \frac{E}{2(1+\nu)}.$$

We thus have a formulation of kinetic energy and potential energy

$$T[\dot{\mathbf{r}}] = \int_\Omega h\rho\|\dot{\mathbf{r}}\|^2\sqrt{\det \bar{\mathbf{a}}}\, dxdy, \quad V[\mathbf{r}] = \int_\Omega W(x,y)\, dxdy,$$

for volumetric density ρ, to which additional external energies and forces (gravity, constraint forces, etc) can be added to yield equations of motion via the usual principle of least action.

2.3 Discretization

We approximate the midsurface $\mathbf{r}$ with a triangle mesh (V, E, F); the positions of the vertices $\mathbf{v} = [\mathbf{v}_1, \mathbf{v}_2 \ldots]$ take the place of the embedding function $\mathbf{r}$. The general strategy we will use is to assume that $\mathbf{a}$ and $\mathbf{b}$, as well

[4] The neo-Hookean material model is also popular in computer graphics and could be used instead, although there is little benefit to doing so when simulating thin shells since strains are typically small.

as their rest counterparts $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$, are constant over each face of the triangle mesh; it will then be straightforward to write down a discrete analogue of the Koiter elastic energy density in Equation (2.2).

Discrete Shell Model As in the continuous setting, the discrete shell does not necessarily have a rest state embeddable as a mesh in $\mathbb{R}^3$, making it impossible to parameterize the deformed configuration of the shell in terms of the rest configuration; additionally we do not want to assume (or compute) a global parameterization of the midsurface. Instead, we independently parameterize each triangle in its own barycentric coordinates (see Figure 2.1). Let $\mathbf{f}_{ijk}$ be a face in F containing the vertices $\mathbf{v}_i$, $\mathbf{v}_j$, $\mathbf{v}_k$, and denote by $\mathcal{T}$ the canonical unit triangle with vertices $(0,0)$, $(1,0)$, $(0,1)$. Then locally the face $\mathbf{f}_{ijk}$ is embedded by the affine function

$$\mathbf{r}_{ijk} : \mathcal{T} \to \mathbb{R}^3, \quad \mathbf{r}_{ijk}(u_1, u_2) = \mathbf{v}_i + u_1(\mathbf{v}_j - \mathbf{v}_i) + u_2(\mathbf{v}_k - \mathbf{v}_i);$$

under this embedding, the Euclidean metric on face $\mathbf{f}_{ijk}$ pulls back to the first fundamental form

$$\mathbf{a}_{ijk} = \begin{bmatrix} \|\mathbf{v}_j - \mathbf{v}_i\|^2 & (\mathbf{v}_j - \mathbf{v}_i) \cdot (\mathbf{v}_k - \mathbf{v}_i) \\ (\mathbf{v}_j - \mathbf{v}_i) \cdot (\mathbf{v}_k - \mathbf{v}_i) & \|\mathbf{v}_k - \mathbf{v}_i\|^2 \end{bmatrix}$$

on $\mathcal{T}$. If we are given an explicit rest configuration $\bar{\mathbf{v}}_i$ of the shell, we can compute the rest first fundamental form $\bar{\mathbf{a}}_{ijk}$ analogously; or alternatively we can set $\bar{\mathbf{a}}_{ijk}$ to any desired symmetric positive-definite 2×2 matrix. Notice that any such matrix corresponds to some choice of "rest lengths" for the edges of face $\mathbf{f}_{ijk}$ that obey the triangle inequality, but that two faces sharing a common edge do not necessarily agree about that length.

These discrete fundamental forms are enough to discretize the stretching term in Equation (2.2): each face contributes a term

$$\int_{\mathcal{T}} \frac{h}{4}\|\bar{\mathbf{a}}_{ijk}^{-1}\mathbf{a}_{ijk} - \mathbf{I}\|_{SV}^2 \sqrt{\det \bar{\mathbf{a}}_{ijk}} = \frac{h}{8}\|\bar{\mathbf{a}}_{ijk}^{-1}\mathbf{a}_{ijk} - \mathbf{I}\|_{SV}^2 \sqrt{\det \bar{\mathbf{a}}_{ijk}},$$

where the division by two is due to the canonical triangle $\mathcal{T}$ having area $\frac{1}{2}$. This energy is quartic in the positions of the mesh vertices, and is exactly the energy of constant-strain triangle stretching elements commonly used in cloth simulation.

For the bending term, we also need a discretization of the second fundamental form. Here there is a significant departure between the smooth theory and the discrete approximation: we would like to apply the Kirchhoff-Love principle to extrude the mid-surface into a shell volume, but unfortunately normal offsets of triangle meshes are no longer guaranteed to be triangle meshes (or even piecewise-affine). One can instead look at weaker notions of mesh parallellity [14]:

- *vertex offsets* require choosing a normal at each mesh vertex (itself a problem without an obvious solution), and moving each vertex a constant distance along this normal usually does not result in faces parallel to the original faces;

- *edge offsets* likewise do not guarantee parallel faces;

- *face offsets* are not conforming: moving each of the faces neighboring a vertex of valence four or higher in their normal directions yields new faces that are not guaranteed to still intersect at a common point.

While there is no perfect choice, we use the discretization that arises from edge parallelity, leading to the so-called "mid-edge" discretization of the second fundamental form [50, 154]. This approach has several advantages: first, computing the edge offsets of a face requires knowing only the geometry of that face and its three edge neighbors, leading to a compact and constant-size discrete stencil for computing the discrete second fundamental form. By contrast, vertex offsets lead to stencils that vary depending on vertex valence. Moreover, the mid-edge formulation is significantly more robust to triangle inversion artifacts. Unlike in a face-offset-based approach, it also allows us to discretize rest second fundamental forms in the same place as the first fundamental forms, on the mesh faces.

Let $\mathbf{e}_i$ denote the edge opposite vertex i on face $\mathbf{f}_{ijk}$, and define the *mid-edge normal* $\hat{\mathbf{n}}_i$ by:

- the face normal $\frac{(\mathbf{v}_j-\mathbf{v}_i)\times(\mathbf{v}_k-\mathbf{v}_i)}{\|(\mathbf{v}_j-\mathbf{v}_i)\times(\mathbf{v}_k-\mathbf{v}_i)\|}$, if $\mathbf{e}_i$ is a boundary edge;
- the mean of the face normals of the two faces incident on $\mathbf{e}_i$, otherwise.

Let $\mathbf{f}^{\epsilon}_{ijk}$ denote the triangle formed by offsetting all of $\mathbf{f}_{ijk}$'s edges in their mid-edge normal direction by a distance ϵ, and let $\mathbf{a}^{\epsilon}_{ijk}$ be the discrete first fundamental form of that offset triangle. Then the discrete second fundamental form $\mathbf{b}$ can be defined, by analogy to Equation (2.1), as the first-order correction $\mathbf{a}^{\epsilon}_{ijk} = \mathbf{a}_{ijk} - 2\epsilon\mathbf{b}_{ijk} + O(\epsilon^2)$, leading to the formula

$$\mathbf{b}_{ijk} = \frac{1}{2}\begin{bmatrix} (\hat{\mathbf{n}}_i - \hat{\mathbf{n}}_j)\cdot(\mathbf{v}_i - \mathbf{v}_j) & (\hat{\mathbf{n}}_i - \hat{\mathbf{n}}_j)\cdot(\mathbf{v}_i - \mathbf{v}_k) \\ (\hat{\mathbf{n}}_i - \hat{\mathbf{n}}_k)\cdot(\mathbf{v}_i - \mathbf{v}_j) & (\hat{\mathbf{n}}_i - \hat{\mathbf{n}}_k)\cdot(\mathbf{v}_i - \mathbf{v}_k) \end{bmatrix}.$$

(Alternatively, this formula can be derived by discretizing the relation $\mathbf{b} = -d\mathbf{r}^T d\hat{\mathbf{n}}$ using divided differences). Although it may not appear so at first, the matrix $\mathbf{b}_{ijk}$ is always symmetric (since each mid-edge normal is orthogonal to that edge); it is not in general positive-definite.[5] We represent the rest second fundamental form $\bar{\mathbf{b}}_{ijk}$ by an arbitrary symmetric 2×2 matrix assigned to each face.

Choosing Rest Fundamental Forms Depending on the mechanism for growth being simulated, there are several choices for how to set and update $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$:

No Growth: A classic shell, whose rest state is fixed, simply has $\bar{\mathbf{a}} = \mathbf{a}^0$ and $\bar{\mathbf{b}} = \mathbf{b}^0$, where $\mathbf{a}^0$ denotes the first fundamental form induced by $\mathbf{v}^0$, the positions of the midsurface vertices at the beginning of the simulation. (And if the shell is pre-strained, $\bar{\mathbf{a}}, \bar{\mathbf{b}}$ can be adjusted appropriately).

[5] As observed by Grinspun et al [50], the *shape operator* $d\hat{\mathbf{n}}$ in the continuous setting always maps tangent vectors to tangent vectors, whereas in the discrete setting the finite difference of mid-edge normals is not always parallel to the mesh triangle. This discrepancy is a consequence of the failure of edge-offset meshes to also be face-offsets. Corrections to the shape operator have been proposed to remedy this quirk, though we found them unnecessary (and in any case, any components of $d\hat{\mathbf{n}}$ that lie orthogonal to the face are annihilated when forming the second fundamental form $-d\mathbf{r}^T d\hat{\mathbf{n}}$).

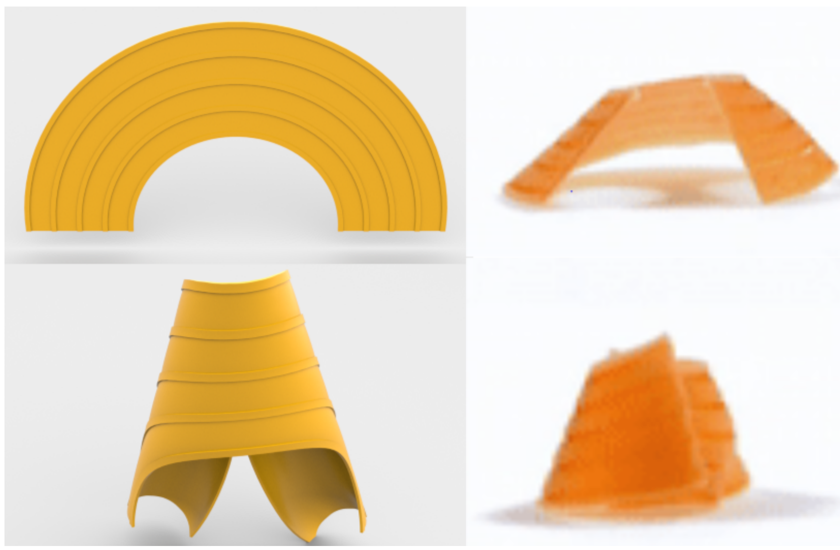

Figure 2.3: We simulate one of Wang et al.'s [150] shape-changing pasta designs, consisting of a half-annulus decorated with concentric rings which are both thicker (and thus more bending-resistant) and less porous than the surrounding material. Our simulation predicts qualitatively identical curling behavior of the pasta as both the physical experiment and volumetric FEM simulation conducted by Wang et al. Top-left: the initial geometry, showing the thickened concentric ribs. Bottom-left: our simulated result. Right: photographs of Wang et al's experimental results, at different stages of swelling. Photographs reproduced with permission.

Pullback Forms: In the case where the shell's initial configuration is flat, it is natural to align it with a region of the xy plane, and prescribe rest fundamental forms in Euclidean (x, y) coordinates, instead of prescribing an $\bar{\mathbf{a}}_{ijk}$ in the barycentric coordinates of each triangle. Let $\bar{\mathbf{a}}^{xy}$ and $\bar{\mathbf{b}}^{xy}$ be such prescribed forms; these can be sampled on each triangle $\mathbf{f}_{ijk}$'s centroid and pulled back to the triangle's parameterization domain to give

$$\bar{\mathbf{a}} = T^T \bar{\mathbf{a}}^{xy}(\xi) T, \qquad \bar{\mathbf{b}} = T^T \bar{\mathbf{b}}^{xy}(\xi) T$$

in barycentric coordinates, where $T = \begin{bmatrix} \mathbf{v}_j^0 - \mathbf{v}_i^0 & \mathbf{v}_k^0 - \mathbf{v}_i^0 \end{bmatrix}$ maps from vectors in the barycentric coordinates of $\mathbf{f}_{ijk}$ to Euclidean space, and $\xi = \frac{1}{3}(\mathbf{v}_i + \mathbf{v}_j + \mathbf{v}_k)$ is the face centroid.

Isotropic Growth: In many cases growth is isotropic and uniform through the thickness of the shell (for instance, when plastic shrinks in response to heat, or biological tissue grows through cell division). In this case $\bar{\mathbf{a}} = e^{2s_{ijk}}\mathbf{a}^0, \bar{\mathbf{b}} = \mathbf{b}^0$ for a per-face conformal factor s encoding the amount of growth (or shrinking, if negative).

Linear Differential Swelling: Porous materials like paper swell when moistened, and differences in water concentration through the thickness of a thin shell can induce metric frustration and buckling. This mechanism is responsible for the buckling of paper when wet, and the curling of leaves as they dry.

We model this swelling mechanism by assuming that the amount of moisture varies linearly in the thickness direction of the shell, and represent the percentage of additional moisture present in the material at the top and bottom of the shell by two scalars $m_i^+, m_i^- \in \mathbb{R}$ at each vertex $\mathbf{v}_i$. We average these values to compute a moisture content $m_{ijk}^{\pm}$ per face $\mathbf{f}_{ijk}$. The additional water content induces swelling, which changes the rest geometry; assuming a linear relationship between rest length and moisture concentration [1], we can write the rest metric of the volumetric shell as

$$\bar{\mathbf{g}} = \begin{bmatrix} \bar{\mathbf{g}}^- \left(\frac{1}{2} - \frac{z}{h}\right) + \bar{\mathbf{g}}^+ \left(\frac{1}{2} + \frac{z}{h}\right) & 0 \\ 0 & 1 \end{bmatrix}.$$

Here $\bar{\mathbf{g}}^+$ and $\bar{\mathbf{g}}^-$ are the metric on the top and bottom of the shell,

$$\bar{\mathbf{g}}^+ = (1 + m^+\mu)^2(\mathbf{a}^0 - h\mathbf{b}^0); \quad \bar{\mathbf{g}}^- = (1 + m^-\mu)^2(\mathbf{a}^0 + h\mathbf{b}^0)$$

for moisture expansion coefficient μ. Then

$$\begin{aligned} \bar{\mathbf{a}} &= \frac{(1+m^+\mu)^2 + (1+m^-\mu)^2}{2}\mathbf{a}^0 + h\frac{(1+m^-\mu)^2 - (1+m^+\mu)^2}{2}\mathbf{b}^0 \\ \bar{\mathbf{b}} &= \frac{(1+m^-\mu)^2 - (1+m^+\mu)^2}{2h}\mathbf{a}^0 + \frac{(1+m^+\mu)^2 + (1+m^-\mu)^2}{2}\mathbf{b}^0. \end{aligned}$$

Piecewise Constant Differential Swelling Instead of a linear moisture gradient through the thickness, in some cases it is more appropriate to model a piecewise constant moisture profile, such as when modeling bilayers with different material properties. For example, Wang et al [150] fabricate exotic pasta geometries by cooking pasta composed of two layers of different porosity. van Rees et al [143] showed that a bilayer of thickness h with piecewise-constant rest metric

$$\bar{\mathbf{g}}(z) = \begin{bmatrix} \begin{cases} \bar{\mathbf{g}}^+, z > 0 \\ \bar{\mathbf{g}}^-, z < 0 \end{cases} & 0 \\ 0 & 1 \end{bmatrix}$$

is *energetically equivalent* to a shell with linearly-varying metric

$$\bar{\mathbf{a}} = \frac{\mathbf{g}^+ + \mathbf{g}^-}{2}, \qquad \bar{\mathbf{b}} = \frac{3}{4h}(\mathbf{g}^- - \mathbf{g}^+);$$

thus the desired piecewise-constant metric taking into account moisture-induced swelling is

$$\bar{\mathbf{g}}^+ = (1 + m^+\mu^+)^2 \left(\mathbf{a}^0 - \frac{2}{3}h\mathbf{b}^0\right); \quad \bar{\mathbf{g}}^- = (1 + m^-\mu^-)^2 \left(\mathbf{a}^0 + \frac{2}{3}h\mathbf{b}^0\right),$$

where μ^+ and μ^- encode the differing moisture-length relationship in the two layers. Converting this piecewise-constant metric back into the equivalent linear metric gives

$$\begin{aligned}
\bar{\mathbf{a}} &= \frac{(1 + m^+\mu^+)^2 + (1 + m^-\mu^-)^2}{2}\mathbf{a}^0 + h\frac{(1 + m^-\mu^-)^2 - (1 + m^+\mu^+)^2}{3}\mathbf{b}^0 \\
\bar{\mathbf{b}} &= 3\frac{(1 + m^-\mu^-)^2 - (1 + m^+\mu^+)^2}{4h}\mathbf{a}^0 + \frac{(1 + m^+\mu^+)^2 + (1 + m^-\mu^-)^2}{2}\mathbf{b}^0.
\end{aligned}$$

Linear Differential Swelling with Machine Direction In paper, leaves, and other materials composed of microscopic fibers, swelling induced by moisture is *anisotropic*, since fibers swell more in their circumferential than axial direction. We can model this behavior by storing a *machine direction* $\mathbf{d}_{ijk}$ per triangle face; this direction, a vector in the barycentric coordinates of the triangle, is the direction in which the fibers are aligned. Given this direction, we can compute the intrinsically orthogonal direction $\mathbf{d}^{\perp}_{ijk}$ (with $\mathbf{d}^T_{ijk}\mathbf{a}^0_{ijk}\mathbf{d}^{\perp}_{ijk} = 0$), and impose different moisture-length constants μ and $\mu_\perp$ in the $\mathbf{d}$ and $\mathbf{d}^\perp$ directions, respectively. Then the desired rest metrics at the top and bottom of the shell are

$$\bar{\mathbf{g}}^{+} = T^T M^{+} T^{-T}(\mathbf{a}^0 - h\mathbf{b}^0)T^{-1}M^{+}T,$$
$$\bar{\mathbf{g}}^{-} = T^T M^{-} T^{-T}(\mathbf{a}^0 + h\mathbf{b}^0)T^{-1}M^{-}T.$$

where $T = \begin{bmatrix} \mathbf{d} & \mathbf{d}^{\perp} \end{bmatrix}^{-1}$ transforms from the triangle's barycentric coordinates to the $\mathbf{d}, \mathbf{d}^{\perp}$ coordinate system, and

$$M^{\pm} = \begin{bmatrix} (1 + m^{\pm}\mu) & 0 \\ 0 & (1 + m^{\pm}\mu_{\perp}) \end{bmatrix}$$

anisotropically stretches in the machine direction. As in the previous cases, the rest fundamental forms can be computed from these metrics using the formula

$$\bar{\mathbf{a}} = \frac{\bar{\mathbf{g}}^{+} + \bar{\mathbf{g}}^{-}}{2}, \qquad \bar{\mathbf{b}} = \frac{\bar{\mathbf{g}}^{-} - \bar{\mathbf{g}}^{+}}{2h}.$$

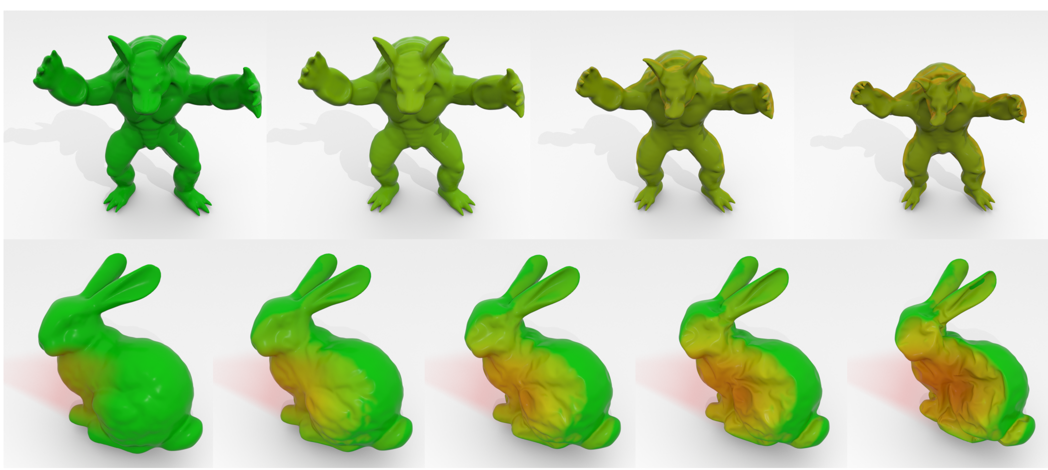

Figure 2.4: Time evolution of a plastic armadillo and a plastic bunny as they shrink when exposed to localized heating (red beam in the figure), where redder parts have higher temperature.

