Diss. ETH No. 20015

Flexible, Unified and Directable Methods for Simulating Deformable Objects

A dissertation submitted to
ETH Zurich

for the Degree of
Doctor of Sciences

presented by
Sebastian Martin
Dipl. Informatik-Ing., ETH Zurich, Switzerland
born 30. October 1980
citizen of Switzerland

accepted on the recommendation of
Prof. Dr. Markus Gross, examiner
Prof. Dr. Mario Botsch, co-examiner
Prof. Dr. Eitan Grinspun, co-examiner

2011

Abstract

Deformable objects are omnipresent in our everyday life. In order to create believable virtual worlds in computer graphics that capture and creatively extend our familiar perception of reality, it is indispensable to reflect physical deformation behavior in a faithful, yet simple and efficient manner. In this thesis we revisit and extend FEM-based simulation techniques in each of its main components in order to achieve more *flexible* numerical handling, to *unify* the specialized geometry-dependent simulation codes, and to provide artists with better *directabilty* of simulation outcomes.

The first part focuses on different *discretization* schemes in order to enable arbitrary polyhedral elements as simulation primitives for FEM, achieve convergent meshless simulations requiring just simple point sets as discretization structures, and enable feature preservation at sub-element scales. The novel methods result in considerably simpler handling of topological changes in the case of cutting or fracturing events, or when refining resolution in adaptive simulations. This discretization flexibility is achieved by employing recent advances in geometric interpolation schemes that enable the definition of suitable simulation subspaces.

In the second part we focus on *specialized simulation* codes for different types of object geometries. Building up on the meshless approach of the first part, we follow the method of resultant-based formulation to its logical extreme and derive a higher-order integration rule, or elaston, measuring stretching, shearing, bending, and twisting along any axis. The theory and accompanying implementation do not distinguish between forms of different dimension (solids, shells, rods), nor between manifold regions and non-manifold junctions. Consequently, a single code accurately models a diverse range of elastoplastic behaviors, including buckling, writhing, cutting and merging.

The third part of the thesis then concentrates on *constitutive relation* — how material responds to deformation — and proposes an example-based approach for simulating complex elastic material behavior. Supplied with a few poses that characterize a given object, the system starts by constructing a space of preferred deformations by means of interpolation. During simulation, this example manifold then acts as an additional elastic attractor that guides the object toward its space of preferred shapes. Added on top of existing solid simulation codes, this example potential

effectively allows us to implement inhomogeneous and anisotropic materials in a direct and intuitive way. Due to its example-based interface, the method promotes an art-directed approach to solid simulation.

Zusammenfassung

Deformierbare Körper sind in unserem täglichen Leben allgegenwärtig. Um in der Computer Grafik glaubhaft virtuelle Welten zu generieren, welche unserer gewohnten Wahrnehmung der Realitt auch gerecht werden, ist es daher unumgänglich, auch ihr physikalisches Verhalten möglichst wahrheitsgetreu, jedoch auch einfach und recheneffizient abzubilden. Diese Doktorarbeit erweitert moderne FEM-basierte Simulationstechniken in ihren drei Hauptfeldern, um die nummerische Handhabung der Diskretisierungsgeometrie *flexibler* zu gestalten, um die verschiedenen Geometrie-abhängige Simulationscodes zu *vereinheitlichen*, und um Benutzer den Ausgang von Simulationen besser *voraussagen* zu lassen.

Im ersten Teil fokussiert sich die Arbeit auf neue Diskretisierungsansätze, um erstens allgemeine polyhedrale Elemente als Simulationsprimitive für FEM verfügbar zu machen, um zweitens konvergente punkt-basierte Simulationen zu ermöglichen, die nur eine einfache Punktmenge als Diskretisierungsstruktur benötigen, und um drittens Erhaltung von feinskaligen Details auf Sub-Element Ebene zu erreichen. Bei Applikationen wie dem Schneiden oder Zerreissen von elastischen Materialien, oder recheneffizienten adaptiven Simulationen, erlauben die neuen Methoden eine wesentlich einfachere Handhabung der Diskretisierungsstruktur. Diese verbesserte *Flexibilität* in der Diskretisierung wird dadurch ermöglicht, dass neue geometrische Interpolationsverfahren verwendet werden, um die nötigen Simulations-Unterräume aufzuspannen.

Im zweiten Teil konzentrieren wir uns auf spezialisierte Simulationsmethoden für die *verschiedenen Klassen von Objektgeometrien*. Auf dem punkt-basierten Ansatz des ersten Teils aufbauend, folgen wir den klassischen reduzierten Formulierungen zu ihrem logischen Schluss and leiten ein Integrationsschema höherer Ordnung her, dem Elaston, welches die Messung von Streckung, Scherung, Biegung und Verdrehung in jede Raumrichtung erlaubt. Die Theorie und entsprechende Implementierung macht keinen Unterschied zwischen Formen unterschiedlicher Dimensionen (Voluminas, Flächen oder Kurven), oder Verbindungen zwischen mannigfaltigen und nicht-mannigfaltigen Regionen. Konsequenterweise kann ein einziger Code ein breites Spektrum von elastoplastischem Verhalten modellieren, wie zum Beispiel dem Knicken, Schneiden oder Verschmelzen.

Der dritte Teil der Arbeit konzentriert sich auf *Materialgesetze*, welche den Zusam-

menhang zwischen Deformation und Materialantwort modelliert, und stellt einen Beispiel-basierten Ansatz zur Modellierung komplexer elastischer Materialverhalten vor. Anhand einer Handvoll Beispielposen, welche charakteristische Objektdeformationen beschreiben, erstellt unser System mittels eines Interpolationsverfahrens zuerst einen Raum präferierter Deformationen. Während der Simulation wirkt dieser Beispielraum als zusätzlicher elastischer Attraktor, welcher das Objekt Richtung Beispielraum lenkt. In Zusammenarbeit mit einem konventionellen Simulationscode erlaubt uns das Beispielpotential, inhomogene und anisotrope Materialien zu modellieren. Die Beispiel-basierte Schnittstelle ermöglicht dem Anwender darum, kontrollierbare Simulationen deformierbarer Körper auf intuitive Weise zu erstellen.

Acknowledgments

My cordial thanks go to my advisor Prof. Markus Gross. He ignited my interest in the fascinating world of computer graphics and his boundless enthusiasm and great talent in opening and investigating new research fields was of great inspiration. His experience, unbroken support and trust during my Ph.D. were invaluable.

I'm also deeply thankful to my close collaborators. It was a big honor and great pleasure to closely collaborate and realize the many fruitful projects with Peter Kaufmann. Also, I was in the lucky position to learn from the best: having Prof. Mario Botsch as supervisor was invaluable, he introduced me into the exciting world of science and his personal support and scientific intuition throughout the years were indispensable. Furthermore, the brilliant mind and creative bursts of Prof. Eitan Grinspun were groundbreaking and opened the way to priceless experiences. Moreover, I also want to thank Dr. Bernhard Thomaszewski for the fruitful collaboration and the many uplifting and inspiring discussions — he was of invaluable help during the last year. I also specially want to thank Mario and Eitan for inviting and having us for inspiring visits in Bielefeld and New York.

Many thanks go also to my additional collaborators Christoph Huber, Jeronimo Bayer and Liana Manukyan, who contributed in various ways to this thesis. It was a pleasure to work with them and I'm thankful for the great commitment and effort they put into the realized projects.

Special thanks go also to Gian-Marco Baschera, Tobias Pfaff, Simon Heinzle, Cengiz Öztireli, Christian Theiler and my former (party-)office mate Bernd Bickel! The many fruitful work and non-work related discussions were indispensable and of great support throughout this time.

Furthermore I also want to thank the current and former members of CGL, DRZ and AGG, which made the time at the lab fun, diverting and a great place to spend the many days and nights.

I'm also deeply thankful to my friends and family, particularly to my parents, for the great support I could experience throughout this years. Last, I'm most thankful to Michela for going through all these sometimes nice, sometimes rough periods which have only been made possible by her enduring love and support.

This work has been made possible by the Swiss National Science Foundation grants 200021-117756 and 200021-130596.

Contents

CHAPTER 1

Introduction

Deformable objects show a wide variety of behaviors: they stretch and compress; twist, curl and knot; rip under large deformations; form complex wrinkles and buckle under pressure; flow viscously and deform plastically into different forms — and we encounter them virtually everywhere. From human tissue, hair and cloth, to paper, wires and plastic bottles in indoor environments; from metal plates, steel cables and flags in urban places up to trunks, branches and leaves in nature — deformable objects are omnipresent. In order to create believable virtual worlds in computer graphics applications that capture and creatively extend our familiar perception of reality, it is indispensable to reflect physical deformation behavior in a faithful, yet simple and efficient manner.

The field of *continuum mechanics* (CM) [Lai et al., 1978] provides the basic physical laws to mathematically describe the complex interactions of deforming geometries and resulting physical forces in a *continuous form*. Complementary, the *Galerkin* approach is nowadays the main numerical principle to formulate their *discrete* counterparts to set up corresponding simulations in a computer. In physically-based animations, the most frequent instance of this principle is the finite element method (FEM) [Bathe, 1995], which divides an object into simplicial *elements* upon which the discrete solution is represented and computed.

While the FEM in its basic form works well for many cases, it does have its limitations. The most important one is that the usually tetrahedral or hexahedral elements must be well-shaped to guarantee stable numerics. However, many graphics applications involve changes in mesh topology: cutting of virtual flesh in surgical simulators, fracturing of highly-stretched materials, or adaptively focusing computational resources onto specific regions all require frequent manipulation of the discretization. Maintaining valid meshes in such applications is a very demanding task [Bargteil et al., 2007; Alliez et al., 2005]. One reason for this difficulty stems from the fact that usual simulation codes only work with simple element shapes and interpolation schemes. As we will show, we do however not need to restrict ourselves to these basic element types. By extending the range of amenable element shapes we can naturally work with meshes gained during the various geometric operations. Further, we will see that it is possible to omit mesh generation completely and still maintain convergence properties, and how to preserve visually important features of deforming materials. Conceptually, these different goals to make conventional FEM more *flexible* are achieved in the same manner by simply exploring suitable choices for the solution subspaces.