Elastic Energy We can now write down the full elastic energy of the shell, in exact analogy to the Koiter energy:

$$E_{\text{elastic}}(\mathbf{v}) = \sum_{\mathbf{f}_{ijk}\in F} \frac{\sqrt{\det \bar{\mathbf{a}}_{ijk}}}{2}\left(\frac{h}{4}\|\bar{\mathbf{a}}_{ijk}^{-1}\mathbf{a}_{ijk} - \mathbf{I}\|_{SV}^2 + \frac{h^3}{12}\|\bar{\mathbf{a}}_{ijk}^{-1}(\mathbf{b}_{ijk} - \bar{\mathbf{b}}_{ijk})\|_{SV}^2\right).$$

It is worth making a few observations about this energy. First, the matrices $\mathbf{a}$, $\bar{\mathbf{a}}$, etc. are *coordinate-dependent*: replacing the parameterization domain $\mathcal{T}$, or even cyclically permuting the order of vertices around a face, would alter the values in the matrix. However, the generalized eigenvalues of $\mathbf{a} - \bar{\mathbf{a}}$ and $\mathbf{b} - \bar{\mathbf{b}}$ with respect to the inner product $\bar{\mathbf{a}}$ are coordinate-*independent*, as is the total energy. Perhaps the easiest way to see this fact is to note that these spectra measure the geometrically exact strain induced by an affine embedding of $\mathcal{T}$, and so *must* be independent of the coordinates chosen. Second, the terms of the form $\|\bar{\mathbf{a}}^{-1}M\|_{SV}^2$ are sometimes instead written as $\|\bar{\mathbf{a}}^{-1/2}M\bar{\mathbf{a}}^{-1/2}\|_{SV}^2$, where $\bar{\mathbf{a}}^{-1/2}$ is the unique positive-definite square root of $\bar{\mathbf{a}}$. The two expressions are equivalent, since the spectrum of a product of matrices is invariant under cyclic permutation, but the form used above is slightly more convenient for computation.

Mass Matrix Whether swelling or shrinking of the surface affects the mass of the surface depends on the mechanism: changes due to growth or moisture absorption/evaporation do change the mass, while plastic polymers contracting when exposed to heat do not. In cases where modeling the mass change is desired, the mass λ_i of each vertex can be recomputed at a given instant in time by

$$\lambda_i = \sum_{\mathbf{f}\sim\mathbf{v}_i} \frac{\rho h}{3}\sqrt{\det \bar{\mathbf{a}}_{\mathbf{f}}}/2;$$

here the sum is over all faces $\mathbf{f}$ incident to $\mathbf{v}_i$ and yields the usual "lumped" or barycentric mass matrix Λ.[6]

Viscous Damping Since the growth and swelling phenomena we want to simulate all take place at relatively long time scales, and paper and plastic are viscoelastic, a damping model is needed to dissipate the elastic waves in the material. We implement a simple Kelvin-Voigt damping model by including a damping potential

$$E_{\text{damp}}(\mathbf{v}, \mathbf{v}^{\text{prev}}) = \frac{\eta}{E}\Delta t \sum_{\mathbf{f}_{ijk}\in F} \frac{\sqrt{\det \bar{\mathbf{a}}_{ijk}}}{2}\left(W_{\text{s}} + W_{\text{b}}\right)$$

$$W_{\text{s}} = \frac{h}{4}\left\| \left[\mathbf{a}_{ijk}^{\text{prev}}\right]^{-1} \frac{\mathbf{a}_{ijk} - \mathbf{a}_{ijk}^{\text{prev}}}{\Delta t} \right\|_{SV}^{2}, \quad W_{\text{b}} = \frac{h^3}{12}\left\| \left[\mathbf{a}_{ijk}^{\text{prev}}\right]^{-1} \frac{\mathbf{b}_{ijk} - \mathbf{b}_{ijk}^{\text{prev}}}{\Delta t} \right\|_{SV}^{2}.$$

where Δt is the time step size, $\mathbf{v}^{\text{prev}}$ denotes the values of $\mathbf{v}$ in the previous time step (and likewise for $\mathbf{a}^{\text{prev}}$, etc), and η is a viscosity parameter.

Thickness	0.1 mm	Viscosity η	$5 \cdot 10^{-13}$ Pa $\cdot$ s
Young's Mod. E	2×10^9 Pa	Swelling const. μ	0.0025
Poisson's Ratio ν	0.3	Swelling const. $\mu^{\perp}$	0.001
Density ρ	250 kg/m^3		

Table 2.1: Table of reasonable physical parameters for ordinary paper.

[6] For absorption/evaporation, one might also want to model the fact that water has a different density than the shell material; we do not do so in our examples.

Summary In the simulation we track the following variables:

- The configuration $\mathbf{v}$ and configurational velocity $\dot{\mathbf{v}}$. These vertex positions completely encode the kinematics of the discrete shell.

- Two matrices $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ per face in F, both symmetric, and with $\bar{\mathbf{a}}$ positive-definite. These matrices store information about the rest state of the discrete shell, and may change over the course of the simulation. In most of our simulations, the mechanism for changes in the shell rest state is either change in temperature or absorption/evaporation of moisture; in this case we store two scalars m^+ and m^- per vertex, indicating temperature/moisture concentration on the top and bottom boundary of the shell, and $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ are computed from these scalars, as described above.

In addition, we track a machine direction $\mathbf{d}$ per face, which stays constant over the course of the simulation; finally Table 2.1 lists the physical parameters and constants on which the simulation depends, as well as reasonable values of these parameters for the special case of ordinary paper. We use these default values in all experiments described below, unless specified otherwise.

Time Integration We integrate the equations of motion using implicit Euler time integration:

$$\Lambda \frac{\dot{\mathbf{v}}^{i+1} - \dot{\mathbf{v}}^{i}}{\Delta t} = \mathbf{F}\left(\mathbf{v}^{i+1}, \mathbf{v}^{i}\right) \qquad \mathbf{v}^{i+1} = \mathbf{v}^{i} + \Delta t \dot{\mathbf{v}}^{i+1}, \tag{2.3}$$

where superscripts denote the time step index, and the total force is given by

$$\mathbf{F}\left(\mathbf{v}^{i+1}, \mathbf{v}^{i}\right) = \mathbf{F}_{\text{ext}} - \nabla E_{\text{elastic}}\left(\mathbf{v}^{i+1}\right) - \nabla E_{\text{damp}}\left(\mathbf{v}^{i+1}, \mathbf{v}^{i}\right)$$

where $\mathbf{F}_{\text{ext}}$ encapsulates contact forces and external forces like gravity. Solving these equations requires computing first and second derivatives of the elastic energy; the derivatives of a triangle's stretching term depend only on the vertices of that triangle, whereas the derivatives of the bending term also depend on vertices of the neighboring three triangles (due to the dependence of $\mathbf{b}$ on the mid-edge normals). The bending term in particular is somewhat unpleasant to differentiate due to its high degree of nonlinearity, and special cases that arise for triangles adjacent to the mesh boundary. We provide source code for calculating the derivatives on the project webpage.

2.4 Moisture Diffusion

Moisture diffuses in both the thickness and in-plane directions of the shell, and from the environment into the shell. We assume that within the shell, moisture diffuses isotropically at a rate uniform throughout the shell, so that the percentage of additional moisture $m(x, y, z; t) : \Omega \times [-h/2, h/2] \times \mathbb{R} \to \mathbb{R}$ obeys the diffusion equation

$$\frac{\partial m}{\partial t}(x, y, z) = \begin{cases} D\Delta_{\mathbf{g}} m, & -h/2 < z < h/2 \\ s(x, y, z), & z = \pm h/2, \end{cases} \tag{2.4}$$

where D is the diffusion coefficient, $\Delta_{\mathbf{g}}$ is the intrinsic Laplace-Beltrami operator with respect to the volumetric metric $\mathbf{g}$, and s is a source term describing diffusion into (or out) of the shell from the environment.

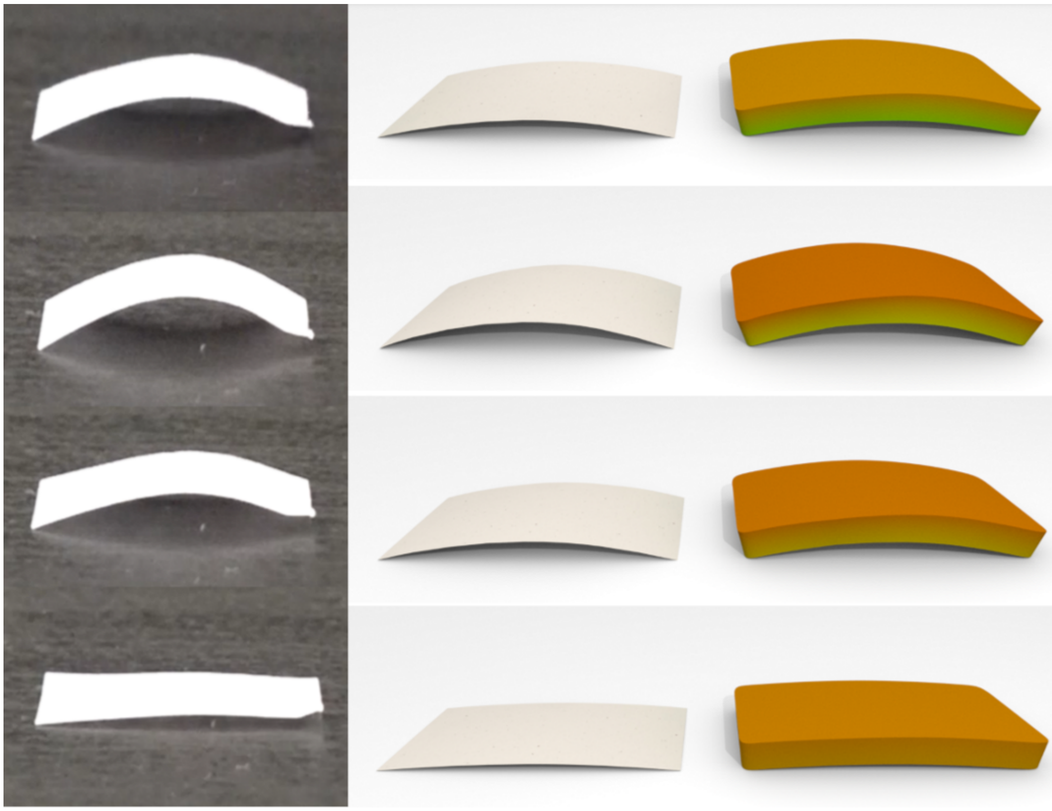

Figure 2.5: A comparison between the experiment (left), simulation (middle), and the thickened cross-section (right) as time progresses for a water-painted paper strip. Orange and green indicate high and low moisture content, respectively.

We discretize equation (2.4) with bilinear Galerkin finite elements on the triangular prisms $F \times [-h/2, h/2]$; the solution m and source term s in the prism surrounding face $\mathbf{f}_{ijk}$ are approximated by linear combinations of basis functions

$$m(u_1,u_2,z) = \sum_{v\in\{i,j,k\}} \left(m_v^+\psi_v^+ + m_v^-\psi_v^-\right)$$
$$s(u_1,u_2,z) = \sum_{v\in\{i,j,k\}} \left(s_v^+\psi_v^+ + s_v^-\psi_v^-\right)$$

parameterized over the prism $\mathcal{T}\times[-h/2,h/2]$ surrounding the canonical unit triangle. In other words

$$\psi_i^\pm = \frac{(1-u_1-u_2)}{h}\left(\frac{h}{2}\pm z\right); \quad \psi_j^\pm = \frac{u_1}{h}\left(\frac{h}{2}\pm z\right); \quad \psi_k^\pm = \frac{u_2}{h}\left(\frac{h}{2}\pm z\right).$$

From the above equations, the moisture is updated each time step using implicit time integration,

$$(M_G + \Delta t D K_G)\mathbf{m}^{i+1} = M_G(\mathbf{m}^i + \Delta t\mathbf{s}^i) \tag{2.5}$$

where $\mathbf{s}^i$ is the discretized source term. This source term $\mathbf{s}_i^\pm$ prescribes at each vertex the rate of diffusion of moisture in or out of the shell at both the top and bottom layer of the shell; the details of this term depend on the problem being modeled.

When the mechanism for swelling/shrinking is heat rather than moisture, the above diffusion formulation remains unchanged, with m reinterpreted as temperature rather than moisture.

2.5 Numerical Issues

Integrating the physics (2.3) requires some care, as the scale-separation between the stretching and bending forces, along with high values of stiffness, pose numerical challenges. (The diffusion equation in Equation (2.5) poses no numerical difficulties.)

Newton's Method We recommend the usual technique of using the explicit Euler step $\tilde{\mathbf{v}} = \mathbf{v}^i + \Delta t \dot{\mathbf{v}}^i$ as the initial guess in Newton's method each iteration; writing $\mathbf{v}^{i+1} = \tilde{\mathbf{v}} + \Delta t \delta \mathbf{v}$, we have from Equation 2.3

$$\Lambda \delta \mathbf{v} - \Delta t \mathbf{F} \left(\tilde{\mathbf{v}} + \Delta t \delta \mathbf{v}, \mathbf{v}^i \right) = 0 = \sigma(\delta \mathbf{v}),$$

which is well-scaled for using Newton's method to solve $\sigma(\delta \mathbf{v}) = 0$. The stiffness and nonlinearity of the elastic forces prohibit very large time steps, though a line search allows time integration using $\Delta t \approx 10^{-4}$ seconds.

The Newton gradient $\nabla \sigma = \Lambda + \Delta t^2 \nabla \mathbf{F}$ is symmetric and almost always positive-definite, and thus amenable to sparse Cholesky decomposition for small-to-medium sized meshes. We use the CHOLMOD solver of the SuiteSparse library [24] for solving the Newton linear system. In the (rare) cases where $\nabla \sigma$ is detected by the solver to be indefinite, we regularize by using $\nabla \sigma = (1 + \alpha)\Lambda + \Delta t^2 \nabla F$ for progressively larger multiples of α, until the Cholesky decomposition succeeds. For larger meshes, where the Cholesky factors do not fit in main memory, we solve the equations using Alglib's implementation of LBFGS [16]. Iterative methods (conjugate gradients) could also be used.

Inexact Hessian The Hessian of the bending energy is expensive to compute, and almost always dominated by the Hessian of the (much stiffer) stretching term; we observed improved performance replacing the exact bending Hessian with a partial approximation. In particular, we can rewrite the bending energy on face $\mathbf{f}_{ijk}$ as

$$E_{\text{bending}}(\mathbf{v}) = \frac{h^3\sqrt{\det \bar{\mathbf{a}}_{ijk}}}{24}\left(r_1(\mathbf{v})^2 + r_2(\mathbf{v})^2\right)$$

$$r_1(\mathbf{v}) = \sqrt{\alpha/2}\,\operatorname{tr}\left(\bar{\mathbf{a}}_{ijk}^{-1}(\mathbf{b}-\bar{\mathbf{b}}_{ijk})\right); \quad r_2(\mathbf{v}) = \sqrt{\beta\,\operatorname{tr}\left(\left[\bar{\mathbf{a}}_{ijk}^{-1}(\mathbf{b}-\bar{\mathbf{b}}_{ijk})\right]^2\right)}$$

(note that the trace of the square of any real matrix is guaranteed to be nonnegative). This allows approximation of the Hessian by the Gauss-Newton-esque

$$HE_{\text{bending}} \approx \frac{h^3\sqrt{\det \bar{\mathbf{a}}_{ijk}}}{12}\left(\nabla r_1 \nabla r_1^T + \nabla r_2 \nabla r_2^T\right).$$

Inverted Triangles We observed a somewhat subtle failure case when running simulations containing neighboring triangles with very different prescribed metrics $\bar{\mathbf{a}}$: if the two triangles are exactly coplanar, and one triangle collapses completely and inverts, then the mid-edge normal on the common edge is undefined (as its direction is now the mean of two anti-parallel vectors). This failure case can be prevented by either maintaining a minimum vertex/edge/face distance using continuous-time collision detection, by adaptively remeshing triangles undergoing excessive deformation [100], or by adjusting the initial mesh. The usual advice of using an intrinsic Delaunay triangulation, and avoiding adjacent triangles of very disparate size, applies.

Symmetry-Breaking Any rest flat ($\bar{\mathbf{b}} = 0$) shell has an extrinsically flat equilibrium configuration, regardless of $\bar{\mathbf{a}}$. In most cases this equilibrium state is unstable and is not observed in the real world, due to small imperfections in the shell material breaking the symmetry of the initial and/or rest state. We apply a small random perturbation to the rest and initial configurations of all of our initially-flat examples with $\bar{\mathbf{b}} = 0$, to force symmetry-breaking.

2.6 Hinge-Based Shape Operator

Gingold et al. [46] use a hinge-based shape operator instead, motivated from the observation that on the edges, the shape operator has rank one, and that averaging this per-edge shape operator over an area should yield an approximation to the smooth shape operator. The formulas as originally presented rely fundamentally on the existence of an embedded, Euclidean undeformed mesh, and so it is not obvious how to extend the formulation to the non-Euclidean setting where arbitrary $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ are prescribed. One possible extension of Gingold et al.'s formula is

$$S_{\mathrm{H}}(\boldsymbol{\theta}) = \sum_{i=1}^{3} \frac{\sqrt{\mathbf{w}_i^T \bar{\mathbf{a}} \mathbf{w}_i} \tan(\theta_i/2)}{\|\mathbf{w}_i\|^2 \sqrt{\det \bar{\mathbf{a}}}} \mathbf{w}_i \mathbf{w}_i^T$$

where i sums over the three edges of the canonical barycentric triangle $\mathcal{T}$; $\mathbf{w}_i$ is the vector normal to edge i and with magnitude equal to that of edge i, and θ_i is the hinge angle associated to edge i. As defined above, S_{H} takes into account length and area distortions of the triangle's rest pose due to the metric $\bar{\mathbf{a}}$, while reducing to Gingold et al.'s formula when $\bar{\mathbf{a}} = \mathbf{I}$. Given rest hinge angles $\bar{\theta}_i$, the difference $S_{\mathrm{H}}(\boldsymbol{\theta}) - S_{\mathrm{H}}(\bar{\boldsymbol{\theta}})$ can be substituted in for $\bar{\mathbf{a}}^{-1}(\mathbf{b} - \bar{\mathbf{b}})$ in the Koiter energy. In the following, we compare this formulation to the one based on the mid-edge shape operator proposed above.