Another important aspect of discretization methods is the geometry under consideration: bodies can take forms from long and slender *rods*, to flat and wide *shells*, to thick and bulky *solids*, but also any combination of them. Over the past decades specialized methods have emerged for the efficient and compelling simulation of each of these forms, overcoming the inherent numerical limitations of solid-based FEM approaches that are, in general, not suited for thin geometries. But this specialization has opened a Pandora's box: the interfaces between multiple specialized codes need to be developed, extended and debugged. Furthermore, not only the software interface but also the mathematical model of such interfaces is inherently difficult and unintuitive to develop. It is also not clear how to model the physics of objects that do not neatly fit into one of the categories. Junctions, for example, are outside the scope of most specialized models, and are usually treated as an afterthought. Even more, for shapes that make a smooth transition between one form and another (either along their spatial dimension, or as they evolve temporally), making such binary decisions to categorize them seems conceptually questionable. There is a need for methods that efficiently simulate a spectrum of forms with a unified computational model while still "getting the physics right". We will present a conceptually elegant approach that indeed treats these regimes in a *unified* manner and furthermore only requires a simple point-based setup.

While forward simulation of a material's underlying physical laws allows generating astonishingly faithful results, computer graphics applications often have additional demands. In production pipelines, the course of interacting physical bodies is often imagined beforehand and ideally, a simulation would agree with an artist's devised outcome. However, *art-directing* physical simulations is a demanding task and is still an active research area [McNamara et al., 2004; Wojtan et al., 2006; Barbič and Popović, 2008; Barbič et al., 2009]. Most solution approaches resort to solving complex optimization problems [Witkin and Kass, 1988], involving both space and time, which are inefficient to solve and not amenable for large problem sizes. Investigation into different approaches where these cumbersome formulations can be circumvented is therefore necessary: Ideally, *directing* simulations takes place without referring to complex optimizations and by just performing and controlling single forward simulations. Indeed, we will present an example-based simulation approach that makes use of novel artistic materials that allow to guide forward simulations in an intuitive manner.

1.1 Overview

During the course of this thesis we will address the previously highlighted problem areas and present suitable solution approaches for each of them. A conceptual overview on the performed research activities and resulting publications can be found in Fig. 1.1.

Flexible Galerkin Methods. We will first see how we can improve the basic FEM in a simple manner by introducing properly designed discretization subspaces for the underlying Galerkin principle. By revisiting newly developed interpolation schemes from the area of geometric modeling [Joshi et al., 2007; Adams et al., 2008; Lipman et al., 2008], we will see different possibilities on how to employ these schemes in the discretization.

These spaces will enable us to simplify the meshing task considerably by introducing more general arbitrary polyhedral elements that can flexibly adapt to topological changes of the mesh structure. Further, by omitting the explicit connectivity information of elements completely, we will arrive at a meshless Galerkin method that works with point samples as only discretization structure. Moreover, using the same approach of designing special purpose solution spaces will also allow us to achieve preservation of geometric features at sub-element scales, being useful for embedded high-resolution surfaces.

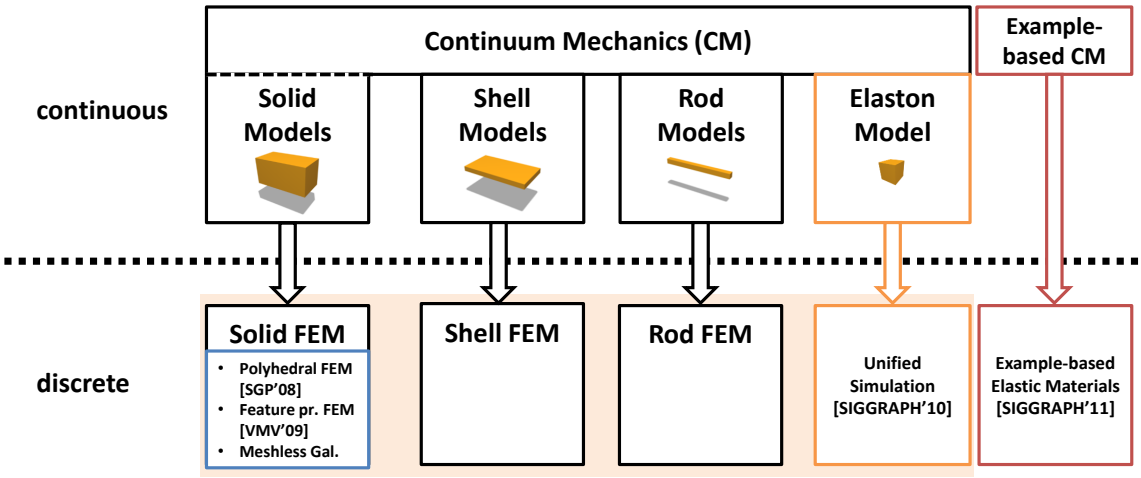


Figure 1.1: *Thesis Overview: In summary, this work focuses on three distinct areas for simulating deformable objects. We first consider the simulation of solids and aim at generalizing the basic FEM to handle more flexible discretization structures by introducing* flexible Galerkin methods *(blue). Then, we will enlarge the scope of possible geometries and present an approach to handle the different geometric forms in a* unified simulation *framework (orange). Lastly, we will consider the problem of enhancing basic forward simulations with an additional control metaphor that allows better* art-directability *of their outcome (red).*

Unified Simulation. We will further see that handling different geometric forms, spanning rods, shells, and solids, with a *unified* solution methodology is possible. By unified, we mean that the code does not distinguish between forms. We will draw motivation from previous unification efforts, but will be set apart by our emphasis on physical correctness, specifically convergence to the continuum model.

The resulting method builds on the presented point-based discretization for solids and demonstrates efficient, accurate simulations which converge to the smooth underlying continuum formulation for any geometric form, ensuring that simulations are consistent under resampling or refinement of the geometry. The gained results show excellent agreement with established benchmarks for rods, shells, and solids, a consequence of the theoretically-grounded development of the method.

But the approach extends beyond the scope of standard benchmarks: without any modification to the implementation, we demonstrate compelling examples on non-manifold geometry (where classical rod or shell models break down) and on hybrid forms that cannot be discretely classified as rods,

shells, or solids (where classical rod or shell models do not apply, and naïve implementations of volumetric elastica suffer from poor numerics).

Art-Directability. We will also present an approach to *art-direct* physically-based animations in a simple, forward simulated way, without requiring formulations of laborious space-time optimizations. Forward simulations usually offer control over material properties as a way of controlling the final deformation. But in creative applications such as computer animation, material properties are just middlemen in a process that really focuses on obtaining some desired deformation. Indeed, we can flip the causality between materials and deformation: when we witness the deformation of an object, we implicitly draw conclusions about its underlying, constitutive material. Therefore, if we can reverse-engineer the material model favoring prescribed artist-specific deformations, this allows directing the outcome of simulations in an intuitive manner.

Inspired by *example-based* graphical methods for texture synthesis [Wei et al., 2009], rigging [Li et al., 2010] and mesh posing [Sumner et al., 2005]), we will present an intuitive and direct method for artistic design and simulation of complex material behavior. Our method accepts a set of poses that provide examples of characteristic *desirable* deformations, created either by hand (digitized from clay sculptures), with a modeling tool, or by taking 3D "snapshots" of previous simulations. With these examples at hand, we provide a novel forcing term for dynamical integration that causes materials to obey the "physical laws" implied by the provided examples.

1.2 Principal Contributions

This thesis makes the following contributions:

- **Arbitrary Polyhedral Elements** We propose harmonic coordinates (HC) [Joshi et al., 2007] as a generalization of classic linear and multilinear shape functions that fit seamlessly into previous FEM codes. They are evoked when complex elements emerge during topological changes and therefore can be applied efficiently and in a convenient manner. Furthermore we also present a simple and efficient approach to approximate the shape functions numerically by means of the method of fundamental solutions (Chapter 4).
- **Meshless Galerkin** We present an extension of the element-free Galerkin (EFG) method [Belytschko et al., 1994] building up on mov-

ing least squares (MLS) basis functions and which we apply to corotated linear elasticity. The resulting framework is lightweight and efficient, using just simple point sampling of the simulation domain (Chapter 4).

- **Feature-Preserving Solid Simulation** We propose a cage-based simulation framework for deformable solids building up on Green Coordinates (GC) [Lipman et al., 2008] as shape functions that preserve small scale features due to their quasi-conformal interpolation property. A suitable linearization of the nonlinear discretization manifold allows to achieve comparable performance as conventional linear discretization approaches (Chapter 4).
- **Unified Simulation** We introduce elastons as a novel point-based model to describe elastic behavior in small volumes of material. Further we present the concept of applying resultant-based models as quadrature schemes for geometries of higher dimensions. When applied to the elaston model, this results in a unified modeling of geometries of arbitrary forms. To complete the meshless simulation framework, we introduce GMLS as a generalization of MLS allowing arbitrarily arranged point sets for solution representation (Chapter 5). Furthermore, two extensions of this basic elasticity simulation framework are introduced. A meshless version of the virtual node algorithm [Molino et al., 2004] for the handling of topological changes is presented which we exemplify for cutting applications. We also introduce an additive plasticity model for the elaston model along with a resampling technique to allow for large deformation.
- **Example-based Simulation** We introduce strain space as a new feature space that allows simple and natural interpolation of different deformations of an object. Given desired key poses, this then allows to define an example manifold of preferred shapes. Further, we present a projection technique allowing associating an arbitrary deformation to a point on the example manifold. This constitutes the foundation for introducing a novel elastic potential, attracting objects toward the example manifold (Chapter 6).

1.3 Thesis outline

The thesis is organized as follows:

- **Chapter 2** discusses related work for the touched fields of deformable

body simulation. Work directly related to more specific concepts will be presented in their respective chapters.

- **Chapter 3** gives a brief introduction to the relevant principles of continuum mechanics and their numerical treatment in order to follow the next three main chapters of the thesis.
- **Chapter 4** focuses on the design of suitable solution subspaces in order to support more flexible discretization structures such as polyhedral elements and individual points, as well as spaces that are able to preserve sub-element geometric features of embedded meshes.
- **Chapter 5** builds up on the previously introduced point-based method and generalizes it in a manner to support arbitrary thin geometries. Further, also handling of topological changes on the example of cutting is discussed as well as plastic deformation and the therein involved resampling required for large deformations.
- **Chapter 6** introduces directable example-based simulations to give artists an intuitive tool to direct the outcome of solid simulations.
- **Chapter 7** finally concludes the thesis by discussing its main contributions and suggesting potential further work.

1.4 Publications

In the context of this thesis, the following peer-reviewed publications have been accepted.