Solving for Hinge Angles Given a desired $\bar{\mathbf{b}}$, rest angles $\bar{\theta}_i$ can be recovered by testing the shape operator against the $\mathbf{w}_i$: we desire

$$\frac{\mathbf{w}_i^T \bar{\mathbf{b}} \mathbf{w}_i}{\mathbf{w}_i \bar{\mathbf{a}} \mathbf{w}_i} = \mathbf{w}_i^T S_{\mathrm{H}}(\bar{\boldsymbol{\theta}}) \mathbf{w}_i, \quad i \in \{1, 2, 3\},$$

a system of three linear equations in $\tan(\bar{\theta}_i/2)$. Note that two neighboring faces might disagree on the value of the rest hinge angle of their common edge; to support non-Euclidean rest geometry, both values must be stored and used when computing the energy density of their respective triangles.

2.7 Results

2.7.1 Analytic Benchmarks

We first test the method on examples where the equilibrium state can be computed exactly. When an embedding $\mathbf{r}$ exists for which $\mathbf{a} = \bar{\mathbf{a}}$ and $\mathbf{b} = \bar{\mathbf{b}}$, that embedding is clearly a global minimizer of the elastic energy, regardless of material parameters. We ran all of the following experiments using the default values in Table 2.1. We ran the simulation for enough time to reach steady state.

Swelling of Square We take Ω to be a unit square, with $\mathbf{r}(x, y) = (x, y, 0)$, and assign static fields for both $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ to the entire shell. Table 2.2 summarizes the values of $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ we used, and the expected shape and curvatures

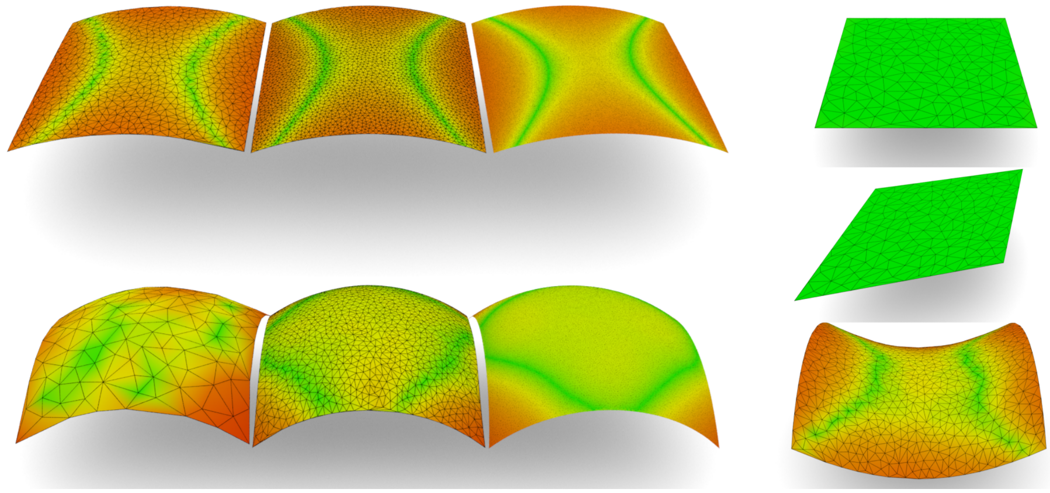

Figure 2.6: Distance between our simulated solutions and analytic solutions for the test cases in table 2.2. Warmer colors mean larger error.

of the exact solution. In cases where $\bar{\mathbf{b}} = 0$ and $\bar{\mathbf{a}}$ is homogeneous over the surface, our method must (and does) reproduce the exact solution. Examples with rest curvature are affected by discretization error; we ran the simulation using three mesh resolutions (820/2686/39210 vertices) and for each, plot the distance of each point from the corresponding point on the exact solution in Figure 2.6.

Isotropic Growth The Riemann mapping theorem guarantees that every surface M with disk topology can be parameterized by the unit disk, with metric conformally equivalent to the Euclidean metric. This insight was exploited by Kim et al [68] to grow an approximate sphere from a square. When M has finite thickness, however, we do not expect the shell to grow *exactly* into the shape M, since embedding the shell as the shape M minimizes stretch-

$\bar{\mathbf{a}}$	$\bar{\mathbf{b}}$	Expected Shape	Mean Curv.	Gauss Curv.	Error (one-sided Hausdorff)
$4\mathbf{I}$	0	Enlarged Square	0	0	$1.38 \cdot 10^{-5}$
$\frac{1}{1-x^2-y^2}\begin{pmatrix} 1-y^2 & xy \\ xy & 1-x^2 \end{pmatrix}$	$\frac{1}{1-x^2-y^2}\begin{pmatrix} 1-y^2 & xy \\ xy & 1-x^2 \end{pmatrix}$	Spherical Cap	1	1	$5.32 \cdot 10^{-2} \,//\, 3.47 \cdot 10^{-2} \,//\, 2.20 \cdot 10^{-2}$
$\begin{pmatrix} 1+x^2 & -xy \\ -xy & 1+y^2 \end{pmatrix}$	$\frac{1}{\sqrt{1+x^2+y^2}}\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$	Hyperboloid Cap	varies	< -1	$1.69 \cdot 10^{-2}$
$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$	0	Rhombus	0	0	$4.82 \cdot 10^{-11}$
$\mathbf{I}$	$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$	Cylindrical Patch	1	0	$2.23 \cdot 10^{-2} \,//\, 1.08 \cdot 10^{-2} \,//\, 1.61 \cdot 10^{-2}$

Table 2.2: Didactic experiments on a unit square. Different rest fundamental forms are prescribed over the square, and in each case we compare the simulated steady state to the expected analytic solution with mesh resolution of 820 (//2686//39210) vertices.

ing energy while ignoring bending energy. Nevertheless, we expect the steady state geometry to closely resemble M, especially when M is convex and the shell thickness h is small. Figure 2.2 shows the result of swelling the unit disk into a sphere: we stereographically project the sphere of radius 1/4 units to the plane, and take the region bounded by the unit disk. The stereographic projection is conformal, so we set $\bar{\mathbf{a}}$ on the unit disk per the conformal factor, and simulate the damped dynamics of the disk returning to its equilibrium state. Notice that the disk transitions through highly deformed intermediate states before "popping" back into a spherical shape. The rest state is not perfectly spherical at the boundary: the shell flares outward, as expected since the perfectly spherical configuration minimizes stretching energy while neglecting bending. A theoretical argument for this behavior at the boundary is given in [36].

Comparison to Finite Elements We compare our method to finite elements to validate both the discretized elastic energy, and the equilibrium configuration, of a slab undergoing differential non-Euclidean growth. More specifically, for different growth magnitudes G we prescribe the rest metric $\bar{\mathbf{g}}^{xy} = \left[\frac{(1+G)z}{h} + \left(1 - \frac{z}{h}\right)\right]^2 \mathbf{I}$ throughout the volume of a 20 by 10 mm slab of thickness $h = 0.1$ mm, and evaluate its elastic energy using three methods: (i) tetrahedral finite elements using a St. Venant-Kirchhoff material model, where each tetrahedron is assigned a piecewise constant rest metric by evaluating $\bar{\mathbf{g}}$ at its centroid; (ii) our method, as described in section 2.3; (iii) our method, using Gingold et al.'s hinge-based shape operator, as discussed in section 2.6. All finite element energies and simulations were computed using the Vega library [7]. We also compute two ground truth energy values: the true elastic energy of a St. Venant-Kirchhoff material integrated over the shell volume, and the energy of the midsurface-based thin limit approximation in equation 2.2. Figure 2.7 shows convergence plots of the elastic energy of the finite element and shell methods as a function of mesh resolution. For a fixed value $G = 0.03$, we compute the energy of a non-equilibrium embedding of the slab $\mathbf{r}(x, y) = (\sin[100x], 100y, \cos[100x])$. Our method converge linearly to the Koiter energy (2.2), in agreement with previous studies of Euclidean shells discretized using the mid-edge $\mathbf{b}$ [154]. Our method converges to an energy slightly higher than the exact shell energy; we believe the discrepency is due to membrane locking, i.e. the impossibility of bending a discrete surface without stretching any triangles [37]. Figure 2.9 plots the error of both shell methods relative to the FEM solution, for different levels of refinement; on a coarse mesh our method slightly underestimates the curvature of the FEM

Method	$G = 0.005$	0.03	0.05	0.08	0.1	0.3	0.5
True Volumetric Energy	$1.60 \cdot 10^{-6}$	$5.64 \cdot 10^{-5}$	$1.55 \cdot 10^{-4}$	$3.87 \cdot 10^{-4}$	$5.97 \cdot 10^{-4}$	$4.77 \cdot 10^{-3}$	$1.20 \cdot 10^{-2}$
Thin Limit Approximation	$1.60 \cdot 10^{-6}$	$5.77 \cdot 10^{-5}$	$1.60 \cdot 10^{-4}$	$4.10 \cdot 10^{-4}$	$6.40 \cdot 10^{-4}$	$5.67 \cdot 10^{-3}$	$1.54 \cdot 10^{-2}$
Our Energy	$1.60 \cdot 10^{-6}$	$5.77 \cdot 10^{-5}$	$1.60 \cdot 10^{-4}$	$4.10 \cdot 10^{-4}$	$6.40 \cdot 10^{-4}$	$5.67 \cdot 10^{-3}$	$1.54 \cdot 10^{-2}$
Hinge-based Formulation	$2.20 \cdot 10^{-6}$	$8.01 \cdot 10^{-5}$	$2.29 \cdot 10^{-4}$	$6.11 \cdot 10^{-4}$	$9.83 \cdot 10^{-4}$	$1.23 \cdot 10^{-2}$	$5.01 \cdot 10^{-2}$
Volumetric Finite Elements	$1.60 \cdot 10^{-6}$	$5.60 \cdot 10^{-5}$	$1.53 \cdot 10^{-4}$	$3.85 \cdot 10^{-4}$	$5.93 \cdot 10^{-4}$	$4.74 \cdot 10^{-3}$	$1.19 \cdot 10^{-2}$

Table 2.3: Comparison of the elastic energy of a thin slab, simulated using tetrahedral finite elements, our method using the mid-edge shape operator, and using Gingold's hinge-based shape operator. Higher G induces more differential growth.

solution; the formulation using Gingold et al.'s operator overestimates it. The overall one-sided Hausdorff distance between the simulation and ground truth (as computed using volumetric FEM) for our method, from coarse to fine, is $2.25 \cdot 10^{-4}$, $8,93 \times 10^{-5}$, and $8.09 \cdot 10^{-5}$ m. Using the hinge-based shape operator instead, the distance is $2.9 \cdot 10^{-4}$, 2.45×10^{-4}, and 2.34×10^{-4} m.

Table 2.3 compares the limit energy values of the discrete methods to the ground truths, for different values of G (the meshes used for this experiment contain 22881 tetrahedra/ 5538 triangles). Our method shows good agreement with the Koiter energy. Notice that for high values of G, even the exact Koiter energy does not agree with the true volumetric elastic energy; this is because as G grows large, the modeling assumption that strain is small relative to the thickness of the shell no longer holds for this imposed embedding.

2.7.2 Comparisons to Experiments

Curling of Paper Ordinary paper, when moistened on one side (by painting water, or placing the paper on a damp sponge), undergoes complex dynamic behavior. Water diffuses into the wet side, and that side swells, causing global curling due to this differential material growth. Over time, water penetrates the entire thickness of the paper and the water concentration becomes uniform; the paper flattens again. This process plays out over about ten seconds. We compare experiment and simulation of this behavior: we use a brush to moisten a piece of real paper, and compare the video of the curling and uncurling of the paper to a simulation of the same phenomenon. Figure 2.5 shows the experiment, the simulation, and a stylized, thickened cross-section of the paper showing the difference in moisture concentration through the paper thickness over time.

Radial Swelling of Disk Sharon and Efrati [124], in their pioneering work on the geometry of non-Euclidean plates, induced non-uniform growth in a punctured disk of NIPA polymer by varying the cross-linking ratio as a function of the radius away from the disk's center. We compare simulation and experiment in figure 2.8.

2.7.3 Effects of Parameters

One advantage of our formulation of swelling thin shells is that it supports physical material parameters. In this section, we show results highlighting the importance of these parameters and their effects on the behavior of swelling and shrinking thin objects.

Thickness Since it controls the relative importance of stretching and bending, the thickness of a shell is its most important physical parameter. Changing the thickness will often dramatically change how a shell deforms. For example, when a moist paper annulus dries, it rolls up. Depending on its thickness, the annulus transitions through several intermediate shapes before rolling up completely: the outer boundary of the annulus lifts to form an n-sided polygon, with n decreasing over time as one metastable configuration cascades into the next. Figure 2.11 shows the range of behavior for an annulus of inner radius 1 cm and outer radius 2 cm, and several paper thicknesses (0.05, 0.1, 0.2 and 0.5 mm).

Machine Direction As mentioned above, when paper is manufactured, its fibers arrange in a preferential direction, leading to anisotropic growth when the paper is moistened. In figure 2.11 (right) we repeat the annulus experiment with paper whose fibers are aligned to the x direction (left to right in the figure). Notice the dramatic change in the annulus dynamics, compared to figure 2.11 (left).

2.7.4 Qualitative Experiments

We end with several experiments that demonstrate the flexibility of the method to simulate complex geometries undergoing nonlinear deformations due to differential swelling.

Shape-changing Food Differential growth has been used to design gelatin films that fold into novel shapes, by exploiting the ability to fabricate the film

Example	Fig	verts	tris	time step (s)	time/step (s)
Bunny	2.4	51k	102k	10^{-5}	5.25
Armadillo	2.4	42k	86k	10^{-5}	33.88
Pasta	2.3	9k	19k	10^{-4}	23.6
Globe	2.2	5k	10k	10^{-4}	0.5
Annuli	2.11	1.5k	3k	10^{-4}	8

Table 2.4: Timing numbers for the simulations shown in the paper. Each sim used one thread on an Intel Xeon E3-1270 desktop with 16GB RAM.

into a bilayer of differing density and porosity [150]. Wang et al. validated their designs using a volumetric simulation in ABAQUS; in figure 2.3 we compare their simulated and experimental results to our thin shell simulations, demonstrating that a reduced shell model can accurately predict the behavior of the bilayer without need of a volumetric simulation.

Melting Plastic We simulate the behavior of several thin-shell plastic objects (sphere, torus bunny, and armadillo) shrinking when exposed to heat. Notice the complex buckling patterns visible in these examples, due to metric incompatibility introduced into the object geometry during heating. See figure 2.4 and figure 2.10.

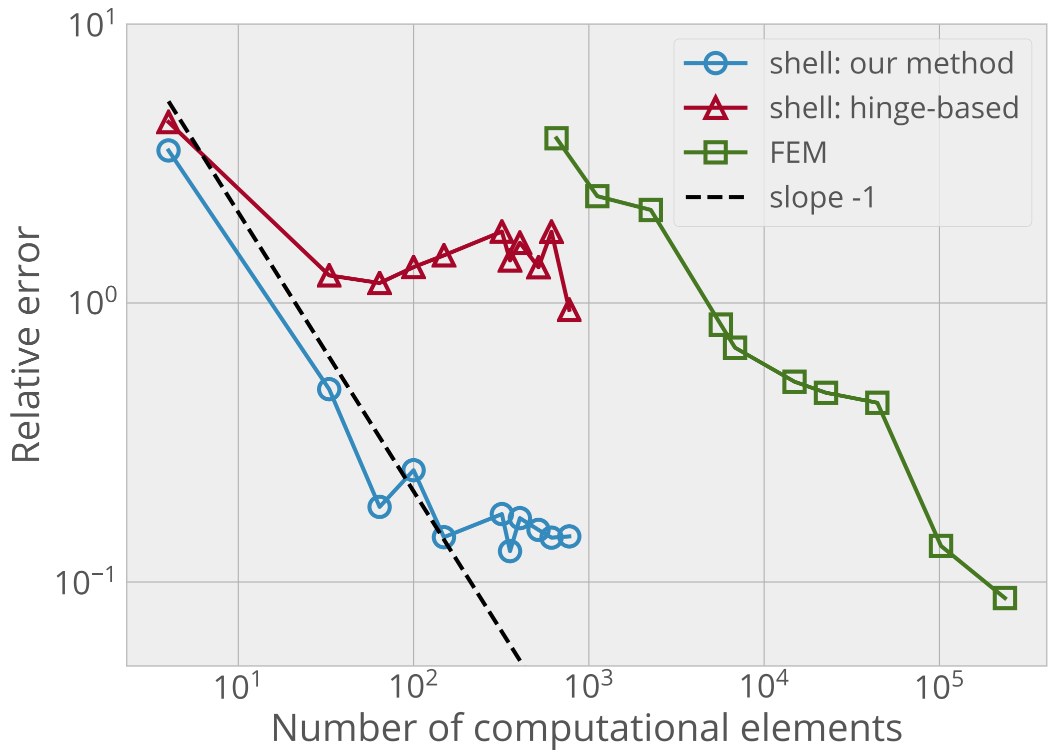

Figure 2.7: Comparison of energy convergence rate as a function of resolution of our method, the alternative hinge-based formulation, and volumetric FEM. The FEM energy was compared to the analytic volumetric elastic energy. For both shell methods, the computed energy was compared to the analytic thin-limit energy in equation (2.2). Both our method and FEM show linear convergence to their respective exact energies.

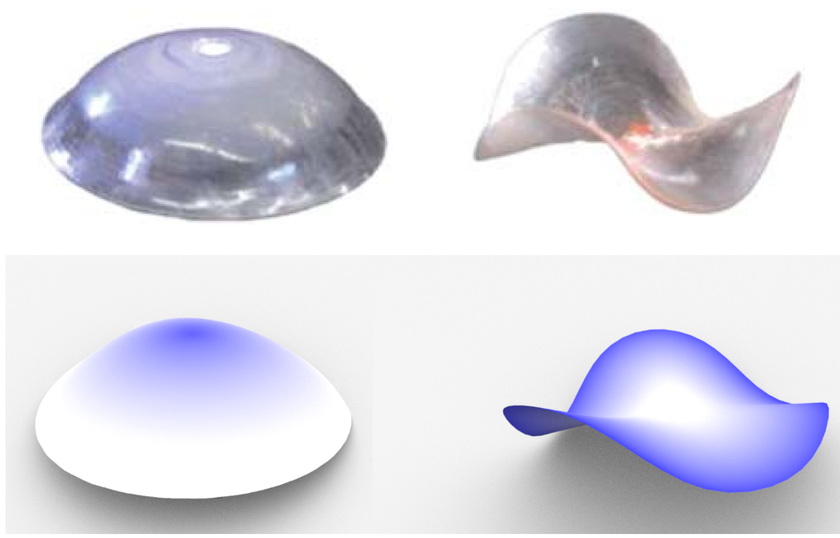

Figure 2.8: We compare the experiments of Sharon and Efrati [124] using polymer disks (top) with our simulation (bottom) for two prescribed radially-varying rest metrics. Blue regions correspond to higher growth rates.

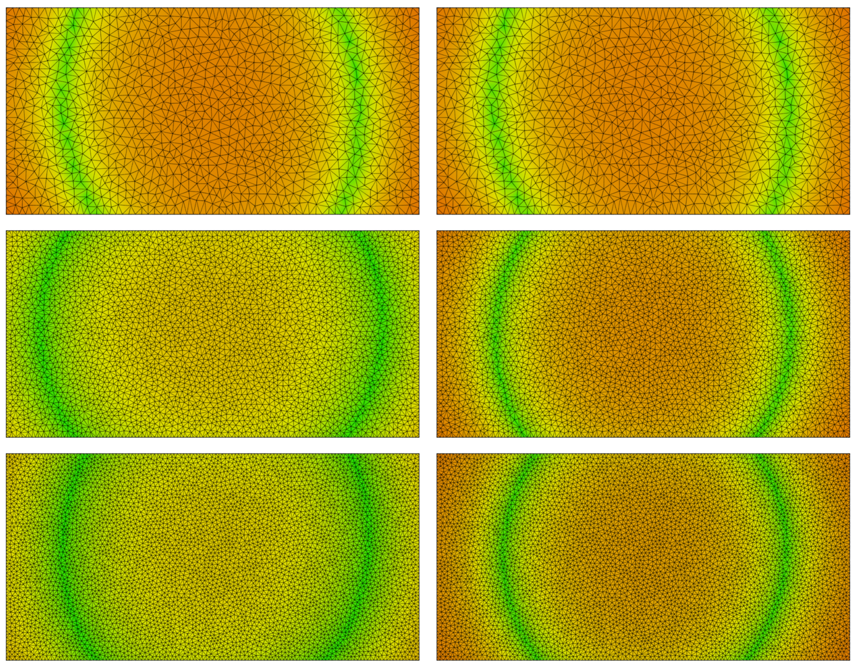

Figure 2.9: Visualization of distance between solutions using thin shell simulations of our method (left), and our method with hinge-based shape operator (right), compared to FEM for meshes with 1277, 3594, and 5538 vertices, plotted over the slab's material domain.

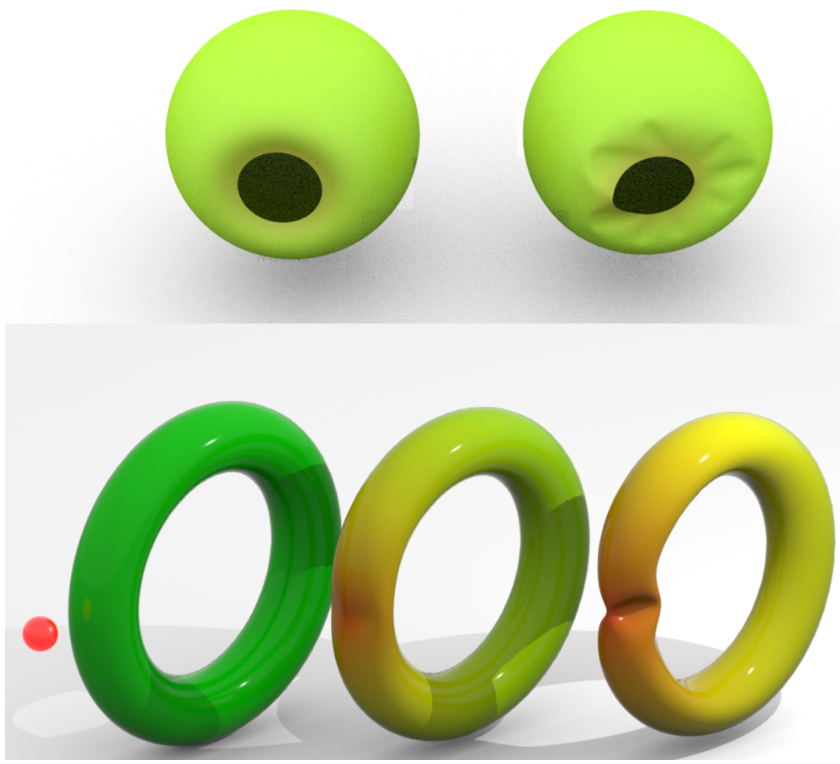

Figure 2.10: Deformed punctured plastic spheres with different temperature distribution with hotter region indicated as red are shown on the top panel. Plastic torus deformation due to heat source (red sphere) through time is shown in the bottom.

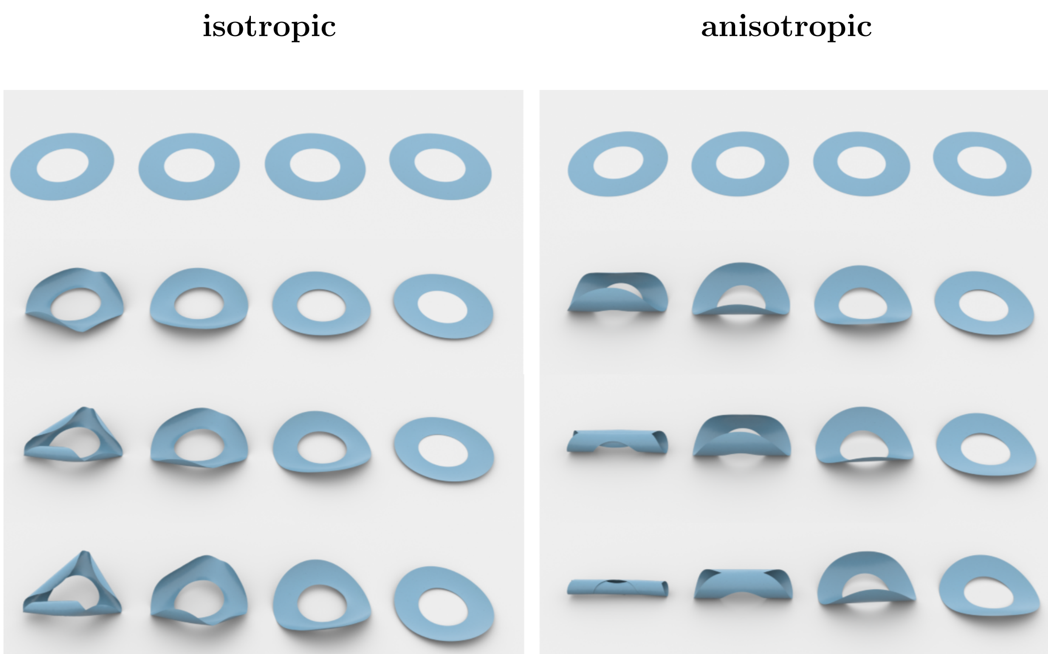

Figure 2.11: The left figure shows the curling behavior of moist paper annuli as they dry, with time increasing from top to bottom. Varying the thickness (from left to right, the thicknesses of the annuli are 0.05 mm, 0.1 mm, 0.2 mm, and 0.5 mm) demonstrates that the amount and type of curling is strongly dependent on the thickness of the material. The right figure shows the same annuli endowed with a principal machine direction along the x axis (directed from left-to-right in the figure): the dynamics are markedly different from the case of no machine direction.

Chapter 3

Constraint-Based Thin Shell Simulation on Point Clouds

Efficiently simulating thin objects such as cloth, paper, or skin remains significantly more challenging than simulating their rigid or volumetric counterparts, due to their kinematic complexity and nonlinear physics. Specifically, numerical discretizations of thin plates must overcome two challenges: the dramatic stiffness disparity between stretching and bending material forces, and the tendency for discretizations of thin surfaces to *lock*.

- **Stiffness scale-separation.** The behavior of thin plates is governed by a combination of *membrane* forces, resisting in-plane stretching and shear of the material, and *bending* forces. These forces have dramatically different stiffnesses: stretching stiffness scales like the thickness of the material, whereas bending scales like thickness cubed, so that the stretching forces are order of magnitude stronger.

  For sufficiently thin objects, the scale separation between these stiffnesses is so vast that stretching of the material is imperceptible; in this setting paying the price in element or time step size in order to resolve the stretching modes does not make sense. Instead, one can look at *isomet-*

ric kinematics, where constraints enforcing zero in-plane strain replace force-level resistance to stretching. However, such a strategy requires discrete surface kinematics that supports cleanly decoupling bending from stretching modes.

- **Membrane locking.** A smooth surface can bend into a general developable surface isometrically; the same is not true for a triangle mesh. Regular equilateral meshes can isometrically bend about their three axes of symmetry only; irregular meshes cannot bend at all without distorting some of the triangle edges. As a consequence, any formulation of stretching forces based on triangle edge lengths will suffer from *locking*: situations, such as holding a piece of cloth up by two corners as shown in Figure **??**, left, where deformations that are bending-dominated in the continuous regime are stretching-dominated in the discrete regime [117], due to failure of bending to kinematically decouple from stretching.

  On the other side of the same coin, insufficiently constraining discrete inextensible plates, so that stretching-dominated deformations in the continuous regime are instead bending-dominated (or completely uncontrolled), allows equally undesirable *spurious deformation modes*.

Main Idea We present a method for simulating isometric thin plates by combining two simple ideas: replacing the very stiff membrane forces by hard constraints, and doing so on a formulation of the strain tensor that is averaged over surface patches via moving least squares, to alleviate locking. We can then simulate the material by adding a bending energy and enforcing the hard inextensibility constraints using standard methods for constrained Lagrangian dynamics, such as the method of Fast Projections [48].

The resulting numerical method is simple to implement, and because stretching forces are replaced with hard constraints, it can simulate infinitesimally thin, inextensible materials without requiring very fine meshes or small time steps. Even very coarse, efficient simulations of thin materials undergoing large amounts of bending and crumpling give good qualitative results.

3.1 Related Work

Membrane Locking is a well-known phenomenon for simulations involving low-order elements, and has been studied extensively by Quaglino [116, 117], who proposed several tests characterizing behavior of a variety of triangle-mesh-based kinematics. English and Bridson [37] suggested gluing triangles at edge midpoints rather than at vertices, which avoids locking but unfortunately suffers from spurious modes. Popular workarounds to locking include adaptive refinement of the mesh, as in the popular ArcSim [100] code and its extensions; avoiding triangles entirely and using higher-order elements; or compensating for locking by tweaking stiffness parameters on an ad-hoc per-example basis.

Isometry/Strain Limiting Treating membrane strain with constraints rather than forces has proven to be a powerful technique for reproducing characteristic wrinkle patterns in thin shell materials. Early application of this idea was used to introduce fine wrinkles where contact introduces compression of the materials [114, 19]. Goldenthal et al. [48] proposed a framework for constraint-based limiting of strain in the warp and weft directions on a quadrilateral mesh, and demonstrated that constraint-based inextensibility is significantly more efficient than force-based simulation of membrane strain, even when using implicit methods. Chen and Tang [21] enforce isometry in a least-squares sense while also respecting collision constraints. Other methods for enforcing strain limits have been proposed, with support for constraining all elements of the strain tensor [138] and for boosting performance by applying constraints in a hierarchical manner [147], though these methods treat strain based on triangle deformations and will lock unless the material is sufficiently compliant when under the strain limit. Also related is the method of position-based dynamics [98, 132], a stable and fast alternative to traditional physical simulation that replaces *all* forces (even the soft bending forces) with constraints.

Meshless Methods Although most cloth solvers are mesh-based, meshless methods have also been explored [163]. They are particularly appealing for problems involving fracture; peridynamics [125] has had significant success in computer graphics [74, 22, 56, 158] and has been extended to shells [26]. Similar in spirit are Elastons [86], a quadrature scheme for deformation energy based on unstructured sample points which can be used to simulate elastic bodies of arbitrary codimension in a unified way. Also related are the "F-bar" methods in finite elements for simulating incompressible volumes, which avoids locking by imposing area/volume constraints on patches of elements rather than on single quadrilaterals [30] or triangles [102]. This idea has been applied to simulation of incompressible volumes [107, 17], including in graphics [61].

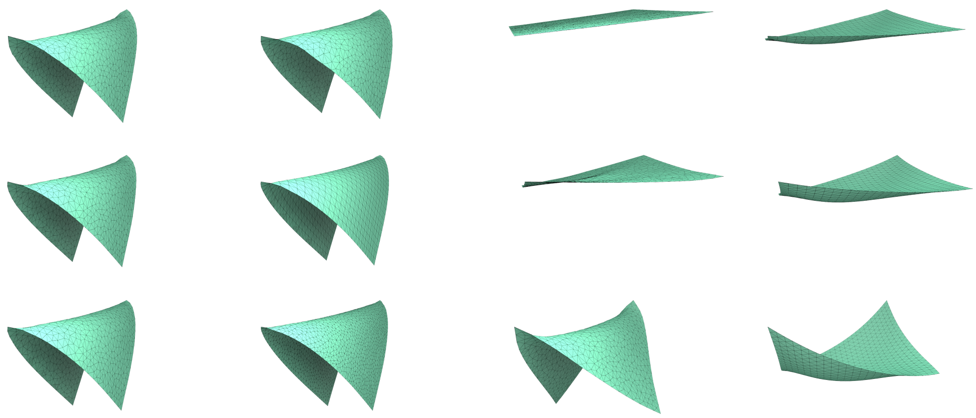

Figure 3.1: Locking test on a square piece of cloth pinned at the corner. *Left column*: simulation using our constraints, with constraints tolerance 0.1, 0.01, and 0.001 from top to bottom. *Second column*: sampling with twice the neighborhood size, on a regular grid, and on a finer mesh. All these simulations have similar curvature and do not lock. *Third column*: simulations using discrete elastic shells [50] for the same bending stiffness as our simulations, but varying stretching stiffnesses, decreasing from top to bottom (Young's moduli: 10^4, 100, and 1 MPa). Notice there is no free lunch between excessive in-plane strain, and locking. *Last column*: Simulation using isometry constraints on all edges of a quad mesh with varying diagonal spring stiffness, decreasing from top to bottom [48].

3.2 Meshless Kinematics

In this section, we describe the meshless kinematics we use, and our notation. The heart of our method will be in Section 3.3, where we formulate constraints on neighborhoods of the material. We assume we are given a surface $\mathcal{S}$ whose rest state is flat. We adopt a standard meshless discretization of the plate, and sample S at N points $\mathcal{P}$ of $\mathcal{S}$, and associate to each point a neighborhood $\mathcal{N}_i \subset \mathcal{P}$ of other sample points. We require that $|\mathcal{N}_i| \geq 2$; the neighboring points can be chosen based on a threshold distance away from the sample points on $\mathcal{S}$, graph distance on a user-provided input triangle mesh, etc.

We assume that locally, each neighborhood $\mathcal{N}_i$ can be isometrically parameterized by a region Ω_i of the plane (this parameterization might be given, in the case of cloth sewing patterns, or precomputed from S by plane-fitting). We will write $X^i_j \in \mathbb{R}^2$ for the position of sample point $j \in \mathbb{N}_i$ on Ω_i; note that a sample point likely belongs to multiple neighborhoods and might have different material coordinates X^i_j in each such neighborhood. Let $Y_i \in \mathbb{R}^3$ be the embedded position of sample point i in 3D. The set of Y_i are then the degrees of freedom of the simulation. Finally we associate to each neighborhood a lumped mass m_i (based on a given material density and the barycentric area of its associated sample point).

3.3 Isometry Constraints

Motivation The most obvious way to enforce isometry of a surface is to discretize it as a triangle mesh, and constrain each edge length to remain constant. However, such constraints are doomed to lock. Consider for instance that for a mesh with $|V|$ vertices and $|E|$ edges, there are $3|V|$ kinematic degrees of freedom and $|E|$ constraints, and yet from the Euler characteristic formula,

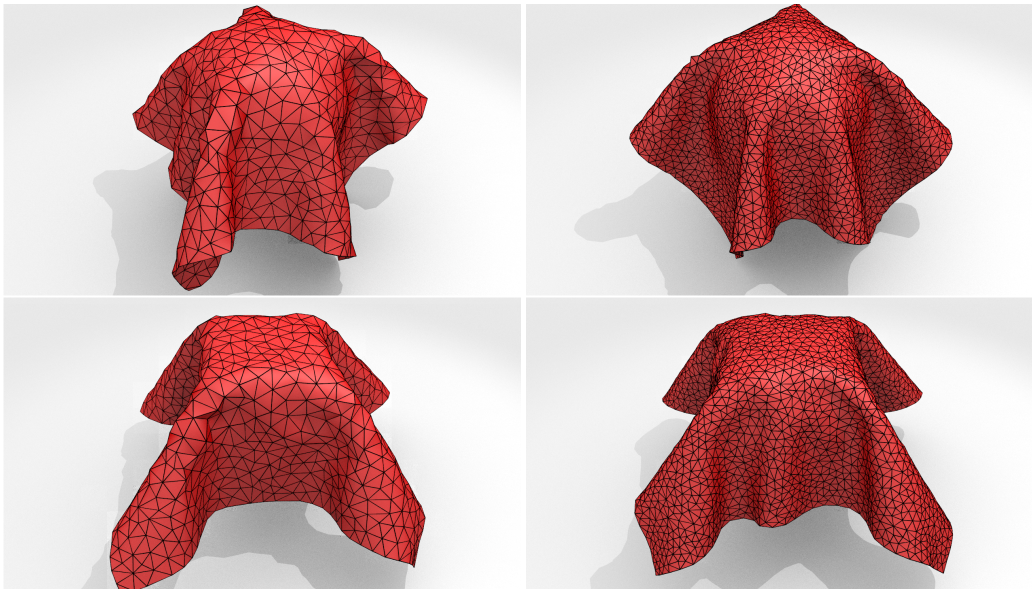

Figure 3.2: Draping experiments using coarse and fine meshes (left and right columns, respectively). Our constraints yields consistent results even for very coarse discretizations of the cloth.

$|E| \approx 3|V|$; the edge-based isometry constraints are therefore expected to remove almost all of the cloth's degrees of freedom. Contrast this situation with the smooth setting, where the cloth can deform into a rich variety of developable surfaces. Note that replacing the triangle mesh with quadrilaterals does not in any way solve the problem: one can then enforce isometry for only the quadrilateral edges (as in Goldenthal et al. [48]), which allows non-isometric shear and stretching in the diagonal direction (see Figure 3.1), or also constrain the lengths of the diagonals, which essentially triangulates the mesh.

We propose instead to enforce isometry on the *vertices*: we will use moving least-squares to formulate a strain on each of the $|V|$ neighborhoods $\mathcal{N}_i$, and constraint the principle strains to be zero. We will then have only $2|V|$, rather than $|E|$, isometry constraints, leaving $|V|$ leftover degrees of freedom for isometric bending modes.

Constraint Formulation Near a sample point i, the 3×2 deformation gradient F of the thin plate, with respect to the local parameterization of $\mathcal{N}_i$, must satisfy

$$FX_j^i \approx Y_j - Y_i \quad \forall j \in \mathcal{N}_i. \tag{3.1}$$

Of course, for points i with more than two neighbors, this relation is over-constrained; we instead work with an *averaged* deformation gradient F_i in $\mathcal{N}_i$, based on satisfying Equation 3.1 in the moving least-squares sense,

$$F_i = \arg\min_F \sum_{j \in \mathcal{N}_i} m_j \left\| FX_j^i - (Y_j - Y_i) \right\|^2.$$

The averaged deformation gradient F_i can then be expressed in closed form as

$$F_i = \mathbf{YWX}^T(\mathbf{XWX}^T)^{-1},$$

where each column of $\mathbf{X}_{2\times|\mathcal{N}_i|}$ is one X_j^i, each column of $\mathbf{Y}_{3\times|\mathcal{N}_i}$ is one current displacement $Y_j - Y_i$, and $\mathbf{W}_{|\mathcal{N}_i|\times|\mathcal{N}_i|} = \operatorname{diag}(m_j)$.