S. MARTIN, B. THOMASZEWSKI, E. GRINSPUN, and M. GROSS. Example-based Elastic Materials. In *Proceedings of ACM SIGGRAPH (Vancouver, Canada, August 7-11, 2011), ACM Transaction on Graphics, vol. 30, no. 4, pp. 72:1-72:8.*

This paper introduces an additional example-based potential on top of classic elastic energies in order to allow for directable and goal-oriented animations.

S. MARTIN, P. KAUFMANN, M. BOTSCH, E. GRINSPUN, and M. GROSS. Unified Simulation of Elastic Rods, Shells, and Solids. In *Proceedings of ACM SIGGRAPH (Los Angeles, USA, July 25-29, 2010), ACM Transaction on Graphics, vol. 29, no. 3, pp. 39:1-39:10.*

This paper introduces elastons, a new quadrature paradigm for simulating rods, shells and solids in a unified manner.

S. MARTIN, C. HUBER, P. KAUFMANN, and M. GROSS. Shape-Preserving Animation of Deformable Objects. In *Proceedings of Vision, Modeling, and Visualization (VMV) (Braunschweig, Germany, November 16-18, 2009).*

This paper proposes an approach for simulating deformable solids on the basis of Green coordinates, enabling shape-preservation of small features by construction.

S. MARTIN,P. KAUFMANN, M. BOTSCH, and M. GROSS. Polyhedral Finite Elements Using Harmonic Basis Functions. In *Proceedings of Eurographics Symposium on Geometry Processing 2008 (Copenhagen, Denmark, July 2-4, 2008), Computer Graphics Forum, (got the Best Student Paper Award).*

This paper introduces harmonic basis functions into the FEM in order allow for arbitrary polyhedral element shapes.

During the course of this thesis, the following peer-reviewed technical papers have been accepted which are not directly related to the presented work.

P. KAUFMANN, S. MARTIN, M. BOTSCH, E. GRINSPUN, and M. GROSS. Enrichment Textures for Detailed Cutting of Shells. In *Proceedings of ACM SIGGRAPH (New Orleans, USA, August 3-7, 2009), ACM Transactions on Graphics, vol. 28, no.3, pp. 50:1-50:10.*

This paper introduces an approach to accurately model material discontinuities at sub-element level inside thin shells.

P. KAUFMANN, S. MARTIN, M. BOTSCH, and M. GROSS. Flexible Simulation of Deformable Models Using Discontinuous Galerkin FEM. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (Dublin, Ireland, July 7-9, 2008).* Extended Version: In *Journal of Graphical Models, 2009.*

This paper applies the discontinuous Galerkin methodology to simplify the handling of complex elements in the FEM and which allows straightforward application of adaptivity and topological changes.

CHAPTER 2

Related Work

Applying physical laws to automatize the process of generating natural animations has a long tradition in computer graphics. Fluid simulations became popular by the introduction of the full Navier-Stokes equations [Foster and Metaxas, 1997; Stam, 1999], while first models for deformable objects have already been proposed a decade before by the pioneering work of Terzopoulos et al. [1987; 1988]. A good overview on these two core areas of physically-based animation can be found in surveys on deformable structures [Gibson and Mirtich, 1997; Nealen et al., 2006], or the textbook by Bridson [2008] on fluids.

We will divide the vast amount of literature on simulation of deformable objects into four areas, on which we will focus throughout this thesis. *Materials* reviews the work on covering and extending the wide spectrum of different visual effects being associated with different material properties. Closely related to the modeling of specific material effects is the choice of subspace or *solution representation* which should be chosen according to the requirements of the specific task. In *thin structures* we will focus on the works focusing on the handling of thin geometric forms such as shells and rods, and on attempts pursuing the unification of such approaches. Often "orthogonal" to these works are *control* approaches which extend the previous discussed methods to additionally support artistic control.

2.1 Materials

Material models describe the connection between geometric deformations and resulting forces — they govern the way in which objects deform and are thus fundamental to every elasticity simulator.

2.1.1 Elastic Models

Nowadays, the most popular way to simulate deformable objects in graphics is to use the elastic theory of continuum mechanics hand in hand with a FEM discretization for its numerical treatment. This combination has already a long tradition in the mechanics community, its fundament having been constituted long before its first application in computer graphics [Hrennikoff, 1941]. The textbooks of Bathe et al. [1995], Bonet and Wood [1997] or Hughes [2000] give good introductions and overviews on the topic. Starting with Terzopoulos et al. [1987], continuum mechanics based methods started their successful run in graphics and became the standard approach for modeling natural effects.

Linear and Corotational Models. Elastic models describe materials that do not deform permanently but recover their rest state after deformation. One such class bases on the *linear* theory of elasticity where both, deformation and material response, is modeled using linear relationships. Although being only valid for small deformations, its *corotational* extension allows overcoming resulting rotational artifacts, enabling efficient runtime performances and making it attractive to graphics applications. While corotation was first presented in the mechanics community in the mid-70s [Veubeke, 1976], its value for graphics applications has only been discovered later by Müller et al. [2002]. Since then, various extensions and improvements have been presented: Müller and Gross [2004] as well as Hauth and Strasser [2004] fix the ghost force problems of the first approach while Mezger et al. [2008] introduce a quadrature-based method to also handle quadratic shape functions. Kaufmann et al. [2008] present an extension for discontinuous Galerkin FEM, while Thomaszewski et al. [2006] show its application to thin shell models.

In Chapter 4 we present a further extension of the corotational idea, for its application to meshless simulations of solids. Only recently, the inherent stability problem for large deformations of these methods have been addressed in a couple of works: Georgii and Westermann [2008] improve rotation extraction, while Chao et al. [2010] and McAdams et al. [2011] basically formulate a

corotational energy from beginning and performing the proper derivations to attain correct forces and Jacobians.

Nonlinear Models. While the linear theory already allows generating plausible material behaviors for moderate deformations, the resulting animations do often lack realism for more extreme deformations due to the linear relationship in the material force response and deformation measures. *Nonlinear* models are still active research items in mechanical engineering and material sciences [Rubin and Bodner, 2002; Mazza et al., 2005] which in general allow for much more precise description of material behaviors. Already Terzopoulos et al. [1987] used nonlinear deformation measures in conjunction with simple material models to set up the driving potential energies. Subsequent work has then focused on different aspects of nonlinear material: Picinbono et al. [2000] and Irving et al. [2007] set focus on accurately modeling volume-preserving materials, Nesme et al. [2006] on handling inhomogeneities, while again Irving et al. [2004; 2006] also concentrate on fixing element inversion problems as well as modeling anisotropic and plastic deformations for the commonly used Saint Venant-Kirchhoff and neo-Hookean material models.

Though accurate, these conventional material models offer only limited and unwieldy control which is often in opposition to the creative thinking of animators. Bickel et al. [2009] describe an interesting alternative for learning material properties directly from experiments. Only recently, Wang et al. [2011] also present a data-driven material model that captures and learns the non-linear anisotropic elastic behavior of cloth.

Most *artistic materials* do not have a real-world counterpart that could be subject to material measurements. Nevertheless, deducing a material description from given target deformations directly is a powerful concept. We follow this idea in Chapter 6 (and [Martin et al., 2011]) by employing examples of desired deformations to directly construct an elastic potential. Our method can be interpreted as a means of describing strongly anisotropic, heterogeneous and nonlinear materials. But while it is easy to design a set of example poses, defining a corresponding material law is a formidable task.

Coarse Models. Independently of the actual complexity of the material models, recent graphics research also focuses on generating realistic *coarse* models that plausibly simulate complex objects on coarse meshes with relatively few degrees of freedom (DOFs). The homogenization approaches of Kharevych et al. [2009] deduces coarse material properties from given

high-resolution models, while Nesme et al. [2009] and Faure et al. [2011] adapt the basis functions to better model the underlying inhomogeneous material at sub-element scales.

To enhance the simulation results *visually*, the idea of embedding of high-resolution surface geometry into coarse element meshes is advanced continuously: Capell et al. [2002] applied the technique in a multiresolution framework, while Teschner et al. [2004] and Müller et al. [2004b] used embedding in constraint-based and corotational simulation frameworks, respectively. The works of Wojtan et al. [2008; 2009; 2010] then extended the idea to also support dynamically changing embedded geometries that deform, split and merge.

Our feature-preservation approach in Chapter 4 (or [Martin et al., 2009]) and example-based simulation approach in Chapter 6 (or [Martin et al., 2011]) both follow and extend this idea. The former introduces an approach to preserve sub-element geometric features of the embedding by employing Green coordinates [Lipman et al., 2008], while the latter presents a method to actually invert the dependence between simulation and visualization mesh. While physics usually drives the simulation mesh's deformation which in turn deforms the visualization mesh, our approach however flips this relationship and give means to intuitively introduce and enforce meaningful deformation of the *embedded* surface.

2.1.2 Inelastic Deformations and Topological Changes

However, real world materials do not only deform elastically but show complex inelastic deformations and effects such as the plastic deformations of compressing cans, the viscous flow of honey, the brittle fracture of shattering glass or enforced discontinuities when cutting paper. In their seminal papers, Terzopoulos and colleagues [1988; 1989] already pursuit these effects which thereon have steadily been improved.

Plastic and Viscous Flow. Desbrun and colleagues [1996] followed particle-based approaches to achieve first elastic and plastic material effects. A variety of different material properties from fluid to viscous solids have been modeled by varying the viscosity in a Eulerian Navier-Stokes solver [Carlson et al., 2002]. Goktekin et al. [2004] then improved on this by introducing additional elastic terms, allowing for visco-elastic material behaviors — an idea applied to particle-based method by Clavet et al. [2005], and extended to more general solid-fluid interactions [Müller et al., 2004a;

Keiser et al., 2005; Solenthaler et al., 2007]. Losasso et al. [2006] further extend this approach to support an arbitrary number of interacting viscous materials. Alternatively, Müller et al. [2004] followed a FEM-based approach for plastic deformations where they employ O'Brien's [2002] additive plasticity model within their corotational framework. Making use of nonlinear FEM, Irving et al. [2007] proposes a multiplicative plasticity model that stably handle large deformations, and Bargteil et al. [2007] support extreme viscous and plastic deformations by proposing an approach constantly remeshing the rest state. This ideas were continued by Wojtan et al. [2008; 2009] who simplified the remeshing by using embedded surfaces in regular grids and introduced flexible approaches to maintain the fine embedded geometry. An Eulerian solid simulation framework has been presented recently by Levin et al. [2011] to support simpler collision handling for complex scenarios with large viscous flows and many colliding objects. Also recently, Gerszewski et al. [2009] presented a purely point-based method for large viscous and plastic flows. In Chapter 5 and [Martin et al., 2010], we also present a meshless approach for animating plastic flow in our elaston-based framework. It builds up on the additive plasticity model of O'Brien et al. [2002] and a variant of the resampling strategies presented by Bargteil et al. [2007] and Wojtan and Turk [2008].