Isometry Given current deformation gradient F_i for a neighborhood of point i, we can formulate the strain tensor,

$$\varepsilon_i = (F_i^T F_i - I),$$

for I is the identity matrix. Typically for dynamics we would then derive forces by applying a constitutive law to this strain; since we instead want to enforce inextensibilty as a hard constraint, we need to write down constraint functions specifying vanishing of the strain. There are several sets of constraints we could choose; unfortunately, all are nonlinear. We use the pair

$$\begin{aligned} g_i^{\mathrm{tr}} &= \mathrm{tr}\left(F_i^T F_i\right) - \mathrm{tr}\,(I) = 0 \\ g_i^{\mathrm{det}} &= \det\left(F_i^T F_i\right) - \det\,(I) = 0, \end{aligned} \tag{3.2}$$

Since the zero matrix is the only symmetric matrix with both eigenvalues zero, these constraints are equivalent to $\varepsilon_i = \mathbf{0}$. The gradients of these constraints, in local coordinates with respect to a variation $\delta\mathbf{Y}$ of $\mathbf{Y}$, are given by

$$\begin{aligned} \nabla g_i^{\mathrm{tr}} \cdot \delta\mathbf{Y} &= 2F(\mathbf{X}\mathbf{W}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{W} : \delta\mathbf{Y} \\ \nabla g_i^{\mathrm{det}} \cdot \delta\mathbf{Y} &= 2F(F^T F)^{\mathrm{adj}}(\mathbf{X}\mathbf{W}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{W} : \delta\mathbf{Y}, \end{aligned}$$

where

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{\mathrm{adj}} = \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

Notice in particular that both constraints have zero as a regular value: neither gradient vanishes when $\varepsilon_i = 0$, a crucial requirement for the numerical stability of techniques like Fast Projections that are based on the method of Lagrange multipliers. Intuitively, the trace of the strain tensor corresponds to the sum of the squared principle stretches, which is in a sense the averaged squared distance to the neighboring points. The determinant of the strain tensor is the product of the squared principle stretches and measures a notion of squared neighborhood area. Both constraints together imply that the stress tensor, averaged in each neighborhood, is the zero matrix, equivalent to isometry of the neighborhood.

We mention in passing some alternative sets of constraints we tried, and abandoned: perhaps the most obvious is to directly constrain each entry of ε_i, which amounts to three constraints per neighborhood since strain is symmetric. However, this formulation bloats the set of constraints by 50%, while introducing rank deficiency in the constraint gradients, leading to poor performance. One might also consider constraining the trace and determinant of the strain tensor, rather than of $F_i^T F_i$—the problem with this approach is that the function $\det(\varepsilon_i)$ has a critical point at the strain-free state, which as mentioned above makes the constraint unsuitable for projection.

3.4 Time Integration

We enforce the constraints (3.2) at every time step using the method of Fast Projections [48], with implicit time integration of bending and external forces. Any bending model can be used, though we note that for materials which are rest flat (including cloth, paper, etc), the simple and efficient quadratic bending model [12] is quite attractive, since isometry of the material is precisely the assumption required for validity of the model:

$$E_{\text{bend}} = \frac{k}{2}\|L\mathbf{Y}\|_{M^{-1}}^2,$$

where M is the system mass matrix, L the Laplace-Beltrami operator (computed from a meshless discretization [113], or from a triangulation of the input surface), and k is a bending stiffness proportional to the Young's modulus and cubed thickness of the material. Note that the quadratic bending energy, true to its name, has constant Hessian which can be prefactored, so that the performance bottleneck of each time integration step is enforcing the constraints.

In most of our experiments, we used a time step size of 10^{-3} seconds and a constraint tolerance of 0.1; Fast Projections typically converges in one to three iterations. However, Fast Projections does not perform a true projection onto the constraint manifold, and comes with no guarantees; we found that regularizing each projection by the geometric stiffness matrix of the constraints [139] improves performance. In cases where Fast Projections fails to decrease the constraint residual after a few iterations, we switch to true projection using the augmented Lagrangian method; in practice this fallback was only needed when high-energy impacts cause large violations in the constraints within a single time step.

3.5 Results

Locking Tests We perform a sequence of tests demonstrating that the constraints (3.2) works for near isometric thin plate and does not exhibit spurious modes. In Figure 3.1, we pin a square sheet of cloth at two corners, and let the sheet relax under gravity. We simulate this experiment using our isometry constraints for different constraint tolerances, for different resolutions of sampling points, and for both ordered and disordered sampling of the cloth. Our simulations consistently give comparable results. Statistics for these and other experiments described in this section can be found in Table 3.1.

Effect of Parameters As we decrease the constraint tolerance as shown in the left column in Figure 3.1, the sagging in the middle of the cloth decreases as expected, since the isometry constraints enforce that the distance between the two pinned corners is identical to that of the flat rest shape.

Comparison to Related Work We compare to force-based shell methods [50], using the same bending stiffness, on the third column of Figure 3.1, where enforcing isometry via high stiffness rigidifies the cloth. Lastly, the same experiment was performed using the constraints proposed by [48]. Due to the fact that the edge length constraints on a quad method do not control the length of each quad diagonal, the cloth sags significantly.

Crushed Cylinder We replicate the cascade of diamond buckling patterns observed by Martin2010, and although only our high-resolution simulation can resolve the fine-scale pattern at the beginning of the simulation, it correctly reproduces the highly-buckled final state.

Spinning tank top We show the dynamics of a tank top draping over a spinning hanger in Figure 3.4. Our method resolves the wrinkles generated during the motion.

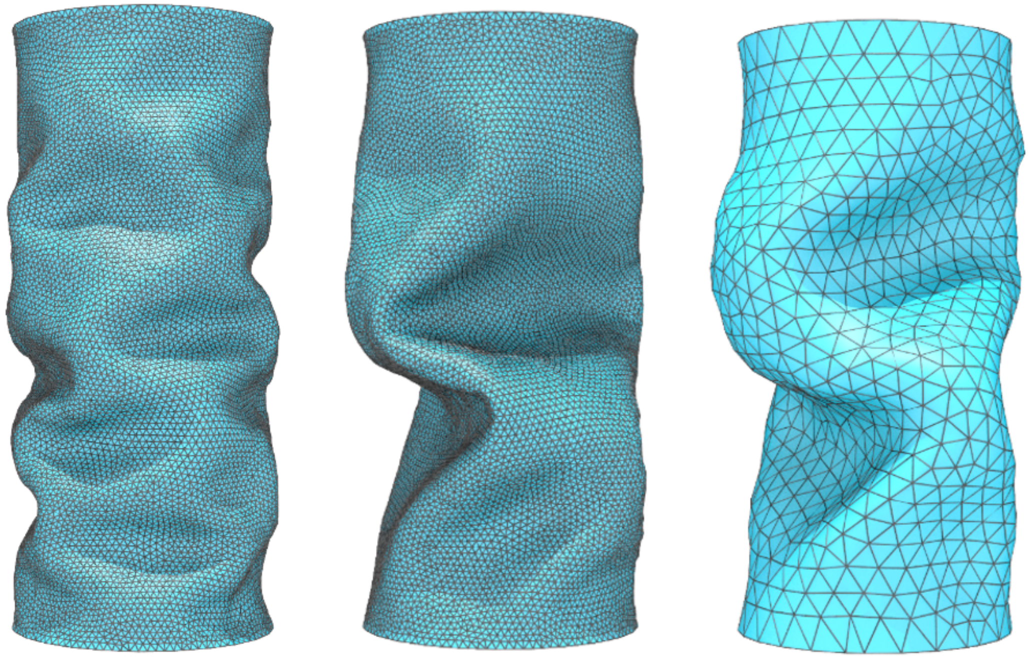

Figure 3.3: Frames of cylinder crumpling using our method, for a fine (*left and middle*) and coarse (*right*) simulation. Only the fine simulation resolves the fine-scale pattern but both simulations settle to similar static states.

Bending Stiffness One important feature of our method is that the bending stiffness of the simulated cloth can be controlled, while preserving isometry. For most thin shell models, extensive tuning of the hyper-parameters is often required to achieve the balance between avoiding locking and maintaining isometry of the material. As shown in Figure 3.5, just by adjusting the bending stiffness, our method produces cloth with different wrinkling patterns without stretching.

Figure 3.4: Dynamics of a spinning tank top.

Skin Wrinkling Several papers [119, 75] describe how to simulate wrinkling of skin via one-way coupling of thin plate simulations to a volumetric elastic substrate. The coupling is enforced via a sparse set of average-position constraints sampled on the skin surface. This setup is ideal for our method, since (i) the behavior of the upper skin layers is governed by bending and by the coupling constraints, so that replacing in-plane strain of the skin with isometry of these fine layers is justified; (ii) the coupling constraints can be trivially incorporated by including them as extra terms during Fast Projections. Figure 3.6 shows the behavior of our simulation when the substrate is compressed in one and two directions.

Table 3.1: Parameters used in each experiment.

	Fig. 3.1 coarse	Fig. 3.1 regular	Fig. 3.1 fine	cylinder(coarse)	cylinder (fine)	tank top
verts	662	625	1656	979	15039	3450
tolerance	0.01	0.01	0.01	0.001	0.001	0.01

Figure 3.5: Our method generates draped skirt shapes with different wrinkle patterns while preserving the isometry.

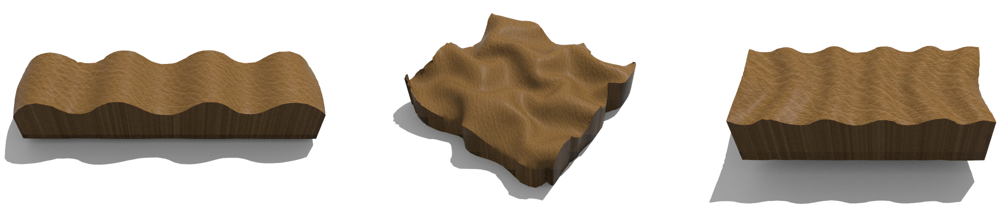

Figure 3.6: Simulations of skin coupled to a volumetric substrate. Even for coarse discretizations, our method reproduces the correct undulating or 3D-folded wrinkling pattern for compression in one or two directions, and pinching.

Chapter 4

Virtual Elastic Objects

3D reconstruction is one of the fundamental problems of computer vision and a cornerstone of augmented and virtual reality. A recent achievement in this direction is the discovery of a fairly general formulation for representing radiance fields [94, 79, 87, 123, 164, 161, 141, 13, 128, 104, 133]. Neural radiance fields are remarkably versatile for reconstructing real-world objects with high-fidelity *geometry* and *appearance*. But static appearance is only the first step: it ignores how an object moves and interacts with its environment. 4D reconstruction tackles this problem in part by incorporating the time dimension: with more intricate capture setups and more data, we can reconstruct objects over time—but can only re-play the captured sequences. Today, in the age of mixed reality, a photo-realistically reconstructed object might still destroy immersion if it is not "physically realistic" because *the object cannot be interacted with.* (For example, if a soft object appears as rigid as the rocks next to it when stepped on.)

By building on advances in computer vision and physics simulation, we begin to tackle the problem of physically-realistic reconstruction and create *Virtual Elastic Objects*: virtual objects that not only look like their real-world counterparts but also behave like them, even when subject to novel interactions. For the first time, this allows for full-loop reconstruction of deforming elastic objects: from capture, to reconstruction, to simulation, to interaction, to re-rendering.

Our core observation is that with the latest advances in 4D reconstruction using neural radiance fields, we can both capture radiance and deformation fields of a moving object over time, and re-render the object given novel deformation fields. The remaining challenge is to capture an object's physics from observations of its interactions with the environment. With the right representation that jointly encodes an object's geometry, deformation, and material behavior, compatible with both differentiable physical simulation and the deformation fields provided by 4D reconstruction algorithms, we can use these deformation fields to provide the necessary supervision to learn the material parameters.

But even with this insight, multiple challenges remain to create Virtual Elastic Objects. We list them together with our technical contributions:

1) Capture. To create VEOs, we need to collect data that not only contains visual information but also information about physical forces. We present the new **PLUSH** dataset[1] containing occlusion-free 4D recordings of elastic objects deforming under known controlled force fields. To create this dataset, we built a multi-camera capture rig that incorporates an air compressor with a

[1] `https://hsiaoyu.github.io/VEO/`

movable, tracked nozzle. More details can be found in Sec. 4.2.1.

2) Reconstruction. VEOs do not require any prior knowledge about the geometry of the object to be reconstructed; the reconstruction thus must be template-free and provide full 4D information (*i.e.*, a 3D reconstruction and deformation information over time). We extend Non-rigid Neural Radiance Fields [140] with novel losses, and export point clouds and point correspondences to create the data required to supervise learning material behavior using physical simulation. We provide further details in Sec. 4.2.2.

3) Simulation. Crucially for creating realistic interactive objects, a physical simulation is required, both to optimize for an unknown object's physical parameters and to generate deformations of that object in response to novel interactions. We implement a differentiable quasi-static simulator that is particle-based and is compatible with the deformation field data provided by our 4D reconstruction algorithm. We present the differentiable simulator and explain how we use it to obtain physical parameters in Sec. 4.2.3, and describe simulations of novel interactions in Sec. 4.2.4.

4) Rendering. Since we convert from a neural representation of the captured object's geometry to a point cloud reconstructing the object's physical properties, we require a function that allows rendering the object given new simulated deformations of the point cloud. We introduce a mapping function that enables us to use deformed point clouds instead of continuous deformation fields to alter the ray casting for the Neural Radiance Fields we used for the original reconstruction. Further details on re-rendering can be found in

Sec. 4.2.5.

4.1 Related Work

Our work integrates together multiple areas of computer vision, computer graphics, and simulation.

Recovering Elastic Parameters for 3D Templates. A number of prior works estimate material parameters of a pre-scanned 3D template by tracking the object over time from depth input. Wang *et al.* [146] were among the first to tackle tracking, rest pose estimation, and material parameter estimation from multi-view depth streams. They adopt a gradient-free downhill simplex method for parameter fitting, and can only optimize a limited number of material parameters. Objects built from multiple types of materials cannot be faithfully captured without manual guidance or prior knowledge of a part decomposition. Hahn *et al.* [55] learn an inhomogeneous viscoelastic model from recordings of motion markers covering the object. Recently, Weiss *et al.* [155] infer homogeneous linear material properties by tracking deformations of a given template with a single depth camera. In contrast to these methods, ours jointly reconstructs not just object deformations and physics *without a need for depth input or markers* but also geometry and appearance *without a need for a template.* Our formulation can model inhomogeneous, nonlinear materials without prior knowledge or annotations.

3D/4D Reconstruction. Representing static scenes remains an open problem, with recent mesh-based [148, 52] and neural approaches [25, 91, 118].

Reconstructing non-rigid objects from a video sequence is an equally long-standing computer vision and graphics problem [165, 142]. Shape-from-Template methods deform a provided template using RGB [162] or RGB-D data [167]. DynamicFusion [103] is a model-free, real-time method for reconstructing general scenes from a single RGB-D video. When reliable 2D correspondences are available from optical flow, non-rigid structure-from-motion (NRSfM) can be used to reconstruct the 3D geometry [2, 71], perhaps even using physics-based priors [3]. There are also image-based approaches that do not yield a true 3D scene [160, 10]. Recently, reconstruction using neural representations have become more common. Whereas OccupancyFlow [105] requires 3D supervision, Neural Volumes [81] reconstructs a dynamic scene from multi-view input only, but does not compute temporal correspondences. See a recent survey on neural rendering [137] for more.

Neural Radiance Fields [94], the seminal work of Mildenhall *et al.*, lays the groundwork for several follow-up reconstruction methods that extend it to dynamic scenes [76, 109, 4, 115, 108, 77, 34, 41, 156, 82]. In this work, we assume multi-view RGB video input with known camera parameters and foreground segmentation masks and so extend Non-Rigid Neural Radiance Fields (NR-NeRF) [140].

Data-Driven Physics Simulation. For simulating elastic objects specifically, one line of work replaces traditional mesh kinematics with a learned deformation representation to improve performance: Fulton *et al.* [40] use an autoencoder to learn a nonlinear subspace for elastic deformation, and

Holden *et al.* [57] train a neural network to predict the deformation of cloth using a neural subspace. Some methods use neural networks to augment coarse traditional simulations with fine details [72, 44].

Another line of work uses data to fit a parameterized material model to observed deformations. This idea has been successfully applied to muscle-actuated biomechanical systems such as human faces [66, 129], learning the rest pose of an object in zero gravity [23], the design of soft robotics [59, 58], and motion planning with frictional contacts [43, 33]. Yang *et al.* [159] learn physical parameters for cloth by analysing the wrinkle patterns in video. While all of these methods learn physical parameters from data, our method is unique in requiring no template or other prior knowledge about object geometry to reconstruct and re-render novel deformations of an object.

Meshless Simulation. Meshless physics-based simulation emerged as a counter-part to traditional mesh-based methods [95] and is ideal for effects such as melting or fracture [95, 111]. These methods have been later extended to support oriented particles and skinning [96, 45, 84]. Another extension of point-based simulations consists in incorporating a background Eulerian grid, which enables more efficient simulation of fluid-like phenomena [131, 64].

4.2 Method

Our method provides an end-to-end solution for constructing a realistic virtual object, which is consist of several different parts, including capture, 4D reconstruction, material parameter learning, and re-rendering, and the overall

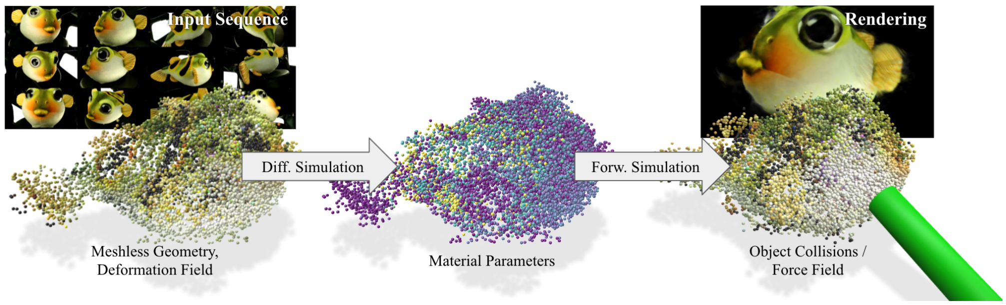

Figure 4.1: *Method overview.* We use a multi-view capture system to record objects deforming under the influence of external forces. Our method reconstructs a meshless geometry and deformation field from these sequences. Using a differentiable simulator, we optimize the material parameters of the objects to match the observations. These parameters allow us to find novel, plausible object configurations in response to new forces fields or collision constraints due to user interactions. Finally, we re-render the deformed state.

pipeline is shown in fig 4.1.

4.2.1 Capture

To create a physically accurate representation of an object, we first need to record visual data of its deformation under known physical forces. For recording, we use a static multi-view camera setup consisting of 19 OpenCV AI-Kit Depth (OAK-D) cameras[2], each containing an RGB and two grey-scale cameras (note that VEO does not use the stereo camera data to infer classical pairwise stereo depth). They represent an affordable, yet surprisingly powerful solution for volumetric capture. In particular, their on-board H265 encoding capability facilitates handling the amount of data produced during recording

[2] `https://store.opencv.ai/products/oak-d`

(5.12GB/s uncompressed). For details on temporal synchronization of the cameras, please see the supplemental material. Since the cameras lack a lens system with zoom capabilities, we keep them close to the object to optimize the pixel coverage and re-configure the system depending on object size. The maximum capture volume has a size of roughly 30cm^3. We put a black sheet around it to create a dark background with the exception of five stage lights that create a uniform lighting environment. A visualization of the camera layout and capture system can be found in the supplementary material.

In addition to the images, we also need to record force fields on the object surface. This raises a problem: if a prop is used to exert force on the capture subject, the prop becomes an occluder that interferes with photometric reconstruction. We solved this problem when capturing our **PLUSH** dataset by actuating the object using transparent fishing line and a compressed air stream; see Sec. 4.3.1 for further details.

4.2.2 4D Reconstruction

Given the captured video of an object deforming under external forces, we need 4D reconstruction to supply a temporally-coherent point cloud that can be used to learn the object material properties. To that end, we use NR-NeRF [140], which extends the static reconstruction method NeRF [94] to the temporal domain. NeRF learns a volumetric scene representation: a coordinate-based Multi-Layer Perceptron (MLP) $\mathbf{v}(\mathbf{x}) = (o, \mathbf{c})$ that regresses geometry (opacity $o(\mathbf{x}) \in \mathbb{R}$) and appearance (RGB color $\mathbf{c}(\mathbf{x}) \in \mathbb{R}^3$) at each point $\mathbf{x}$ in 3D space. At training time, the weights of $\mathbf{v}$ are optimized through 2D supervision by RGB images with known camera parameters: for a given pixel of an input image, the camera parameters allow us to trace the corresponding ray $\mathbf{r}(s)$ through 3D space. We then sample the NeRF at $|S|$ points $\{\mathbf{r}(s) \in \mathbb{R}^3\}_{s \in S}$ along the ray, and use a volumetric rendering equation to accumulate the samples front-to-back via weighted averaging: $\tilde{\mathbf{c}} = \sum_{s \in S} \alpha_s \mathbf{c}(\mathbf{r}(s))$ (*i.e.*, alpha blending with alpha values $\{\alpha_s \in \mathbb{R}\}_s$ derived from the opacities $\{o_s\}_s$). A reconstruction loss encourages the resulting RGB value $\tilde{\mathbf{c}}$ to be similar to the RGB value of the input pixel.

On top of the static geometry and appearance representation $\mathbf{v}$ (the *canonical model*), NR-NeRF models deformations explicitly via a jointly learned ray-bending MLP $\mathbf{b}(\mathbf{x}, \mathbf{l}_t) = \mathbf{d}$ that regresses a 3D offset $\mathbf{d}$ for each point in space at time t. ($\mathbf{l}_t$ is an auto-decoded latent code that conditions $\mathbf{b}$ on the deformation at time t.) When rendering a pixel at time t with NR-NeRF, $\mathbf{b}$ is queried for each sample $\mathbf{r}(s)$ on the ray in order to deform it into the canonical model: $(o, \mathbf{c}) = \mathbf{v}\left[\mathbf{r}(s) + \mathbf{b}(\mathbf{r}(s), \mathbf{l}_t)\right]$. Unlike NR-NeRF's monocular setting, we have a multi-view capture setup. We thus disable the regularization losses of NR-NeRF and only use its reconstruction loss.

Extensions. We improve NR-NeRF in several ways to adapt it to our setting. The input videos contain background, which we do not want to reconstruct. We obtain foreground segmentations for all input images via image matting [78] together with a hard brightness threshold. During training, we use a background loss $L_{background}$ to discourage geometry along rays of background pixels. When later extracting point clouds, we need opaque samples on the inside of the object as well. However, we find that $L_{background}$ leads the canonical model to prefer empty space even inside the object. We counteract this effect with a density loss $L_{density}$ that raises the opacity of point samples of a foreground ray that are 'behind' the surface, while emptying out the space in front of the surface with $L_{foreground}$. During training, we first build a canonical representation by pretraining the canonical model on a few frames and subsequently using it to reconstruct all images. Our capture setup not only provides RGB streams but also grey-scale images. We use these for supervision as well. In practice, we use a custom weighted combination of these techniques for each sequence to get the best reconstruction.

Point Cloud Extraction In order to extract a temporally-consistent point cloud from this reconstruction, we require a forward deformation model, which warps from the canonical model to the deformed state at time t. However, NR-NeRF's deformation model $\mathbf{b}$ is a backward warping model: it deforms each deformed state into the canonical model. We therefore jointly train a coordinate-based MLP $\mathbf{w}$ to approximate the inverse of $\mathbf{b}$. After training, we need to convert the reconstruction from its continuous MLP format into an explicit point cloud. To achieve that, we cast rays from all input cameras and extract points from the canonical model that are at or behind the surface and whose opacity exceeds a threshold. These points can then be deformed from the canonical model into the deformed state at time t via $\mathbf{w}$. See the supplemental material for further details. We thus obtain a 4D reconstruction in the form of a 3D point cloud's evolving point positions $\{P_t\}_t$, which are in correspondence across time. To keep the computational cost of the subsequent reconstruction steps feasible, we downsample the point cloud to 9-15k points if necessary.

4.2.3 Learning Material Parameters

Before we can simulate novel interactions with a captured object, we need to infer its physical behavior. Given that we have no prior knowledge of the object, we make several simplifying assumptions about its mechanics, with an eye towards minimizing the complexity of the physical model while also remaining flexible enough to capture heterogeneous objects built from multiple materials.

First, we assume a *spatially varying, isotropic nonlinear Neo-Hookean material model* for the object. Neo-Hookean elasticity well-approximates the behavior of many real-world materials, including rubber and many types of plastic, and is popular in computer graphics applications because its nonlinear stress-strain relationship guarantees that no part of the object can invert to have negative volume, even if the object is subjected to arbitrary large and nonlinear deformations. Finally, Neo-Hookean elasticity admits a simple parameterization: a pair of Lamé parameters $(\mu_i, \lambda_i) \in \mathbb{R}^2$ at each point i of the point cloud P.

Second, we assume that the object deforms *quasistatically* over time: that at each point in time, the internal elastic forces exactly balance gravity and applied external forces. The quasistatic assumption greatly simplifies learning material parameters, and is valid so long as inertial forces in the captured video sequences are negligible (or equivalently, so long as external forces change sufficiently slowly over time that there is no secondary motion, which is true for the air stream and string actuations in our **PLUSH** dataset).

Overview. We first formulate a differentiable, mesh-free *forward* physical simulator that is tailored to work directly with the (potentially noisy) reconstructed point cloud. This forward simulator maps from the point cloud P_0 of the object in its *reference pose* (where it is subject to no external forces besides gravity), an assignment of Lamé parameters to every point, and an assignment of an external force $\mathbf{f}_i \in \mathbb{R}^3$ to each point on the object surface, to the deformed position $\mathbf{y}_i \in \mathbb{R}^3$ of every point in the point cloud after the object equilibrates against the applied forces.

Next, we learn the Lamé parameters that match the object's observed behavior by minimizing a loss function $\mathbf{L}$ that sums, over all times t, the distance between $\mathbf{y}_i$ and the corresponding target position of the point in the 4D point cloud P_t.

Quasistatic Simulation. To compute the equilibrium positions $\mathbf{y}_i$ of the points in P for given external loads and material parameters, we solve the variational problem

$$\arg\min_{\mathbf{y}} \mathbf{E}(\mathbf{y}), \tag{4.1}$$

where $\mathbf{E}$ is the total energy of the physical system, capturing both the elastic energy of deformation as well as work done on the system by external forces. In what follows, we derive the expression for $\mathbf{E}$, and discuss how to solve Eq. 4.1.

Following Müller *et al.* [95], we adopt a mesh-free, point-based discretization of elasticity to perform forward simulation. For every point $\mathbf{x}_i$ in the reference point cloud P_0, we define a neighborhood $\mathcal{N}_i$ containing the 6 nearest neighbors of $\mathbf{x}_i$ in P_0. For any given set of deformed positions $\mathbf{y}_j$ of the points in $\mathcal{N}_i$, we estimate strain within the neighborhood in the least-squares sense. More specifically, the local material deformation gradient $\mathbf{F}_i \in \mathbb{R}^3$ maps the neighborhood $\mathcal{N}_i$ from the reference to the deformed state:

$$\mathbf{F}_i(\mathbf{x}_i - \mathbf{x}_j) \approx \mathbf{y}_i - \mathbf{y}_j \quad \forall \mathbf{x}_j \in \mathcal{N}_i. \tag{4.2}$$

For neighborhoods larger than three, Eq. 4.2 is over-determined, and we hence solve for $\mathbf{F}_i$ in the least-squares sense, yielding the closed-form solution:

$$\mathbf{F}_i = \mathbf{Y}_i \mathbf{W}_i \mathbf{X}_i^T (\mathbf{X}_i \mathbf{W}_i \mathbf{X}_i^T)^{-1}, \tag{4.3}$$

where the j-th column of $\mathbf{X}_i$ and $\mathbf{Y}_i$ are $\mathbf{x}_i - \mathbf{x}_j$ and $\mathbf{y}_i - \mathbf{y}_j$, respectively, and $\mathbf{W}_i$ is a diagonal matrix of weights depending on the distance from $\mathbf{x}_j$ to $\mathbf{x}_i$ [95].

The elastic energy of the object can be computed from the classic Neo-Hookean energy density [106]:

$$\Psi_{NH}^i = \frac{\mu_i}{2}(I_c - 3) - \mu_i \log J + \frac{\lambda_i}{2}(J-1)^2, \tag{4.4}$$

where I_c is the trace of the right Cauchy-Green tensor $\mathbf{F}_i^T\mathbf{F}_i$, and J is the determinant of $\mathbf{F}_i$. μ_i and λ_i are the Lamé parameters assigned to point i. The total elastic energy is then:

$$\mathbf{E}_{NH} = \sum_i V_i \Psi^i_{NH}, \tag{4.5}$$

where $V_i \in \mathbb{R}$ approximates the volume of $\mathcal{N}_i$.

We also need to include the virtual work done by the external force field to Eq. 4.1:

$$\mathbf{E}_W = \sum_i \mathbf{f}_i \cdot \mathbf{y}_i, \tag{4.6}$$

where $\mathbf{f_i}$ is the force applied to point i (the force of the air stream on the boundary). If we measured the tension in the fishing lines, we could also include the forces they exert on the object in Eq. 4.6. But since a fishing line is effectively inextensible relative to the object we are reconstructing, we instead incorporate the fishing lines as soft constraints on the positions of the points $Q \subset P$ attached to the lines: we assume that at time t, points in Q should match their observed positions in P_t, and formulate an attraction energy:

$$\mathbf{E}_A = \alpha \sum_{q \in Q} \|\mathbf{y}_q - \mathbf{x}^*_q\|^2, \tag{4.7}$$

where $\mathbf{x}^*_q$ is the position of the point corresponding to $\mathbf{y}_q$ in P_t, and α is a large penalty constant. We found that this soft constraint formulation works better in practice than alternatives such as enforcing $\mathbf{y}_q = \mathbf{x}^*_q$ as a hard constraint; see the supplemental material for more discussion.

The total energy in Eq. 4.1 is thus $\mathbf{E} = \mathbf{E}_{NH} + \mathbf{E}_W + \mathbf{E}_A$, which we minimize using Newton's method. Since Newton's method can fail when the Hessian $\mathbf{H}$ of $\mathbf{E}$ is not positive-definite, we perform a per-neighborhood eigen-decomposition of $\mathbf{H}$ and replace all eigenvalues that are smaller than a threshold $\epsilon > 0$ with ϵ; note that this is a well-known technique to improve robustness of physical simulations [136]. We also make use of a line search to ensure stability and handling of position constraints at points where the capture subject touches the ground; see the supplemental material for further implementation details.

Material Reconstruction. Given the 4D point cloud P_t and forces acting on the object $\{\mathbf{f}_i\}_i$, we use our forward simulator to learn the Lamé parameters that best explain the observed deformations. More specifically, at each time t we define the loss:

$$\mathbf{L}_t = \sum_{i \in \partial\Omega} \|\mathbf{y}_{t,i} - \mathbf{x}^*_{t,i}\|^2 \tag{4.8}$$

where $\mathbf{x}^*_{t,i}$ is the position of point i in P_t, and $\mathbf{y}_{t,i}$ is the output of the forward simulation. We use an ℓ_2 loss to penalize outliers strongly.

We choose a training subsequence T of 20-50 frames from the input where the impact of the air stream roughly covers the surface so that we have some reference for each part of the object, and compute the desired Lamé parameters by minimizing the sum of the loss over all $t \in T$ using the gradient-based Adam optimizer [69]:

$$\mu^*, \lambda^* = \arg\min_{\mu,\lambda} \sum_{t \in T} \mathbf{L}_t. \tag{4.9}$$

It is not trivial to back-propagate through the Newton solve for $\mathbf{y}_{t,i}$, even if we ignore the line search and assume a fixed number of Newton iterations K. The gradient of $\mathbf{y}$ with respect to the Lamé parameters (μ for instance) can be computed using the chain rule:

$$\frac{\partial \mathbf{L}}{\partial \mu} = \frac{\partial \mathbf{L}}{\partial \mathbf{y}^K} \frac{\partial \mathbf{y}^K}{\partial \mu}, \tag{4.10}$$

and, for any $1 \leq k \leq K$,

$$\begin{aligned} \frac{\partial \mathbf{y}^k}{\partial \mu} = \frac{\partial \mathbf{y}_{k-1}}{\partial \mu} &- \left(\frac{\partial \mathbf{H}_{k-1}^{-1}}{\partial \mu} + \frac{\partial \mathbf{H}_{k-1}^{-1}}{\partial \mathbf{y}_{k-1}} \frac{\partial \mathbf{y}_{k-1}}{\partial \mu} \right) \nabla \mathbf{E}_{k-1} \\ &- \mathbf{H}_{k-1}^{-1} \left(\frac{\partial \nabla \mathbf{E}_{k-1}}{\partial \mu} + \frac{\partial \nabla \mathbf{E}_{k-1}}{\partial \mathbf{y}_{k-1}} \frac{\partial \mathbf{y}_{k-1}}{\partial \mu} \right). \end{aligned} \tag{4.11}$$

To avoid an exponentially-large expression tree, we approximate the derivative of the kth Newton iterate $\mathbf{y}^k$ by neglecting the higher-order derivative of the Hessian and of the gradient of the energy with respect to the previous position update:

$$\frac{\partial \mathbf{y}^k}{\partial \mu} \approx \frac{\partial \mathbf{y}_{k-1}}{\partial \mu} - \frac{\partial \mathbf{H}_{k-1}^{-1}}{\partial \mu} \nabla \mathbf{E}_{k-1} - \mathbf{H}_{k-1}^{-1} \frac{\partial \nabla \mathbf{E}_{k-1}}{\partial \mu}$$

Although it is not guaranteed that the higher-order terms are always negligible, this approximation provides a sufficiently high-quality descent direction for all examples we tested. To improve performance and to capture hysteresis in cases where $\mathbf{E}$ has multiple local minima at some times t, we warm-start the Newton optimization at time t using the solution from time $t-1$. See the supplementary material for more details.

4.2.4 Novel Interactions

Given a reconstructed VEO, we can use the same physical simulator used for material inference to re-simulate the captured object subject to novel interactions. New force fields can easily be introduced by modifying $\mathbf{f}_i$ in the energy $\mathbf{E}_W$. Other possible interactions include changing the direction of gravity, adding contact forces to allow multiple objects to mutually interact, or to allow manipulation of the object using mixed-reality tools, etc.

We demonstrate the feasibility of re-simulating novel interactions by implementing a simple penalty energy to handle contact between a VEO and a secondary object, represented implicitly as a signed distance field $d : \mathbb{R}^3 \to \mathbb{R}$. The penalty energy is given by:

$$\Psi_c(\mathbf{y}) = \begin{cases} \alpha_c d(\mathbf{y})^2 & \text{if } d(\mathbf{y}) < 0 \\ 0 & \text{otherwise,} \end{cases} \tag{4.12}$$

$$\mathbf{E}_c = \sum_i V_i \Psi_c(\mathbf{y}_i), \tag{4.13}$$

where α_c is chosen large enough to prevent visually-noticeable penetration of the VEO by the secondary object.

4.2.5 Rendering

We are able to interact freely with the VEO in a physically plausible manner. Hence, we can close the full loop and realistically render the results of simulated novel interactions using neural radiance fields. While we used $\mathbf{b}$ for deformations during the reconstruction, we are now given a new deformed state induced by a discrete point cloud: a canonical reference point cloud $P_0 =$

$\{\mathbf{x}_s^0\}_s$ and its deformed version $S_d = \{\mathbf{y}_s^d\}_s$. We need to obtain a continuous backward-warping field from that point cloud in order to replace $\mathbf{b}$, which bends straight rays into the canonical model. To that end, we interpolate the deformation offsets $\mathbf{d}_s^b = \mathbf{x}_s^0 - \mathbf{y}_s^d$ at a 3D sample point $\mathbf{p}^d$ in deformed space using inverse distance weighting (IDW):

$$\mathbf{p}^c = \mathbf{p}^d + \sum_{s \in \mathcal{N}} \frac{w_s}{\sum_{s' \in \mathcal{N}} w_{s'}} \mathbf{d}_s^b, \tag{4.14}$$

where $\mathcal{N}$ are the $K = 5$ nearest neighbors of $\mathbf{p}^d$ in S_d, and $w_s = w'_s - \min_{s' \in \mathcal{N}} w'_{s'}$ with $w'_s = \|\mathbf{p}^d - \mathbf{y}_s^d\|^{-1}$. We can then sample the canonical model at $\mathbf{p}^c$ as before: $(o, \mathbf{c}) = \mathbf{v}(\mathbf{p}^c)$. To remove spurious geometry that o might show, we set $o(\mathbf{x}) = 0$ for $\mathbf{x}$ that are further than some threshold from S_d. Thus, we can now bend straight rays into the canonical model and render the interactively deformed state of the object in a realistic fashion.

When needed, we can upsample the point cloud from the simulation to make it denser. Unlike for rendering, we need to consider forward warping for this case.

4.3 Results

4.3.1 Dataset

The **PLUSH** dataset consists of 12 soft items encountered in everyday life (see Fig. 4.2): a pillow, a sponge, and various plush toys. We chose items that are composed of soft (and in some cases, heterogeneous) material, complex geometry, and rich texture and color to enable successful background

Figure 4.2: *The **PLUSH** dataset* consists of 12 items from everyday life: a pillow, a sponge and several plushies. ‖ indicates that we recorded extremity motion for the object, * indicates that the recording has significant second order motion. We additionally provide the mass and recording duration for each object. **Lower right:** *Lamé parameter visualizations for Baby Alien and Pony.* Colors tending towards purple show a softer region, colors tending towards green and yellow a harder region. Our method clearly identifies different material properties on the objects, for example the arms and ears for the Baby Alien, and the mane and tail of the Pony.

subtraction, 4D reconstruction and tracking. We provide purchase links for all objects in the supplementary material to enable other researchers to reproduce our experiments. Our strategy for applying external forces is based on the observation that our chosen objects consist of *bulk volumes* (such as the body of a plush toy) along with *flexible extremities* (ears and fingers of the toy). We move object extremities by using transparent fishing line, and we use a stream of compressed air to exert force on bulk volumes. The nozzle position and stream direction must be tracked during video capture to provide the direction and magnitude of forces acting on the object at every point in time. Of the 19 cameras in our capture rig, we use three to track the nozzle using an attached ArUco marker [42, 120]. Using this system, we generate multi-part video sequences for each capture subject, where we sequentially actuate

Object	average (mm)	95% (mm)	max (mm)
Baby Alien	3.8	14.4	29.3
Gray Fish	1.1	6.6	18.5
Leaf	0.4	1.1	9.8
Gray Mr. Seal	0.4	1.9	171.9
Pillow	1.5	7.8	18.35
Gray Dog	1.7	7.5	28.8
Sponge	0.2	1.8	15.8
Gray Dino Rainbow	4.0	14.6	171.4
Dino Blue	5.5	56.0	105.8
Gray Dino Green	6.2	68.4	132.0
Pony	21.1	164.3	204.9
Gray Serpentine	7.5	43.1	94.7
Average*	2.5	18.0	70.2
Average	4.4	32.3	83.5

Table 4.1: *ℓ_2 distance of simulated point clouds compared with reconstructed point clouds on the test set.* We record the average distance per point per frame, the 95th percentile of average point distances of all frames, and the maximum distance of all points. Average* excludes the data from Pony and Serpentine.

the fishing lines (when applicable) followed by sweeping the air stream over the object. See the supplemental material for details of our methodology for applying and recording external forces. We record between 32s and 67s of video for each object, at a frame rate of 40FPS.