Fracturing and Cutting. Not surprisingly, the first fracture models have already been introduced by Terzopoulos [1988] which have then been extended to cloth [Norton et al., 1991], and rigid body [Müller et al., 2001; Bao et al., 2007] simulations. Fracturing of brittle and ductile materials has been presented in [O'Brien and Hodgins, 1999; O'Brien et al., 2002], while Smith et al. [2001] propose an approach to actually control fracture patterns. A main challenge in fracture and cutting applications lies in the management of the changing discretization structures induced by the evolving discontinuities.

For FEM-based frameworks, this can be accomplished most simply by just allowing discontinuities to coincide with element faces [Müller and Gross, 2004]. However, this can be restrictive such that the element decomposition approaches have been envisaged in various works of Bielser et al. [1999; 2000; 2003] and Steinemann et al. [2006a] to handle cuts more accurately and flexibly. Alternatively, another class of approaches tackled the problems induced by topologically changing meshes using continuous remeshing [O'Brien and Hodgins, 1999; O'Brien et al., 2002; Steinemann et al., 2006b]. In contrast, the virtual node algorithm of Molino et al. [2004] and its generalizations by Sifakis et al. [2007a; 2007b] dupli-

cate elements instead of splitting them, and embeds the surface parts into those copies. This is similar in mind to the XFEM method [Daux et al., 2000; Areias and Belytschko, 2005] which has recently found its way to graphics for cutting solids [Jeřábková and Kuhlen, 2009] and detailed cutting and fracturing of shells using enrichment textures [Kaufmann et al., 2009b].

Alternatively, point-based approaches do not have to maintain a consistent simulation mesh [Müller et al., 2004a], but on the other hand have to update special shape functions [Pauly et al., 2005] or distance graphs [Steinemann et al., 2006b]. In contrast, Wicke et al. [2007] and Kaufmann et al. [2008] avoid costly remeshing by supporting general convex polyhedra in FEM simulations. Our mesh-based approach presented in Chapter 4 (and [Martin et al., 2008]) generalizes possible element shapes to general polyhedra while still being compatible with the classic element types. The meshless method discussed in Chapter 5 (and [Martin et al., 2010]) presents a generalization of the virtual node algorithm to meshless discretizations, being similar in mind to Rabczuk and Belytschko's [2004] work on cracking particles.

2.2 Solution Representation

The term discretization describes the procedure of transforming a set of differential equations into a corresponding discrete problem achieving an approximate solution. A central role in such discretizations plays the *solution representation*, i.e., the function subspace from which the approximated solution can be chosen. We divide such approaches into three areas: *mesh-based methods* require meshes to model explicit connectivity information between DOFs while *meshless methods* solely rely on points and generate required neighborhood information "on-the-fly". In contrast, *global bases* do not use local basis functions to span the solution space but rather work with globally supported functions as bases, resulting in fully coupled systems.

2.2.1 Mesh-based Methods

FEM. In computer graphics, the FEM is the most popular approach for representing and computing discrete solutions nowadays [Hughes, 2000], although other mesh-based alternatives such as boundary element methods (BEM) [James and Pai, 1999] or finite difference methods (FDM) [Terzopoulos et al., 1987] are also used at times. Only recently, also Eulerian FDM are explored [Levin et al., 2011] to simplify complex contact scenarios.

In graphics, FEMs are implemented almost exclusively using tetrahedral (e.g. by O'Brien and Hodgins [1999]) or hexahedral meshes (e.g. by Müller et al. [2004b] or James et al.[2004]). These discretizations allow for simple and efficient implementations of the FEM, but in turn require rather complex mesh restructuring in case of topological changes, for instance due to fracture and cutting discussed in the last section, or mesh refinement for adaptive simulations.

Adaptivity. The need for accurately simulating highly detailed models has led to the development of adaptive and hierarchical simulation techniques [Debunne et al., 2001; Wu et al., 2001; Grinspun et al., 2002; Capell et al., 2002; Otaduy et al., 2007], where the solution space is locally refined in order to allow for improved accuracy and visual fidelity. Hanging nodes or T-junctions occurring in adaptively refined simulation meshes often pose problems and hence have to be either avoided or treated separately. For instance by refining basis functions instead of elements [Grinspun et al., 2002; Capell et al., 2002], or by constraining hanging nodes to edge midpoints [Sifakis et al., 2007b].

In Chapter 4 (and [Martin et al., 2008]), we propose to handle the hanging nodes due to adaptive refinement within a single, consistent simulation framework based on arbitrary polyhedral elements. The key to such a generalization is to find basis functions, and hence a suitable solution space, that allows for a larger class of element shapes being generated on refinement while still fulfilling all necessary requirements for convergent FEM schemes [Hughes, 2000].

Geometric Modeling and Coordinates. Geometric modeling is another widely studied topic in computer graphics and offers a rich repertoire of techniques for deforming meshes [Sheffer and Kraevoy, 2004; Sorkine et al., 2004; Lipman et al., 2005; Botsch et al., 2006]. These methods are often related to physically-based simulation techniques in that they seek for minimizers of *geometric* energies — pretty much in the same way as simulations seek the minimum of *physcial* potential energies for (quasi-)static problems. Physically-based techniques do just make particular choices in how to penalize certain geometric measures. We refer to the surveys of Milliron et al. [2002] and Botsch and Sorkine [2008] for surveys on the different classes of methods. Moreover, also the actual representation of deformation shares many similarities with physically-based techniques: Many solution representations build on suitable finite function subspaces or manifolds featuring specific

geometric properties, and next to their application in modeling, they also become relevant for us in the design of suitable solution subspaces for physical applications in Chapter 4.

Classic barycentric coordinates are defined on simplicial domains like triangles (2D) and tetrahedra (3D), have been known for centuries and are usually used to interpolate nodal values inside elements in the FEM. Different attempts have been made to generalize them to convex domains [Wachspress, 1975; Meyer et al., 2002; Ju et al., 2005a]. Floater [2003] introduced mean value coordinates (MVC) which have later been generalized to 3D and been applied to surface deformation [Floater et al., 2005; Ju et al., 2005b]. Joshi et al. [2007] introduced a different kind of cage-based coordinates called harmonic coordinates, which are defined on arbitrary non-convex polyhedral cages. While all of these coordinates have the property of being affine-invariant, they are not able to allow for feature preservation of the enclosed geometry. Lipman et al. [2008] introduced a new set of coordinates called Green coordinates which, being associated to both vertices and face normals, are quasi-conformal and have exactly this property. Only recently, Weber et al. [2009] generalized Green coordinates to complex barycentric coordinates for the 2D case, and Ben-Chen et al. [2009] extended them to achieve better control over deformation and conformality.

Next to the applications in modeling, different attempts have been made in graphics to take advantage of their inherent geometric properties in physical simulation problems. Wicke et al. [Wicke et al., 2007] defined basis functions by Mean Value Coordinates for FEM simulations, allowing discretizations with more general convex elements. In Chapter 4 and [Martin et al., 2008], we present generalized polyhedral elements by using harmonic coordinates to construct the solution space, allowing us to circumvent bothersome remeshing operations. Moreover, in the same chapter (or [Martin et al., 2009]), we will also see how to apply Green Coordinates [Lipman et al., 2008] to achieve feature preserving simulations of deformable solids.

2.2.2 Meshless Methods

In contrast to mesh-based approaches, meshless methods do not require explicit connectivity information (we refer to textbook of Gross and Pfister [2007] for a general overview over meshless methods in graphics). They are particularly suited for applications where frequent reorganization of the discretization structure is required (such as large deformations, plastic flow, discontinuities due to fracture or cutting).

Collocation methods basing on smoothed-particle hydrodynamics (SPH) interpolation [Gingold and Monaghan, 1977] allow for flexible simulation of a large range of materials [Desbrun and paule Gascuel, 1996; Clavet et al., 2005; Solenthaler et al., 2007], but also the more general moving least squares (MLS) interpolation scheme [Fries and Matthies, 2004] has been applied successfully to solid simulation by Müller et al. [2004a]. Their basic approach has then been extended to support fracturing [Pauly et al., 2005] by material distance-based shape functions; to solid-fluid coupling [Keiser et al., 2005]; to viscoelastic flow [Gerszewski et al., 2009], or to cutting [Steinemann et al., 2006b] who make use of distance graphs.

In contrast to the element-free Galerkin (EFG) method [Belytschko et al., 1994], all these methods represent collocation approaches of the underlying equations, leading to under-integration and inexact solutions. In Chapter 4, we therefore present a corotational variant of the EFG for solid geometries that allows lightweight Galerkin discretizations with scalable accuracy. In Chapter 5 (or [Martin et al., 2010]) we then present an extension to this basic method which builds up on generalized MLS (GMLS) [Fries and Matthies, 2004] for representing the solution, enabling the unified simulation of bodies of any geometric form.

If accuracy is not the focus, there exist also geometry-driven approaches based shape matching [Müller et al., 2005; Rivers and James, 2007; Steinemann et al., 2008]) that feature superior stability and performance at the cost of physical correctness. Very recent works on coarse meshless discretization [Gilles et al., 2011; Faure et al., 2011; Müller and Chentanez, 2011] carry on the idea of using derivative information in the DOF [Martin et al., 2010] to allow for unrestricted arbitrary point samplings.

2.2.3 Global Bases

A third class of solution spaces can be characterized by their use of *global bases* for representing solutions. James et al. use globally supported Green's functions in an boundary element method (BEM) only requiring a boundary mesh for representing the solution [1999]. But most prominently under these approaches are *model reduction* techniques that only provide DOFs for an object's important deformation modes. Extracting a small representing set from a full deformation basis leads to high simulation performances while keeping high fidelity for characteristic deformations [James and Fatahalian, 2003; Choi and Ko, 2005; Treuille et al., 2006]. Barbič et al. [2005] extend this idea to nonlinear Saint-Venant Kirchhoff models, while An et al. [2008] increase performance further by optimizing the numerical integration scheme. Kim and

James [2009] use full and reduced simulation steps in an interleaved manner to also increase performance. In the context of simulation control, Barbič et al. [2009] also use reduced spaces to significantly improve computation times for the involved spacetime optimizations. Only recently, they extended the original model reduction technique to divide objects into individual parts that are each treated independently [Barbič and Zhao, 2011], similar in mind to Wicke et al.'s [2009] work on modular bases for fluids. Our approach in Chapter 6 is related to these techniques in that we also build a reduced space, the *example manifold*, being a subspace of preferred deformations. However, our simulation framework still uses the full number of DOFs and uses the subspace just to guide the simulation.