4.3.2 Virtual Elastic Objects

For each of the 12 examples, we create a VEO using 20-50 frames from the reconstruction and evaluate on the remaining 500-1500 frames. We sample $100k$ points from NR-NeRF and down-sample to $10k$–$30k$ points for simulation, depending on the object. We use the ℓ_2 distance between the surface points of the VEO to the reconstructed point cloud from the captured data to evaluate the quality of the reconstructed parameters. For all examples except for the Baby Alien we use the external force field data obtained using the air stream. For the Baby Alien, we specifically use the arm and ear motion to demonstrate the versatility of our method in this scenario. We present the results in Tab. 4.1.

The error is relatively small for all objects, which shows that our method is applicable to objects with different geometries, and can learn the corresponding material parameters even for heterogeneous objects. Larger errors are observed for objects with a thin and tall component (see the last 4 rows of the table). This error is largely caused by tracking inaccuracies of the nozzle: even slight inaccuracies can cause large errors when, for example, the neck of the dinosaur moves while the recorded air stream direction does not, or barely, touch the object.

Inhomogeneous Material. An important feature of our method is that it can identify different material parameters for different parts of the object (c.t. Fig. 4.2, lower right). This is crucial for building a detailed physics model with no prior knowledge of the object. Even more, our method can reliably learn 'effective' softness of the material even in places with unreliable tracking, for example thin geometrical structures close to joints. In case of Baby Alien, our method learns that the ears and arms are softer compared to the other body parts; the mane and tail of the Pony are softer, even though these regions are very hard to track. Both reconstructions match the properties of their real

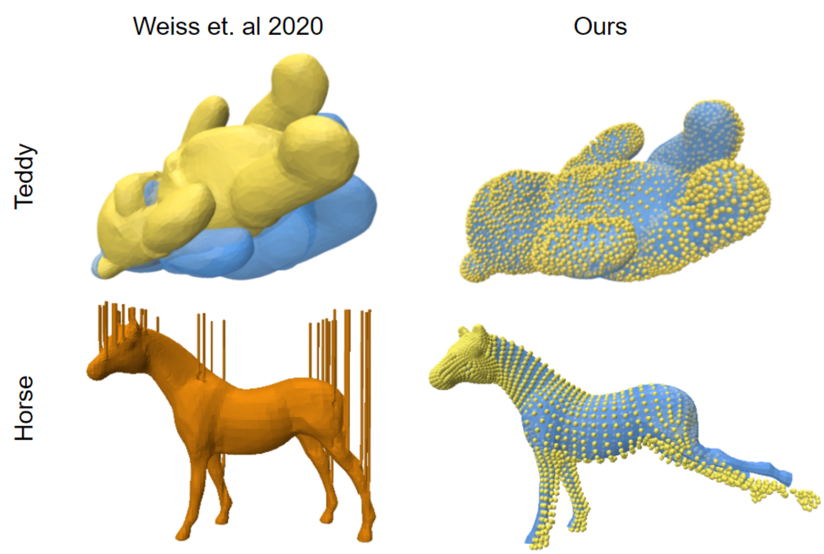

Figure 4.3: *Comparison with Weiss et al.* Comparison on two examples. Blue meshes are the ground truth, simulation results are shown in yellow. Weiss *et al.* fails at reconstructing the horse (orange); our heterogeneous model produces overall more reliable results.

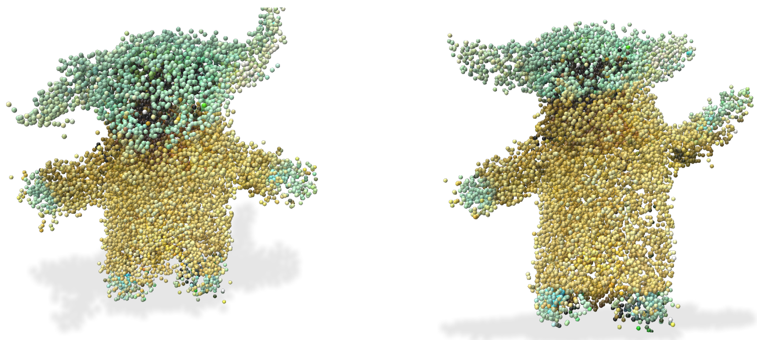

Figure 4.4: *Simulation of Baby Alien in poses unseen in the dataset.* Using the material model and simulator our method generalizes well to these asymmetric postures for ears and arms; we only observe symmetric forward and backward motions during training.

counterparts.

We compare our method with the mesh-based work of Weiss et al. [155] (which requires a mesh template). We use the teddy mesh from their paper and simulate it with a heterogeneous material under gravity. We provide 200 depth images and the template to [155], and use the equilibrium point cloud under gravity as input to our method. We then compare both under a novel gravity force; see Fig. 4.3. Our method is able to estimate the material parameters much better, due to the use of a heterogeneous material model as opposed to the homogeneous model in [155]. We also use a horse model in the same setup. Weiss et al. fail at reconstructing the parts with more detailed geometry (behavior has been confirmed as correct by the authors of [155]) and cannot simulate the object due to the severe artifacts.

Generalization to Novel Poses. The strength of the underlying physics simulator is the ability to generalize to scenarios that are not encountered

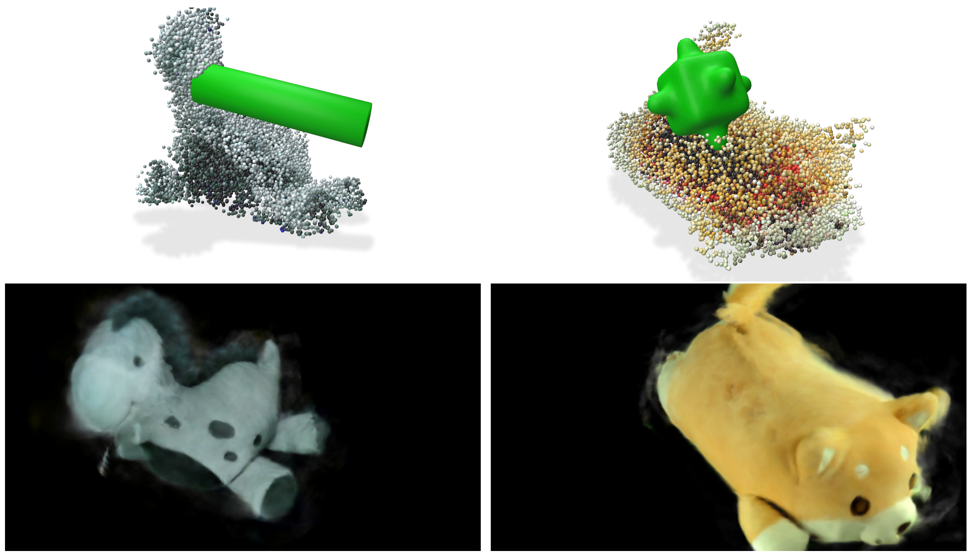

Figure 4.5: *Rendering of the Dino Blue and Dog VEOs during interactions with secondary objects.* The dinosaur neck bends correctly, and dents are forming on the Dog's back.

in the training set. We show different simulated poses of the Baby Alien in Fig. 4.4, such as pulling the ears in opposite directions, and moving just one single arm. This deformation is particularly challenging for purely data-driven methods since both ears and arms only move synchronously in the training data.

Runtime. The upper bound of the run time required for each step of our pipeline is $20h$ for reconstruction on four NVIDIA V100 GPUs, $13h$ for learning the parameters on a AMD Ryzen 5 1600 six-core processor, $5min$ for running a new simulation, and $10min$ for rendering a frame.

Interaction with Virtual Objects. The physical model of the object enables interactions with all kinds of different virtual items. Fig. 4.5 shows the one-way coupled interaction of the learned elastic objects with other virtual items.

Rendering. Our pipeline ends with re-rendering an object under novel interactions not seen during training. Fig. 4.5 contains renderings of the Dino Blue and Dog objects, including interactions with two virtual objects. For additional qualitative results, we refer to the supplemental video. Tab. 4.2 contains quantitative results, where we compare the renderings obtained from the reconstructed point clouds (which are used for supervision when learning the material parameters) and the simulated point clouds.

4.4 Limitations

Artifacts. Unlike in the simple static setting, we require reliable long-term correspondences on top of the visual reconstruction, which for NeRF only [108, 115, 140] provide. They differ mainly in how they handle the respective monocular problem setting. Our reconstruction method can be seen as a multi-view extension of any of them to our problem setting. Several remaining artifacts arise from the setup. Due to the sparse camera setup (16 cameras for 360 degree coverage), we found NeRF unable to reconstruct viewpoint dependent effects, leading to artifacts around specular regions like eyes. Furthermore, the air compressor leads to quickly oscillating surfaces (*e.g.*, the fins of the fish), which pose a challenge for reconstruction and material parameter estimation, and impacts calibration. These issues impact the extracted point clouds as well as the final renderings (artifacts visible in Fig. 4.5). The physical simu-

	Simulated						Reconstructed					
	Not Masked			Masked			Not Masked			Masked		
Object	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Baby Alien	18.40	0.734	0.255	21.17	0.840	0.174	18.75	0.747	0.249	21.92	0.853	0.167
Gray Fish	19.75	0.692	0.239	22.55	0.808	0.173	20.03	0.701	0.235	22.96	0.818	0.169
Leaf	25.14	0.901	0.091	27.32	0.935	0.065	25.19	0.901	0.091	27.37	0.935	0.065
Gray Mr. Seal	20.61	0.697	0.240	24.03	0.801	0.180	20.65	0.698	0.239	24.11	0.802	0.180
Pillow	21.45	0.743	0.223	23.18	0.806	0.174	21.92	0.760	0.218	23.84	0.823	0.169
Gray Dog	18.98	0.751	0.206	24.68	0.904	0.104	19.05	0.757	0.203	25.24	0.912	0.100
Sponge	21.94	0.846	0.130	26.99	0.925	0.070	21.92	0.846	0.130	27.01	0.925	0.070
Gray Dino Rainbow	18.64	0.754	0.302	23.87	0.839	0.232	20.22	0.778	0.281	26.21	0.859	0.213
Dino Blue	18.48	0.702	0.244	20.70	0.848	0.160	19.56	0.726	0.227	22.06	0.871	0.143
Gray Dino Green	18.94	0.779	0.190	21.49	0.863	0.135	20.46	0.794	0.180	23.59	0.879	0.121
Pony	16.54	0.758	0.245	19.20	0.859	0.163	19.31	0.798	0.200	24.65	0.906	0.108
Gray Serpentine	18.22	0.798	0.181	21.39	0.903	0.111	19.95	0.813	0.162	23.14	0.916	0.091
Average*	20.23	0.760	0.212	23.60	0.857	0.145	20.78	0.771	0.205	24.43	0.868	0.140
Average	19.76	0.763	0.212	23.05	0.861	0.147	20.58	0.777	0.201	24.34	0.875	0.133

Table 4.2: *Rendering evaluation.* We report the classic error metrics PSNR and SSIM [151] (-1 to $+1$), where higher is better for both, and the learned perceptual metric LPIPS [166] (0 is best). We use deformed point clouds to render deformed states of the canonical model, see Sec. 4.2.5. We use both, the point cloud P_t that the reconstruction (Sec. 4.2.2) provides directly ('Reconstructed') or the point cloud that the simulator provides after learning the material parameters (Sec. 4.2.3, 'Simulated'). We report two versions: we either apply the segmentation masks of the input images to the rendered image to remove all artifacts that spill over onto the background ('Masked') or we do not ('Not Masked'). Note, that the values on the reconstructed point cloud are a (soft) upper bound for what the simulator can achieve. The simulated results are close the reconstructed results, demonstrating that the learned material parameters yield deformation fields that allow to re-render the object as well as the reconstruction can.

lator turned out to be remarkably robust towards noise and can run with any point cloud with temporal correspondences.

Known Forces. The simulator requires the forces impacting the object during capture to be known. This limits the variety of forces that can be applied and hence the kind of objects that are compatible with the presented method. We expect an extension handling unknown forces an exciting direction for future work. Finding good force priors could be a viable approach in this direction.

Bibliography

[1] Iggesund Paperboard AB. *Paperboard reference manual.* Iggesund Paperboard AB, 1993.

[2] Antonio Agudo, J. M. M. Montiel, Lourdes de Agapito, and Begoña Calvo. Online dense non-rigid 3d shape and camera motion recovery. In *BMVC*, 2014.

[3] Antonio Agudo, Francesc Moreno-Noguer, Begoña Calvo, and José María Martínez Montiel. Sequential non-rigid structure from motion using physical priors. *IEEE transactions on pattern analysis and machine intelligence*, 38(5):979–994, 2015.

[4] Benjamin Attal, Eliot Laidlaw, Aaron Gokaslan, Changil Kim, Christian Richardt, James Tompkin, and Matthew O'Toole. Törf: Time-of-flight radiance fields for dynamic scene view synthesis. In *Neural Information Processing Systems (NeurIPS)*, 2021.

[5] Kosala Bandara and Fehmi Cirak. Isogeometric shape optimisation of shell structures using multiresolution subdivision surfaces. *Computer-Aided Design*, 95(Supplement C):62 – 71, 2018.

[6] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and*

Interactive Techniques, SIGGRAPH '98, pages 43–54, New York, NY, USA, 1998. ACM.

[7] Jernej Barbič, Fun Shing Sin, and Daniel Schroeder. Vega FEM Library, 2012. http://www.jernejbarbic.com/vega.

[8] Klaus Jürgen Bathe, Eduardo Dvorkin, and Lee W. Ho. Our discrete-kirchhoff and isoparametric shell elements for nonlinear analysis - an assessment. *Computers & Structures*, 16(1):89 – 98, 1983.

[9] Jean-Louis Batoz, Klaus-Jürgen Bathe, and Lee-Wing Ho. A study of three-node triangular plate bending elements. *International Journal for Numerical Methods in Engineering*, 15(12):1771–1812, 1980.

[10] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light- and time-image interpolation. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2020)*, 39(6), 2020.

[11] D.J. Benson, Y. Bazilevs, M.C. Hsu, and T.J.R. Hughes. Isogeometric shell analysis: The reissner-mindlin shell. *Comput. Methods Appl. Mech. Eng.*, 199(5):276 – 289, 2010.

[12] Miklos Bergou, Max Wardetzky, David Harmon, Denis Zorin, and Eitan Grinspun. A quadratic bending model for inextensible surfaces. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*,

SGP '06, pages 227–230, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[13] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020.

[14] Alexander I. Bobenko, Helmut Pottmann, and Johannes Wallner. A curvature theory for discrete surfaces based on mesh parallelity. *Mathematische Annalen*, 348(1):1–24, Sep 2010.

[15] Alexander I. Bobenko and Peter Schröder. Discrete willmore flow. In *Proceedings of the Third Eurographics Symposium on Geometry Processing*, SGP '05, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.

[16] Sergey Bochkanov. Alglib, 1999.

[17] B. Boroomand and B. Khalilian. On using linear elements in incompressible plane strain problems: a simple edge based approach for triangles. *International Journal for Numerical Methods in Engineering*, 61(10):1710–1740, 2004.

[18] Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. Projective dynamics: fusing constraint projections for fast simulation. *ACM Trans. Graph.*, 33:154:1–154:11, 2014.

[19] R. Bridson, S. Marino, and R. Fedkiw. Simulation of clothing with folds and wrinkles. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 28–36. Eurographics Association, 2003.

[20] Robert Bridson, Ronald Fedkiw, and John Anderson. Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Graph.*, 21(3):594–603, July 2002.

[21] Ming Chen and Kai Tang. A fully geometric approach for developable cloth deformation simulation. *The Visual Computer*, 26(6):853–863, Jun 2010.

[22] Wei Chen, Fei Zhu, Jing Zhao, Sheng Li, and Guoping Wang. Peridynamics-based fracture animation for elastoplastic solids. *Computer Graphics Forum*, 37(1):112–124, 2018.

[23] Xiang Chen, Changxi Zheng, Weiwei Xu, and Kun Zhou. An asymptotic numerical method for inverse elastic shape design. *ACM Transactions on Graphics*, 33(4), 2014.

[24] Yanqing Chen, Timothy A. Davis, William W. Hager, and Sivasankaran Rajamanickam. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. *ACM Trans. Math. Softw.*, 35(3):22:1–22:14, October 2008.

[25] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Computer Vision and Pattern Recognition*, pages 5939–5948, 2019.

[26] Shubhankar Roy Chowdhury, Pranesh Roy, Debasish Roy, and J N Reddy. A peridynamic theory for linear elastic shells. *International Journal of Solids and Structures*, 84:110 – 132, 2016.

[27] Jun Young Chung, Hunter King, and L. Mahadevan. Evaporative microclimate driven hygrometers and hygromotors. *EPL (Europhysics Letters)*, 107(6):64002, 2014.

[28] Philippe G. Ciarlet. *Theory of Shells, Volume 3 (Mathematical Elasticity)*. North Holland, 2000.

[29] Fehmi Cirak, Michael Ortiz, and Peter Schröder. Subdivision surfaces: a new paradigm for thin-shell finite-element analysis. *International Journal for Numerical Methods in Engineering*, 47(12):2039–2072, 2000.

[30] E.A. de Souza Neto, D. Perić, M. Dutko, and D.R.J. Owen. Design of simple low order finite elements for large strain analysis of nearly incompressible solids. *International Journal of Solids and Structures*, 33(20):3277 – 3296, 1996.

[31] Julien Dervaux and Martine Ben Amar. Morphogenesis of growing soft tissues. *Phys. Rev. Lett.*, 101:068101, Aug 2008.

[32] Dimitar Dinev, Tiantian Liu, Jing Li, Bernhard Thomaszewski, and Ladislav Kavan. Fepr: Fast energy projection for real-time simulation of deformable objects. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018.

[33] Tao Du, Kui Wu, Pingchuan Ma, Sebastien Wah, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. Diffpd: Differentiable projective dynamics with contact. *arXiv preprint arXiv:2101.05917*, 2021.

[34] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B. Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[35] E. Efrati, E. Sharon, and R. Kupferman. Elastic theory of unconstrained non-euclidean plates. *J. Mech. Phy. Solids*, 57(4):762 – 775, 2009.

[36] Efi Efrati, Eran Sharon, and Raz Kupferman. Buckling transition and boundary layer in non-euclidean plates. *Physical Review E*, 80(1), jul 2009.

[37] Elliot English and Robert Bridson. Animating developable surfaces using nonconforming elements. *ACM Trans. Graph.*, 27(3):66:1–66:5, August 2008.

[38] FaveCrafts. How to make roses from plastic spoons, 2012.

[39] Federico Fuentes, Brendan Keith, Leszek F. Demkowicz, and Patrick Le Tallec. Coupled variational formulations of linear elasticity and the DPG methodology. *J. Comput. Phys.*, 348:715–731, 2017.

[40] Lawson Fulton, Vismay Modi, David Duvenaud, David I. W. Levin, and Alec Jacobson. Latent-space dynamics for reduced deformable simulation. *Computer Graphics Forum*, 2019.

[41] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. *arXiv preprint arXiv:2105.06468*, 2021.

[42] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and R. Medina-Carnicer. Generation of fiducial marker dictionaries using Mixed Integer Linear Programming. *Pattern Recognition*, 51:481–491, mar 2016.

[43] Moritz Geilinger, David Hahn, Jonas Zehnder, Moritz Bächer, Bernhard Thomaszewski, and Stelian Coros. Add: Analytically differentiable dynamics for multi-body systems with frictional contact. *ACM Transactions on Graphics (TOG)*, 39(6), 2020.

[44] Zhenglin Geng, Daniel Johnson, and Ronald Fedkiw. Coercing machine learning to output physically accurate results. *Journal of Computational Physics*, 406:109099, Apr 2020.

[45] Benjamin Gilles, Guillaume Bousquet, Francois Faure, and Dinesh K Pai. Frame-based elastic models. *ACM transactions on graphics*

(TOG), 30(2):1–12, 2011.

[46] Yotam Gingold, Adrian Secord, Jefferson H. Han, Eitan Grinspun, and Denis Zorin. Poster: A discrete model for inelastic deformation of thin shells. In *ACM/Eurographics Symposium on Computer Animation '04*, 2004.

[47] A. Sydney Gladman, Elisabetta A. Matsumoto, Ralph G. Nuzzo, L. Mahadevan, and Jennifer A. Lewis. Biomimetic 4d printing. *Nature Materials*, 15:413–418, 2016.

[48] Rony Goldenthal, David Harmon, Raanan Fattal, Michel Bercovier, and Eitan Grinspun. Efficient simulation of inextensible cloth. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, 26(3):to appear, 2007.

[49] Alain Goriely and Martine Ben Amar. Differential growth and instability in elastic shells. *Phys. Rev. Lett.*, 94:198103, May 2005.

[50] Eitan Grinspun, Yotam Gingold, Jason Reisman, and Denis Zorin. Computing discrete shape operators on general meshes. *Computer Graphics Forum*, 25(3):547–556, 2006.

[51] Eitan Grinspun, Anil N. Hirani, Mathieu Desbrun, and Peter Schröder. Discrete shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 62–67. Eurographics Association, 2003.

[52] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Computer Vision and Pattern Recognition*, pages 216–224, 2018.

[53] Christophe Guberan. Hydro-fold, 2012.

[54] Ruslan Guseinov, Eder Miguel, and Bernd Bickel. Curveups: Shaping objects from flat plates with tension-actuated curvature. *ACM Trans. Graph.*, 36(4):64:1–64:12, July 2017.

[55] David Hahn, Pol Banzet, James M. Bern, and Stelian Coros. Real2sim: Visco-elastic parameter estimation from dynamic motion. *ACM Transactions on Graphics (TOG)*, 38(6), 2019.

[56] X. He, H. Wang, and E. Wu. Projective peridynamics for modeling versatile elastoplastic materials. *IEEE Transactions on Visualization and Computer Graphics*, 24(9):2589–2599, Sep. 2018.

[57] Daniel Holden, Bang Chi Duong, Sayantan Datta, and Derek Nowrouzezahrai. Subspace neural physics: Fast data-driven interactive simulation. In *Proceedings of the 18th annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, New York, NY, USA, 2019. Association for Computing Machinery.

[58] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable

programming for physical simulation, 2020.

[59] Yuanming Hu, Jiancheng Liu, Andrew Spielberg, Joshua B Tenenbaum, William T Freeman, Jiajun Wu, Daniela Rus, and Wojciech Matusik. Chainqueen: A real-time differentiable physical simulator for soft robotics. In *2019 International conference on robotics and automation (ICRA)*, pages 6265–6271. IEEE, 2019.

[60] Thomas J.R. Hughes and Wing Kam Liu. Nonlinear finite element analysis of shells: Part i. three-dimensional shells. *Computer Methods in Applied Mechanics and Engineering*, 26(3):331 – 362, 1981.

[61] Geoffrey Irving, Craig Schroeder, and Ronald Fedkiw. Volume conserving finite element simulations of deformable models. *ACM Trans. Graph.*, 26(3), July 2007.

[62] SoHyeon Jeong, Tae-hyeong Kim, and Chang-Hun Kim. Shrinkage, wrinkling and ablation of burning cloth and paper. *Vis. Comput.*, 27(6-8):417–427, June 2011.

[63] SoHyeon Jeong, Si-Hyung Park, and Chang-Hun Kim. Simulation of morphology changes in drying leaves. *Computer Graphics Forum*, 32(1):204–215, 2013.

[64] Chenfanfu Jiang, Theodore Gast, and Joseph Teran. Anisotropic elastoplasticity for cloth, knit and hair frictional contact. *ACM Transactions on Graphics (TOG)*, 36(4):1–14, 2017.

[65] Ning Jin, Wenlong Lu, Zhenglin Geng, and Ronald P. Fedkiw. Inequality cloth. In *SCA '17 Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 1–10, Los Angeles, California, 07 2017. ACM.

[66] Petr Kadlecek and Ladislav Kavan. Building accurate physics-based face models from data. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2(2):1–16, 2019.

[67] J. Kiendl, K.-U. Bletzinger, J. Linhard, and R. Wüchner. Isogeometric shell analysis with kirchhoff-love elements. *Computer Methods in Applied Mechanics and Engineering*, 198(49):3902 – 3914, 2009.

[68] J. Kim, J.A. Hanna, M. Byun, C.D. Santangelo, and R.C. Hayward. Designing responsive buckled surfaces by halftone gel lithography. *Science*, 335(6073):1201–1205, 2012.

[69] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations, ICLR*, 2015.

[70] Yael Klein, Efi Efrati, and Eran Sharon. Shaping of elastic sheets by prescription of non-euclidean metrics. *Science*, 315(5815):1116–1120, 2007.

[71] Suryansh Kumar, Anoop Cherian, Yuchao Dai, and Hongdong li. Scalable dense non-rigid structure-from-motion: A grassmannian perspec-

tive. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 254–263, 06 2018.

[72] Zorah Lähner, Daniel Cremers, and Tony Tung. Deepwrinkles: Accurate and realistic clothing modeling. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 698–715. Springer International Publishing, 2018.

[73] Caroline Larboulette, Pablo Quesada, and Olivier Dumas. Burning paper: Simulation at the fiber's level. In *Proceedings of Motion on Games*, MIG '13, pages 25:47–25:52, New York, NY, USA, 2013. ACM.

[74] J. A. Levine, A. W. Bargteil, C. Corsi, J. Tessendorf, and R. Geist. A peridynamic perspective on spring-mass fracture. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '14, pages 47–55. Eurographics Association, 2014.

[75] Pengbo Li and Paul G. Kry. Multi-layer skin simulation with adaptive constraints. In *Proceedings of the Seventh International Conference on Motion in Games*, MIG '14, pages 171–176, New York, NY, USA, 2014. ACM.

[76] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, and Zhaoyang Lv. Neural 3D Video Synthesis. 2021.

[77] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021.

[78] Shanchuan Lin, Andrey Ryabtsev, Soumyadip Sengupta, Brian Curless, Steve Seitz, and Ira Kemelmacher-Shlizerman. Real-Time High-Resolution Background Matting. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8762—-8771, dec 2021.

[79] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020.

[80] Shiguang Liu, Qiguang Liu, Tai An, Jizhou Sun, and Qunsheng Peng. Physically based simulation of thin-shell objects' burning. *The Visual Computer*, 25(5):687–696, May 2009.

[81] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, July 2019.

[82] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Trans. Graph.*, 40(4), July 2021.

[83] Frank Losasso, Geoffrey Irving, Eran Guendelman, and Ron Fedkiw. Melting and burning solids into liquids and gases. *IEEE Transactions on Visualization and Computer Graphics*, 12(3):343–352, May 2006.

[84] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014.

[85] E.H. Mansfield. *The Bending and Stretching of Plates*, volume 6. Pergamon, Amsterdam, The Netherlands, 1964.

[86] Sebastian Martin, Peter Kaufmann, Mario Botsch, Eitan Grinspun, and Markus Gross. Unified simulation of elastic rods, shells, and solids. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pages 39:1–39:10, New York, NY, USA, 2010. ACM.

[87] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021.

[88] Zeki Melek and John Keyser. Interactive simulation of burning objects. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, PG '03, pages 462–, Washington, DC, USA, 2003. IEEE Computer Society.

[89] Zeki Melek and John Keyser. Multi-representation interaction for physically based modeling. In *Proceedings of the 2005 ACM Symposium on Solid and Physical Modeling*, SPM '05, pages 187–196, New York, NY, USA, 2005. ACM.

[90] Zeki Melek and John Keyser. Driving object deformations from internal physical processes. In *Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling*, SPM '07, pages 51–59, New York, NY, USA, 2007. ACM.

[91] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[92] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. *Discrete Differential-Geometry Operators for Triangulated 2-Manifolds*, pages 35–57. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[93] ANDRO MIKELIĆ and MARY F. WHEELER. On the interface law between a deformable porous medium containing a viscous fluid and an elastic body. *Mathematical Models and Methods in Applied Sciences*, 22(11):1250031, 2012.

[94] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[95] M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa. Point based animation of elastic, plastic and melting objects. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '04, page 141–151, 2004.

[96] Matthias Müller and Nuttapong Chentanez. Solid simulation with oriented particles. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, New York, NY, USA, 2011. Association for Computing Machinery.

[97] Matthias Müller, Nuttapong Chentanez, Tae-Yong Kim, and Miles Macklin. Strain based dynamics. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '14, pages 149–157. Eurographics Association, 2014.

[98] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109 – 118, 2007.

[99] Rahul Narain, Tobias Pfaff, and James F. O'Brien. Folding and crumpling adaptive sheets. *ACM Transactions on Graphics*, 32(4):51:1–8, July 2013. Proceedings of ACM SIGGRAPH 2013, Anaheim.

[100] Rahul Narain, Armin Samii, and James F. O'Brien. Adaptive anisotropic remeshing for cloth simulation. *ACM Trans. Graph.*, 31(6):152:1–152:10, November 2012.

[101] P. Neff. A geometrically exact cosserat shell-model including size effects, avoiding degeneracy in the thin shell limit. part i: Formal dimensional reduction for elastic plates and existence of minimizers for positive cosserat couple modulus. *Continuum Mechanics and Thermodynamics*, 16(6):577–628, Oct 2004.

[102] E. A.de Souza Neto, F. M. Andrade Pires, and D. R. J. Owen. F-bar-based linear triangles and tetrahedra for finite strain analysis of nearly incompressible solids. part i: formulation and benchmarking. *International Journal for Numerical Methods in Engineering*, 62(3):353–383, 2005.

[103] Richard A. Newcombe, Dieter Fox, and Steven M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[104] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021.

[105] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5379–5389, 2019.

[106] R.W. Ogden. *Non-linear Elastic Deformations*. Ellis Horwood series in mathematics and its applications. E. Horwood, 1984.

[107] A. Ortiz-Bernardin, M.A. Puso, and N. Sukumar. Improved robustness for nearly-incompressible large deformation meshfree simulations on Delaunay tessellations. *Computer Methods in Applied Mechanics and Engineering*, 293:348 – 374, 2015.

[108] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021.

[109] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021.

[110] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[111] Mark Pauly, Richard Keiser, Bart Adams, Philip Dutré, Markus Gross, and Leonidas J. Guibas. Meshless animation of fracturing solids. *ACM Trans. Graph.*, 24(3):957–964, July 2005.

[112] Jesus Perez, Miguel A. Otaduy, and Bernhard Thomaszewski. Computational design and automated fabrication of kirchhoff-plateau surfaces. *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH)*, 36(4):62.1–62.12, 2017.

[113] F. Petronetto, A. Paiva, E. S. Helou, D. E. Stewart, and L. G. Nonato. Mesh-free discrete laplace-beltrami operator. *Comput. Graph. Forum*, 32(6):214–226, September 2013.

[114] Xavier Provot. Deformation constraints in a mass-spring model to describe rigid cloth behavior. *Graphics Interface*, 23(19):147–154, 1995.

[115] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[116] A. Quaglino. *Membrane locking in discrete shell theories.* PhD thesis, Georg-August-Universität Göttingen, 2012.

[117] A. Quaglino. A framework for creating low-order shell elements free of membrane locking. *International Journal for Numerical Methods in Engineering*, 108(1):55–75, 2016.

[118] Edoardo Remelli, Artem Lukoianov, Stephan Richter, Benoit Guillard, Timur Bagautdinov, Pierre Baque, and Pascal Fua. Meshsdf: Differentiable iso-surface extraction. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[119] Olivier Rémillard and Paul G. Kry. Embedded thin shells for wrinkle simulation. *ACM Trans. Graph.*, 32:50:1–50:8, 2013.

[120] Francisco J. Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. Speeded up detection of squared fiducial markers. *Image and Vision Computing*, 76:38–47, aug 2018.

[121] Jessica Rosenkrantz and Jesse Louis-Rosenberg. Nervous systems, 2018.

[122] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

[123] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[124] Eran Sharon and Efi Efrati. The mechanics of non-euclidean plates. *Soft Matter*, 6:5693–5704, 2010.

[125] S.A. Silling. Reformulation of elasticity theory for discontinuities and long-range forces. *Journal of the Mechanics and Physics of Solids*, 48(1):175 – 209, 2000.

[126] J.C. Simo and D.D. Fox. On a stress resultant geometrically exact shell model. part i: Formulation and optimal parametrization. *Computer Methods in Applied Mechanics and Engineering*, 72(3):267 – 304, 1989.

[127] Mélina Skouras, Bernhard Thomaszewski, Peter Kaufmann, Akash Garg, Bernd Bickel, Eitan Grinspun, and Markus Gross. Designing inflatable structures. *ACM Trans. Graph.*, 33(4):63:1–63:10, July 2014.

[128] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7495–7504, 2021.

[129] Sangeetha Grama Srinivasan, Qisi Wang, Junior Rojas, Gergely Klár, Ladislav Kavan, and Eftychios Sifakis. Learning active quasistatic physics-based models from data. *ACM Transactions on Graphics (TOG)*, 40(4), 2021.

[130] D. J. Steigmann. Tension-field theory. *Proceedings of the Royal Society of London A*, 429:141–173, 1990.

[131] Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A material point method for snow simulation. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013.

[132] Thomas Stumpp, Jonas Spillmann, Markus Becker, and Matthias Teschner. A Geometric Deformation Model for Stable Cloth Simulation. In Francois Faure and Matthias Teschner, editors, *Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2008)*, pages 1–8, Grenoble, France, 2008. The Eurographics Association.

[133] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021.

[134] John M. Sullivan. *Curvatures of Smooth and Discrete Surfaces*, pages 175–188. Birkhäuser Basel, Basel, 2008.

[135] Rasmus Tamstorf and Eitan Grinspun. Discrete bending forces and their jacobians. *Graphical Models*, 75(6):362 – 370, 2013.

[136] Joseph Teran, Eftychios Sifakis, Geoffrey Irving, and Ronald Fedkiw. Robust quasistatic finite elements and flesh simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 181–190, 2005.

[137] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. Nießner, R. Pandey, S. Fanello, G. Wetzstein, J.-Y. Zhu, C. Theobalt, M. Agrawala, E. Shechtman, D. B Goldman, and M. Zollhöfer. State of the Art on Neural Rendering. *Computer Graphics Forum (EG STAR 2020)*, 2020.

[138] Bernhard Thomaszewski, Simon Pabst, and Wolfgang Straßer. Continuum-based strain limiting. *Computer Graphics Forum*, 28(2):569–576, 2009.

[139] Maxime Tournier, Matthieu Nesme, Benjamin Gilles, and François Faure. Stable constrained dynamics. *ACM Trans. Graph.*, 34(4):132:1–132:10, July 2015.

[140] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2021.

[141] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d scene representation and rendering. *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[142] Tony Tung, Shohei Nobuhara, and Takashi Matsuyama. Complete multi-view reconstruction of dynamic scenes from probabilistic fusion of narrow and wide baseline stereo. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1709–1716, 09 2009.

[143] Wim M. van Rees, Etienne Vouga, and L. Mahadevan. Growth patterns for shape-shifting elastic bilayers. *Proceedings of the National Academy of Sciences*, 114(44):11597–11602, 2017.

[144] Roman Vetter, Norbert Stoop, Thomas Jenni, Falk K. Wittel, and Hans J. Herrmann. Subdivision shell elements with anisotropic growth. *International Journal for Numerical Methods in Engineering*, 95(9):791–810, 2013.

[145] Bin Wang, Paul G. Kry, Yuanmin Deng, Uri M. Ascher, Hui Huang, and Baoquan Chen. Neural material: Learning elastic constitutive material and damping models from sparse data. *CoRR*, abs/1808.04931, 2018.

[146] Bin Wang, Longhua Wu, KangKang Yin, Uri M Ascher, Libin Liu, and Hui Huang. Deformation capture and modeling of soft objects. *ACM Transactions on Graphics*, 34(4), 2015.

[147] Huamin Wang, James F. O'Brien, and Ravi Ramamoorthi. Multi-resolution isotropic strain limiting. *ACM Transactions on Graphics*, 29(6):156:1–10, December 2010. Proceedings of ACM SIGGRAPH Asia 2010, Seoul, South Korea.

[148] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *European Conference on Computer Vision (ECCV)*, pages 52–67, 2018.

[149] Stephanie Wang and Albert Chern. Computing minimal surfaces with differential forms. 40(4), 2021.

[150] Wen Wang, Lining Yao, Teng Zhang, Chin-Yi Cheng, Daniel Levine, and Hiroshi Ishii. Transformative appetite: Shape-changing food transforms from 2d to 3d by water interaction through cooking. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 6123–6132, New York, NY, USA, 2017. ACM.

[151] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[152] Max Wardetzky, Miklós Bergou, David Harmon, Denis Zorin, and Eitan Grinspun. Discrete quadratic curvature energies. *Comput. Aided Geom. Des.*, 24(8-9):499–518, November 2007.

[153] Anna Wawrzinek, Klaus Hildebrandt, and Konrad Polthier. Koiter's Thin Shells on Catmull-Clark Limit Surfaces. In Peter Eisert, Joachim Hornegger, and Konrad Polthier, editors, *Vision, Modeling, and Visualization (2011)*. The Eurographics Association, 2011.

[154] Clarisse Weischedel. *A discrete geometric view on shear-deformable shell models*. PhD thesis, Georg-August-Universität Göttingen, 2012.

[155] S. Weiss, R. Maier, D. Cremers, R. Westermann, and N. Thuerey. Correspondence-free material reconstruction using sparse surface constraints. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, June 2020.

[156] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9421–9431, 2021.

[157] Hang Xiao and Xi Chen. Modeling and simulation of curled dry leaves. *Soft Matter*, 7(22):10794, 2011.

[158] Leo Xu, Xiaowei He, Wei Chen, Sheng Li, and Guoping Wang. Reformulating hyperelastic materials with peridynamic modeling. *Computer Graphics Forum*, 37:121–130, 10 2018.

[159] Shan Yang, Junbang Liang, and Ming C. Lin. Learning-based cloth material recovery from video. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4393–4403, 2017.

[160] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5336–5345, 2020.

[161] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021.

[162] Rui Yu, Chris Russell, Neill DF Campbell, and Lourdes Agapito. Direct, dense, and deformable: Template-based non-rigid 3d reconstruction from rgb video. In *Proceedings of the IEEE international conference on computer vision*, pages 918–926, 2015.

[163] Weiran Yuan, Yujun Chen, Andre Gagalowicz, and Kaixin Liu. Application of meshless local petrov-galerkin (mlpg) method in cloth simulation. *CMES - Computer Modeling in Engineering and Sciences*, 35:133–154, 10 2008.

[164] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020.

[165] Li Zhang, Brian Curless, and Steven M. Seitz. Spacetime stereo: shape recovery for dynamic scenes. *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, 2:II–367, 2003.

[166] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[167] Michael Zollhöfer, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon,

Charles Loop, Christian Theobalt, et al. Real-time non-rigid reconstruction using an rgb-d camera. *ACM Transactions on Graphics (ToG)*, 33(4):1–12, 2014.

Index

Vita

Hsiao-yu Chen was born and raised in Taiwan. She received her Bachelor and Master of Science degree in Electrical Engineering from National Chiao Tung University in Taiwan. She later started graduate studies at the Oden Institute for computation engineering and sciences at the University of Texas at Austin. During this period, she did one research internship at Walt Disney Animation Studios and another one at Facebook Reality Labs.

Permanent address: 201 E 24th St, Austin, TX 78712

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.