2.3 Thin Structures

Terzopoulos et al. [1987] argue that differential geometry provides a natural language for describing the physics of curves, surfaces, and volumetric bodies, naturally establishing a trichotomy of geometric forms whose strains are given by torsion, curvature, and the metric tensor. The exposition mirrors the mechanics literature, where the three forms are each analyzed individually [Malvern, 1969] and *specialized approaches* have been developed for each of them. However, depending on the geometric structure under consideration, certain applications also require the joint interplay of these methods in single environments, leading to the demand for *unification*.

2.3.1 Specialized Approaches

Shells. Reduced models that only use the (mid-)surface to describe thin shell mechanics have a long history in mechanical engineering [Naghdi, 1972] and also drew great attention in graphics for modeling the many thin-walled structure we encounter in our environments, such as cloth or paper. Early techniques rely on mass-spring networks [Provot, 1995; Baraff and Witkin, 1998; Bridson et al., 2002; Choi and Ko, 2002] to approximate the internal membrane mechanics. In parallel, more accurate specialized FEM techniques have been developed [Cirak et al., 2000; Thomaszewski et al., 2006; English and Bridson, 2008; Volino et al., 2009; Thomaszewski et al., 2009], allowing to simulate larger ranges of materials by supporting constitutive material models and which are convergent due to their theoretical basis.

Other research focuses on formulating simple discretization schemes: making use of discrete differential geometry [Desbrun et al., 2008], discrete models

of shells have been developed [Grinspun et al., 2003] which have been extended to support inextensibility [Goldenthal et al., 2007] or to improve its performance [Bergou et al., 2006; Garg et al., 2007; Wardetzky et al., 2007] by efficient bending energy computations.

Also meshless discretization methods have been presented that are working on loose fiber networks [Wicke et al., 2005] or global parameterizations [Guo et al., 2006]. In Chapter 3 we will present a "volumetric" model for thick shell mechanics that also captures normal shear. This approach will then be generalized in Chapter 5 to simulate arbitrary geometries in a unified manner.

Rods. If an object has vanishing extent in two dimensions, such as hair or cables, specialized models usually model the object with simple centerline curves [Hadap et al., 2007]. Basing on Cosserat theories [Rubin, 1985], Pai [2002] presents an elaborate framework for simulating various types of strands. Spillmann et al. [2007] present a method basing on the Kirchhoff rod theory [Langer and Singer, 1996] where they use a quaternion representation for twist, while Bergou et al. [2008] present a different approach basing on notions of discrete differential geometry. They later further improve their original approach to support viscosity and introduce the concept of time-parallel transport to increase performance [Bergou et al., 2010]. Focusing on hair simulation, Bertails et al. [2006] introduce a super-helix model for curled hair, while McAdams et al. [2009] and Sueda et al. [2011] present large scale simulations using a hybrid collisions and reduced dynamics, respectively. Recently, rod models have also been used to create realistic simulation of knitted cloth modeled at the yarn level [Kaldor et al., 2008; 2010]. Our volumetric rod model presented in Chapter 3 also makes use reduced centerline representation, but models thick rods, which can then be unified in Chapter 5.

2.3.2 Unification

Different thin geometric forms appear almost instantaneously when considering complex sceneries for simulation and therefore, also the requirements for flexible unified methods emerge. On one hand, challenges arising from specialization spur researchers to explore techniques that *tie* together existing models, and on the other hand to investigate *unified* approaches.

Tying Models. Finite element (FE) packages such as ABAQUS and SOFEA encapsulate a plethora of diverse elements and techniques, often using dynamic method dispatch to provide a unified application interface to distinct underlying codes [Krysl, 2005]; additional specialized code is required to capture the physics of interactions between differing models. Sifakis et al. [2007b] formulate an "all-purpose glue" to pass forces and constraints between various physical models using *soft* and *hard bindings*; this method models the physics of interactions between two black-box codebases and also easily extends existing codes to handle non-manifold geometry, however it can require the computation of a non-physical mass associated to the binding particles. Similar in mind is the recent work of Twigg et al. [2010], presenting the Procrustes transform as mean to stably define point-wise coupling. Glue techniques are particularly attractive when there is a need to quickly combine a number of distinct existing codebases.

Unifying Elastic Models. However, a gluing strategy does not lighten the burden of maintaining multiple codes (rather it introduces additional code). To keep code manageable and extendable, various researchers advocate sacrificing some benefits of specialization for the simplicity and ultimately scalability of a unified treatment of elastica. Many works in graphics use networks of point masses because they can be connected by any combination of stretching, bending [Baraff and Witkin, 1998; Bridson et al., 2003], and altitude springs [Selle et al., 2008]; constraints can replace stiff springs for more stable integration and can also allow for unilateral action [Provot, 1995; Müller et al., 2007]. Stam's *Nucleus* [2009] efficiently enforces competing constraints on arbitrary simplicial complexes capturing a diverse range of materials. Our approach presented in Chapter 5 and [Martin et al., 2010] is inspired by the goals of generality and simplicity, and departs by additionally asking for a convergent approximation of a continuum formulation. Similar in mind, Cosserat points [Rubin, 1985] model a small elastic volume equipped with its own mass, DOFs, and elastic energy. Cosserat points must be glued together explicitly by kinematic constraints; since our elastons are quadrature points embedded in a separately-defined deformation field, no formulation of glue is necessary.

Some works attempt to approach a continuum result by careful choice of masses and spring coefficients [Etzmuss et al., 2003; Zerbato et al., 2007] using prestressed configurations [Wang and Devarajan, 2005; Lloyd et al., 2007] or biquadratic springs [Delingette, 2008], or by factoring and approximating FEs [Kikuuwe et al., 2009], with applications to cloth simulation [Volino et al., 2009]. However, researchers agree with Van Gelder's claim [1998] that

it is impossible to expect mass-spring networks to converge to continuum models for the general setting of arbitrary material parameters and network topology.

2.4 Control

Directing simulations is of paramount importance for setting bounds to otherwise uncontrolled physics in practical scenarios. If simulation is left aside, the general field of *shape interpolation* plays a central role in animation from which we draw inspiration for our control approach.

Directing Simulations. Many methods have been proposed for this purpose: explicit control forces enforce keyframes in a direct manner [Thürey et al., 2006], while space-time constraints minimize the induced forces by optimizing over time [Witkin and Kass, 1988]. The original approach has been extended to fluid control by Treuille et al. [2003] but still suffers from poor performance due large-scale optimizations. McNamara et al. [2004] and Wojtan et al. [2006] both apply the adjoint method to improve speed, while Barbič et al. [2009] perform the space-time optimization in a reduced space. A different control paradigm is followed by tracking approaches [Kondo et al., 2005; Bergou et al., 2007; Barbič and Popović, 2008] where the simulation is constrained in order to follow some predefined trajectories while still being able to provide high-frequent physical features. Yet another methodology is pursued by methods for editing existing animations [Popović et al., 2000; Kircher and Garland, 2006] or the sampling of probable animations [Twigg and James, 2007] from which suitable ones are selected. Our approach presented in Chapter 6 (or [Martin et al., 2011]) resembles existing approaches in that it also induces additional forces into the simulation. A striking difference is, however, that these forces derive from a *conservative potential* as opposed to the *non-conservative control forces* of existing methods. As contrast to methods based on trajectory control, our approach does not require (or imply) a fixed plot of keyframes — it rather promotes a style of context-sensitive deformation control in which objects are guided towards preferred shapes but are otherwise unrestricted in their motion.

Shape Interpolation. Interpolating between poses is a general problem in character animation [Lasseter, 1987]. The particular case of interpolating between surface geometry without skinning, i.e., referring to higher-order information such as a skeleton, has also attracted the focus of research [Alexa

et al., 2000; Kilian et al., 2007; Winkler et al., 2010; Chao et al., 2010]. Similarly, also the transfer of deformations between two different meshes in a semantically meaningful way is of practical interest [Sumner and Popović, 2004; Baran et al., 2009].

Central to all of these types of applications is the construction of an adequate space for representing shapes, where the actual interpolation operation is defined. Most similar to our control approach is the work of Sumner et al. [2005], whose *MeshIK* method combines shape interpolation and editing into an intuitive modeling paradigm. While we draw valuable inspiration from MeshIK, our method differs in two important aspects. First, whereas Sumner et al. interpolate between factored deformation gradients of triangles, we employ a nonlinear strain measure evaluated on tetrahedra. This is closer to the approach by Winkler et al. [2010], who interpolate between dihedral angles and edge lengths, corresponding to discrete strain measures on triangular surfaces [Grinspun et al., 2003]. Second and more importantly, whereas MeshIK is restricted to static geometric modeling, our approach is the first to leverage example-based methods for *dynamic* physical simulation to achieve example-based simulation control.

CHAPTER 3

Prerequisites

This chapter discusses elements from continuum mechanics of elasticity and its numerical treatment which will serve as a theoretical basis for the later chapters. We divide the vast topic into three parts: We first discuss the continuous linear and nonlinear theory (Section 3.1), present some parts of the specialized theories for shells and rods (Section 3.2), and then discuss the Galerkin method for discretizing the continuous forms to arrive at the actual simulation frameworks (Section 3.3).

3.1 Continuum Mechanics of Elastic Objects

Depending on the actual type of material and the scenarios in which these materials are used, different continuum mechanical models are best suited to describe their behavior. We will briefly introduce two commonly used theories for describing the behavior of elastically deformable *solids*, i.e., objects that deform completely elastic and whose geometric extends are sufficiently large to be well-captured by these theories. We will first introduce linear elasticity and the popular corotational extension, and then describe the more general nonlinear theory. For a more detailed treatment of the topic we refer to the textbooks [Chung, 1996; Hughes, 2000] and the excellent survey of Nealen et al. [2006].

3.1.1 Linear Elasticity

The theory of linear elasticity is commonly used in computer graphics due to its relatively simple formulation and resulting efficient simulations. We will now review the main components of the classical linear theory of solid elastica and briefly discuss the commonly known corotational extension [Müller et al., 2002; Hauth and Strasser, 2004] used to fix rotation artifacts.

We can divide the theory of elasticity into three essential parts which we will study here in the light of the linear theory: The first considers *geometry*, the formal study of deformation a body can undergo, without describing the cause. The second block considers the effect of internal and external *forces* acting on a body and how they affect an object's *equilibrium* or *dynamics*. Last, these two parts are related to each other by a *constitutive relation*, describing how deforming material's geometry reacts to present internal forces.

Notation. Let us use the following notation: The comma denotes partial differentiation, e.g., $\mathbf{x}_{,i} \equiv \partial \mathbf{x}/\partial \theta_i$, $\mathbf{u}_{,ik} \equiv \frac{\partial^2 \mathbf{u}}{\partial \theta_i \partial \theta_k}$, while the dot ($\cdot$), cross ($\times$), and colon (:) denote the vector dot product, vector cross product, and tensor double contraction, respectively. If required, we will also make use of index notation and Einstein's summing convention (we refer to [Bonet and Wood, 1997] for a good introduction), but will try to stick to standard vector and matrix notation where possible.

Geometry

Classically, deformation can either be studied from the *Lagrangian* point of view, i.e., by describing deformation of single material particles, or from the *Eulerian* point of view, i.e., by describing deformation occurring in a small spatial region. While both approaches are valid and have their advantages, we will restrict ourselves to a Lagrangian description in the following.

Consider a material whose undeformed positions $\bar{\mathbf{x}}(\boldsymbol{\theta})$ are parameterized by curvilinear coordinates $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)^T$ over the material domain Ω. When the material undergoes a deformation, an undeformed material point $\bar{\mathbf{x}}(\boldsymbol{\theta})$ is displaced to the new position

$$\mathbf{x}(\boldsymbol{\theta}) \;=\; \bar{\mathbf{x}}(\boldsymbol{\theta}) + \mathbf{u}(\boldsymbol{\theta})\,. \tag{3.1}$$

Since the deformation function $\mathbf{u}(\boldsymbol{\theta})$ gives us a complete description of the deformed state of an object, it can be used to derive other deformation measures upon which constitutive relations will be formulated. Note that we could

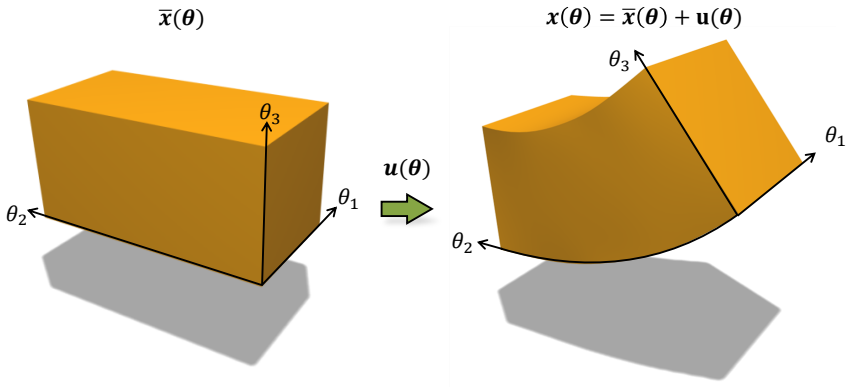


Figure 3.1: *A undeformed solid block described by $\bar{\mathbf{x}}(\boldsymbol{\theta})$ which is mapped to its deformed configuration by the displacement field $\mathbf{u}(\bar{\mathbf{x}})$.*

also directly describe the undeformed geometry in a Cartesian coordinate system without giving a parameterization of the two domains, as it is done commonly (see e.g. [Müller et al., 2002]). However, since we will later deduce simplified models on top of the solid formulation, it is more convenient to introduce the theory with more general curvilinear coordinates.

Cauchy Strain. The amount of stretch and shear that a material undergoes during a deformation are important measures which can be related to resulting physical forces. As we will see later, these two measures can be captured by considering how dot products between infinitesimal direction vectors change during a deformation. By assuming only small displacements, the linear 3×3 *Cauchy strain* $\boldsymbol{\epsilon}$ can be formulated

$$\boldsymbol{\epsilon}_{ij} = \frac{1}{2}\left(\mathbf{u}_{,i} \cdot \bar{\mathbf{x}}_{,j} + \bar{\mathbf{x}}_{,i} \cdot \mathbf{u}_{,j}\right), \tag{3.2}$$

where $\bar{\mathbf{x}}_{,i}$ and $\mathbf{x}_{,i}$ are the local frames of the undeformed and deformed configuration, respectively. The main diagonal of the tensor measures the amount of stretch in the three spatial directions, while the off-diagonal values measure the amount of shear in the according planes. The Cauchy strain is the linearization of the more general nonlinear Green strain that we will encounter in Section 3.1.2.

Forces, Equilibrium and Dynamics

As next, we will give a formal description on how forces act on materials and how we can state corresponding conditions for equilibrium or the equation of motions for the dynamics of the system.

Cauchy Stress. Consider a small neighborhood of particles inside a elastic body. By deforming the object, their relative positions change which results in restoring forces induced by the elastic material. By introducing a virtual cut plane with normal $\mathbf{n}$ through this neighborhood, we get the restoring forces $\mathbf{f_n}$ acting across the plane area.Since this plane can have an arbitrary orientation the force distribution at a point can compactly be described by the product of the 3×3 *Cauchy stress* tensor σ with the plane normal, giving

$$\sigma \cdot \mathbf{n} = \mathbf{f_n}\,. \tag{3.3}$$

Similar to the Cauchy strain, the main diagonal holds the normal stress while the off-diagonal holds the shear stress components.

Forces. The total force acting on a given point can then be computed by summing up all traction forces on the side of the corresponding infinitesimal cube. Using the divergence operator, this leads to the simple expression

$$\nabla \cdot \sigma \; + \mathbf{f} = 0,$$

for the total force.

Energy. The internal forces acting in an elastic body are *conservative*, i.e., the work performed by such forces is independent of the actual (deformation) path taken. Such forces can be related to an underlying scalar energy potential $W_{int}(\mathbf{u})$, characterizing the amount of work required to achieve to a given deformation $\mathbf{u}$. This energy functional is formed by taking the tensor products of strain ("length") and stress ("force") integrated over the material domain, i.e., by

$$W_{int}(\mathbf{u}) = \int_{\Omega} \epsilon(\mathbf{u}) : \sigma(\mathbf{u})\, d\Omega. \tag{3.4}$$

Then, the conservative forces are simply deduced by taking the variational derivative [Gelfand and Fomin, 2000] of the elastic potential:

$$\mathbf{f}_{int} = -\frac{\partial W_{int}(\mathbf{u})}{\partial \mathbf{u}}\,. \tag{3.5}$$

Knowing the potential of an elastic body therefore gives us a handy way to deduce the necessary information required for actual simulations.

Equilibrium. In a *static equilibrium*, all internal and external forces acting on a object (in its interior or on its boundary) need to cancel each other out.

Denoting external forces by $\mathbf{f}_{ext}$, we can formulate the governing partial differential equation (PDE) as

$$\nabla \cdot \sigma = \mathbf{f}_{ext}. \tag{3.6}$$

Making use of the elastic potential gives then also the alternative formulation of the static equilibrium equation

$$-\frac{\partial W_{int}(\mathbf{u})}{\partial \mathbf{u}} = \mathbf{f}_{ext}. \tag{3.7}$$

Later, we will mostly rely on this second form since it is better suited for setting up the corresponding discrete problems.

Equations of Motion. If an object is not in static equilibrium, the difference between internal and external forces results in net forces which lead to acceleration of the material according to Newton's second law. The dynamic behavior is described by the governing equation of motion

$$\rho\, \ddot{\mathbf{u}} + \mathbf{f}_d(\dot{\mathbf{u}}) + \nabla \cdot \sigma \;=\; \mathbf{f}_{ext}\,,$$

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

Again, making use of the elastic energy potential gives the alternative formulation

$$\rho\, \ddot{\mathbf{u}} + \mathbf{f}_d(\dot{\mathbf{u}}) - \frac{\partial W_{int}(\mathbf{u})}{\partial \mathbf{u}} \;=\; \mathbf{f}_{ext}\,, \tag{3.8}$$

that will show more convenient for deriving the discrete problems.

Constitutive Relation

The simplest constitutive relation, relating the geometric deformation measure (3.2) with resulting internal stresses (3.3), is given by a Hookean material that yields a simple linear relationship

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

This material, given formally as a 4-tensor $\mathbf{C}$, is characterized by only two parameters. The Young modulus E describes the material's stiffness, while the Poisson's ratio ν defines how much (linearized) change in volume is penalized during deformation of the material.

Strong Form, Weak Form and Energy Formulation

Strong Form. The given derivation results in a pointwise description of the relationship between material deformation, internal and external forces. The relations seen in Eq. (3.6) and Eq. (3.8) are usually called the *strong form* of the respective problems and are formulated as PDEs. However, the problem can also be stated in different forms by taking variational viewpoint on the problem. The concepts of Lagrangian mechanics [Landau and Lifshitz, 2003] are of great importance for the Galerkin discretization that we will later perform to transform the continuous problem into a discrete one and whose different applications will be discussed throughout this thesis.

Energy Formulation. From the point of view of variational calculus [Gelfand and Fomin, 2000], Equation (3.6) is nothing else than the Euler-Lagrange equation of a corresponding energy functional, i.e., it describes the necessary condition for being at the minimum of the potential energy. In the framework of linear elasticity we discuss here, this energy functional is simply

$$W_{tot}(\mathbf{u}) = \frac{1}{2}\int_{\Omega} (\boldsymbol{\epsilon}(\mathbf{u}) : \boldsymbol{\sigma}(\mathbf{u}) - \mathbf{f}_{ext} \cdot \mathbf{u})\, d\Omega,$$

combining stored internal deformation energy with externally applied working forces to get the total energy of the system. By using the constitutive relation, the stored elastic energy W_{tot} can also be written as

$$W_{tot}(\mathbf{u}) \;=\; \frac{1}{2}\int_{\Omega} (\boldsymbol{\epsilon}(\mathbf{u}) : \mathbf{C} : \boldsymbol{\epsilon}(\mathbf{u}) - \mathbf{f}_{ext} \cdot \mathbf{u})\, d\Omega\,. \tag{3.9}$$

Since both relationships deformation-strain and strain-stress are linear, the resulting energy potential is a quadratic function in the deformation, leading to linear problems for statics and dynamics, enabling the use of efficient numerical solvers. However, this conceptual simplicity comes also at a price: linear theory is missing rotation invariance, i.e., material undergoing pure rotation should not generate any internal forces which can however not be captured with the linear formulation [Müller et al., 2002]. Nevertheless, we can apply this otherwise very convenient theory by introducing a modification of the basic theory. These modification is of discrete nature and will be discussed after the discretization has been introduced in Section 3.3.

Weak Form. For numerically solving such problems using a Galerkin discretization, a different viewpoint of the problem is considered, referred to as the *weak form* of the problem. This formulation can basically be found "on the

way", when deriving the strong form from the energy formulation by means of variational calculus and states the problem not in a pointwise manner but rather in an *averaged* sense.

One possibility to derive the weak form consists in starting with the elastic energy (3.9) and to state the condition for its minimum by asking that the *directional derivative* should vanish for all variations $\mathbf{v}$ of the deformation (called *test functions*, being possible direction vectors in a suitable functions space V). This can be formulated by the requirement that

$$\frac{dW_{tot}(\mathbf{u}+s\mathbf{v})}{ds}\Big|_{s=0} = 0 \qquad \forall \mathbf{v} \in V\,.$$

Further evaluation of this expression leads then to the weak form

$$\int_\Omega \boldsymbol{\epsilon}(\mathbf{u}) : \mathbf{C} : \boldsymbol{\epsilon}(\mathbf{v})\, d\Omega = \int_\Omega \mathbf{f}_{ext} \cdot \mathbf{v}\, d\Omega \qquad \forall \mathbf{v} \in V\,.$$

This form of the problem has some nice properties that are also important when performing the discretization. First, note that the left hand side expression is a symmetric positive definite (spd) bilinear form in both arguments $\mathbf{u}$ and $\mathbf{v}$, leading to well-behaving system matrices after discretizing. Second, as opposed to the strong form, the weak form only requires first derivatives of the solution leading to weaker restrictions for the solution space, i.e., candidate solutions.

To see the connection to the strong form, we can apply Green's first identity and by rearranging terms this leads to

$$\int_\Omega (\nabla \cdot (\mathbf{C} : \boldsymbol{\epsilon}(\mathbf{u})) - \mathbf{f}_{ext}) \cdot \mathbf{v}\, d\Omega = 0 \qquad \forall \mathbf{v} \in V\,.$$

Since this equation must hold for any direction $\mathbf{v} \in V$, we can conclude that also the strong form (3.6) must hold.

3.1.2 Nonlinear Elasticity

If we are willing to leave the linear theory whose physical validity is limited to the domain of small deformation we get into the realm of nonlinear elasticity which is more complex but also allows for a much more realistic description of a wide variety of scenarios. As for linear elasticity its formal description can be divided into geometry, forces and equilibrium, and constitutive relations. While the description of forces and equilibrium equations remain the same, the geometric measures and constitutive relations are more versatile and will be presented briefly. More information can be found in the excellent textbook [Bonet and Wood, 1997].

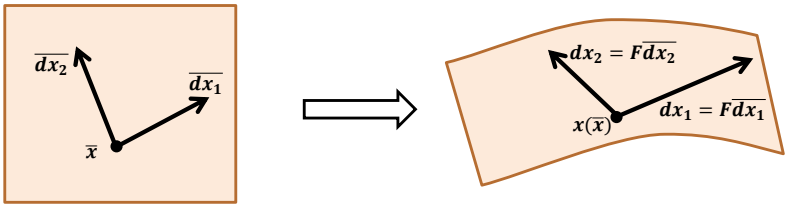


Figure 3.2: *The deformation gradient maps infinitesimal direction vectors from the undeformed to the deformed configuration.*

Geometry

Different than in the linear theory, we describe the deformed geometry now directly by the mapping $\mathbf{x}(\bar{\mathbf{x}})$, deforming domain points $\bar{\mathbf{x}} \in \Omega \subset \mathbb{R}^3$ and do not refer anymore to a displacement field $\mathbf{u}(\boldsymbol{\theta})$. However, also for the nonlinear case we will only discuss the Lagrangian viewpoint and introduce the necessary concepts used later on.

Deformation Gradient. The probably most important geometric quantity in nonlinear elasticity is the notion of the deformation gradient tensor

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \bar{\mathbf{x}}}, \tag{3.10}$$

describing how neighboring material particles are deformed relative to each other. For a given particle at position $\bar{\mathbf{x}}$, a neighbor located at $\bar{\mathbf{x}} + d\bar{\mathbf{x}}$ will be deformed to $\mathbf{x}(\bar{\mathbf{x}} + d\bar{\mathbf{x}})$. For the considered small distances $d\bar{\mathbf{x}}$, this can be approximated by a first order Taylor series as

$$\mathbf{x}(\bar{\mathbf{x}} + d\bar{\mathbf{x}}) \approx \mathbf{x}(\bar{\mathbf{x}}) + \mathbf{F}d\bar{\mathbf{x}} = \mathbf{x}(\bar{\mathbf{x}}) + d\mathbf{x},$$

showing up the relationship

$$d\mathbf{x} = \mathbf{F}d\bar{\mathbf{x}}.$$

Green Strain. The deformation gradient forms the basis for many measures of deformation. One of these interesting quantities is how length changes during the deformation, i.e., a measure of material stretch, but also by how much angles have changed during the deformation. As depicted in Fig. 3.2, both can be measured by considering the scalar products of two arbitrary direction vectors $d\bar{\mathbf{x}}_1$ and $d\bar{\mathbf{x}}_2$ after deformation

$$d\mathbf{x}_1 \cdot d\mathbf{x}_2 = d\bar{\mathbf{x}}_1 \cdot \mathbf{F}^T\mathbf{F}d\bar{\mathbf{x}}_2 = d\bar{\mathbf{x}}_1 \cdot \mathbf{C}d\bar{\mathbf{x}}_2, \tag{3.11}$$

leading to the so-called *right Cauchy-Green deformation tensor* $\mathbf{C} = \mathbf{F}^T\mathbf{F}$. Choosing identical direction vectors results in squared lengths of the deformed vectors, while choosing different direction vectors gives the new dot products that, together with the new lengths, leads to the new angles.

In order to measure the amount of stretch and shear occurring during the deformation, the given measures can simply be subtracted from each other, resulting in the well-known *Green strain tensor*

$$\mathbf{E} = \frac{1}{2}(\mathbf{C} - \mathbf{I}). \tag{3.12}$$

The Cauchy strain $\boldsymbol{\epsilon}$ defined in the linear theory (Eq. (3.2)) is simply the linearized Green strain $\mathbf{E}$.

While the deformation gradient is still linear in the deformed configuration $\mathbf{x}(\bar{\mathbf{x}})$, the right Cauchy-Green as well as the Green strain tensor are both quadratic in the deformation, which is also often referred as *geometric nonlinearity*.

Constitutive Relation

Moving from linear material models to nonlinear ones opens a multitude of different possibilities on how to describe elastic material behavior, ranging from simpler ones like the Saint Venant-Kirchhoff and the Neo-Hookean models [Bonet and Wood, 1997] up to newer and more complex ones like e.g. the Rubin-Bodner material model [Rubin and Bodner, 2002].

While these material models could be stated as relations between kinematic measures like Green strain or deformation gradient and suitable stress measures, it is more convenient for us to directly formulate the relation by stating the resulting energy densities, relating local deformation measures directly to scalar potentials. Having a formulation of the energy allows us to straightforwardly formulate the discrete problems by turning the continuous energies into discrete ones, such that the required forces and Hessians can be described in the discrete domain.

The use of powerful material models can be of great benefit in practice since they allow to further increase the realism of simulated materials and allow better capturing of specific deformation characteristics. While this is a very interesting research direction where computer graphics issued some fascinating work [Bickel et al., 2009], this thesis will just make use the more basic models, commonly used in graphics. Furthermore, the concepts presented throughout the thesis are not restricted to particular material models but are generally applicable.

Saint-Venant Kirchhoff. The *Saint-Venant Kirchhoff* material is among the simplest material models and extends the linear stress-strain relation of linear elasticity to the nonlinear regime. The strain energy density is simply

$$\Psi_{StVK} = \frac{\lambda}{2} tr(\mathbf{E})^2 + \mu tr(\mathbf{E}^2), \tag{3.13}$$

where λ and μ are the *Lamé constants* that describe the isotropic stiffness and (linearized) volume preservation properties of the modeled material.

Neo-Hookean. The *Neo-Hookean* material is another frequently used nonlinear isotropic hyperelastic material model. A particularly simple instance of this is the *compressible* Neo-Hookean model which shares characteristics and material parameters that are known from linear elasticity. The energy density is defined as

$$\Psi_{NH} = \frac{\mu}{2}(tr(\mathbf{C}) - 3) - \mu \, lnJ + \frac{\lambda}{2}(lnJ)^2. \tag{3.14}$$

Again, λ and μ are the material constants and $J = det\mathbf{F}$ measures the change in volume.

In the scope of this thesis, the nonlinear theory will become particularly important in the last part when directable simulations for elastic materials will be discussed. In there, the use of the correct full nonlinear geometric measures will be of great importance to correctly formulate the novel energy potentials.

3.1.3 Boundary Conditions and Collisions

While the two theories presented give a general description of the internal mechanics of elastic bodies, this does not yet allow simulating such objects interacting with a possible complex surrounding scenery, as it is required in practical applications. Two main points that require further discussion are the handling of boundary conditions and collisions. While the focus of this thesis did not lie on the particular handling of these, we only briefly discuss how these were realized in the different simulation frameworks.

Boundary Conditions. We used two different approaches to enforce Dirichlet boundary conditions (BCs) $\mathbf{u}(\boldsymbol{\theta}) = \mathbf{u}_{BC}(\boldsymbol{\theta}), \boldsymbol{\theta} \in \Gamma_{BC} \subset \Gamma = \partial\Omega$, depending on the kind of solution subspace that were used in the discretization.

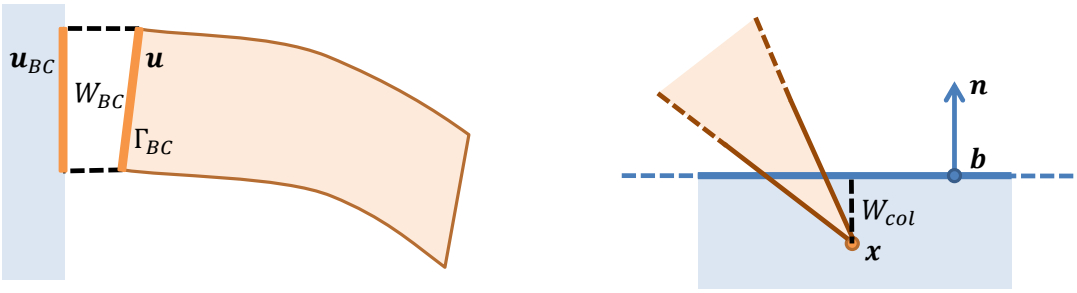


Figure 3.3: *Left: An potential W_{BC} glues the part Γ_{BC} of the boundary to a prescribed position $\mathbf{u}_{BC}$. Right: As soon as a vertex $\mathbf{x}$ penetrates the collision plane, a potential W_{col} is introduced to resolve the collision.*

While *hard constraints* are simple to formulate in the continuous case, they can be hard to enforce once a discretization has been chosen. For the solid theory we are discussing here, we can describe the set of candidate solution functions by the Sobolev space H^1, i.e., candidate functions must be C^1-continuously differentiable up to a set of size 0 (requiring first derivatives of the deformation to compute strain). In order to enforce BCs, the candidate solution space can be restricted by requiring that it only contains candidates fulfilling the BCs. Depending on the chosen discretization, an analogous enforcement on the finite solution space is not straightforward such that a second approach can be chosen.

BCs can also be enforced *weakly* such that they are only fulfilled approximately. In this case, the actual solution space is left untouched and the boundary condition is modeled as an additional elastic energy, gluing the solution to the intended goal position (see also left part of Fig. 3.3). This can be realized by a simple quadratic energy of the form

$$W_{BC}(\mathbf{u}) = \frac{\beta}{2} \int_{\Gamma_{BC}} |\mathbf{u} - \mathbf{u}_{BC}|^2, \tag{3.15}$$

where $\mathbf{u}_{BC}$ describes the desired displacement of the fixed boundary Γ_{BC}. Furthermore, the magnitude of the penalty parameter β steers the enforcement of the BC. This is commonly used technique also used in the context of meshless methods in computational mechanics [Fries and Matthies, 2004].

Collisions. They are of foremost importance when physically modeling complex sceneries and are still an active field of research. For simplicity, we only incorporated penalty-based planar and spherical collisions in our framework which were enough to demonstrate the feasibility of our methods.

For planar collisions, we first detect a collision by testing the surface points of the deformed body against a given plane defined by the point $\mathbf{b}$ and normal $\mathbf{n}$ (see also right part of Fig. 3.3). For colliding points $\mathbf{x}$, we then define the simple quadratic penalty potential

$$W_{plane} = \frac{\gamma}{2}((\mathbf{x} - \mathbf{b}) \cdot \mathbf{n})^2, \tag{3.16}$$

which will resolve the collision. Collisions with spheres are handled in an analogous manner by considering distances to spherical surfaces. Again, this approach only resolves collisions weakly but leads to stable and straightforward incorporation into implicit time-stepping schemes.

3.2 Resultant-based Formulations

So far we discussed elements of the linear and nonlinear theory of elasticity that describes the general mechanics of elastic material. For thin geometries, however, there exist specialized variants of this theory that take account of these particular geometric circumstances and which are more efficient and numerically better suited than the classic models.

If a material has only small extent in certain spatial directions, the mechanics can be simplified by making certain assumption on how the material can deform in these directions leading to so-called *resultant-based models*. If the reduction takes place along one direction, i.e., the material becomes membrane-like, we talk of *thin shell* theories. If the reduction takes place along two directions, we talk about *rod* theories.

3.2.1 Shells

Let us recall the material description given by curvilinear coordinates, where $\mathbf{x}(\bar{\boldsymbol{\theta}})$ describes the undeformed and $\mathbf{x}(\boldsymbol{\theta})$ the deformed configuration. If we consider a volumetric surface-like solid whose extent along θ_1 and θ_2 (the "tangent directions") is much greater than along θ_3 (the "normal direction"), we arrive at the special case of *thin shells*. In this case, a naïve discretization using linear tetrahedral FEs leads to arbitrarily poor numerical conditioning and slow convergence to the smooth limit (*locking*) and wastefully allocates DOFs along the normal direction [Yang et al., 2000].

These difficulties motivate the development of *resultant-based formulations*, where thin bodies are treated by a reduced set of equations governing a lower-dimensional object—the thin shell *middle surface*—and specifying the

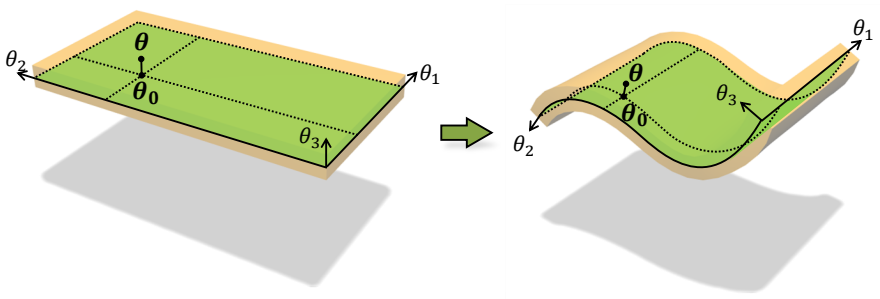


Figure 3.4: *The undeformed thin block of material is described by its midsurface (green) and normal information.*

displacement field both on the surface and in its (normal) vicinity. The reduced representation offers superior numerical conditioning, convergence, and more efficient allocation of DOFs [Naghdi, 1972].

Strain About Middle Surface. For notational convenience, let the middle surface $\mathcal{S}$ be parameterized by the material-domain surface $\boldsymbol{\theta}_0 = (\theta_1, \theta_2, 0)$, as depicted in Fig. 3.4. Assuming the shell to be sufficiently thin in normal direction, we expand to first-order the positions and displacements:

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

In the view of the linear elasticity theory, we can substitute this approximation into the linear Cauchy strain (3.2) yielding the strain about the mid-surface

$$\boldsymbol{\epsilon}(\boldsymbol{\theta}) \approx \boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \theta_3 \boldsymbol{\beta}^3(\boldsymbol{\theta}_0), \tag{3.17}$$

expressed in terms of the *membrane strain*

$$\boldsymbol{\alpha}_{ij} = \frac{1}{2}\left(\mathbf{u}_{,i} \cdot \bar{\mathbf{x}}_{,j} + \bar{\mathbf{x}}_{,i} \cdot \mathbf{u}_{,j}\right) \tag{3.18}$$

and the *bending strain* related to direction θ_k

$$\boldsymbol{\beta}^k_{ij} = \frac{1}{2}\left(\mathbf{u}_{,ik} \cdot \bar{\mathbf{x}}_{,j} + \bar{\mathbf{x}}_{,i} \cdot \mathbf{u}_{,jk} + \mathbf{u}_{,i} \cdot \bar{\mathbf{x}}_{,jk} + \bar{\mathbf{x}}_{,ik} \cdot \mathbf{u}_{,j}\right). \tag{3.19}$$

Equation (3.17) is an approximation since higher order terms in the thin direction θ_3 have been discarded from (3.19). An assumption being valid for sufficiently thin diameters. In the master thesis of Jeronimo Bayer [2011], a similar procedure has been performed for the nonlinear Green strain measure (3.12) to develop the thin shell equations for the nonlinear theory. For our purpose, the describing the linear derivation is sufficient.

Energy Integration. The elastic energy of the volumetric shell model can be derived straightforwardly from (3.9) by replacing the small strain $\boldsymbol{\epsilon}$ by our approximation (3.17):

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

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

$$W \;=\; \frac{h_3}{2}\int_{\mathcal{S}} \boldsymbol{\alpha} : \mathbf{C} : \boldsymbol{\alpha} \;+\; \frac{h_3^2}{12}\,\boldsymbol{\beta}^3 : \mathbf{C} : \boldsymbol{\beta}^3\,\mathrm{d}\mathcal{S}\,. \tag{3.20}$$

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

This resulting model for the energy basically characterizes thin shell behavior for a solution field $\mathbf{u}$ that also holds information in normal direction to the shell's midsurface, i.e., whose solution field is still volumetric around the reduced geometry. This model will become particularly important in Chapter 5 where it will form the base for the derivation of unified model for deformable objects.

Kirchhoff-Love Shells. Note that $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}^k$ are volumetric 3×3 tensors so far. At this point in the derivation, a typical resultant-based formulation would assume that a normal vector to the undeformed mid-surface is deformed such that it retains its orthogonality to the mid-surface [Cirak et al., 2000]. This *Kirchhoff-Love* assumption yields vanishing normal components for the strain, reducing the 3×3 volumetric to 2×2 surface strain tensors, i.e., instead of $2 \cdot 6$ only $2 \cdot 3$ strain measures are taken into account. While the membrane strain entries of Eq. (3.18) stay the same, the bending strain becomes

$$\begin{aligned}\boldsymbol{\beta}^k_{ij} \;\;=\;\; & -\mathbf{u}_{,ij}\cdot\bar{\mathbf{x}}_{,3} + \frac{1}{D}(\mathbf{u}_{,1}\cdot(\bar{\mathbf{x}}_{i,j}\times\bar{\mathbf{x}}_2) + \mathbf{u}_{,2}\cdot(\bar{\mathbf{x}}_1\times\bar{\mathbf{x}}_{i,j})) \\ & + \frac{\bar{\mathbf{x}}_3\cdot\bar{\mathbf{x}}_{i,j}}{D}(\mathbf{u}_{,1}\cdot(\bar{\mathbf{x}}_2\times\bar{\mathbf{x}}_3) + \mathbf{u}_{,2}\cdot(\bar{\mathbf{x}}_3\times\bar{\mathbf{x}}_1)),\end{aligned} \tag{3.21}$$

where $D = |\bar{\mathbf{x}}_1 \times \bar{\mathbf{x}}_2|$ describes the elemental surface area.

Note that a similar derivation can also be performed for nonlinear elasticity, by linearizing the Green strain around the mid-surface. The resulting membrane and bending strains then correspond to the first and second fundamental forms [Terzopoulos et al., 1987] which build the basis for nonlinear shell models. When performing, in addition, a linearization in the deformation variable $\mathbf{u}$, the model described above is again recovered.

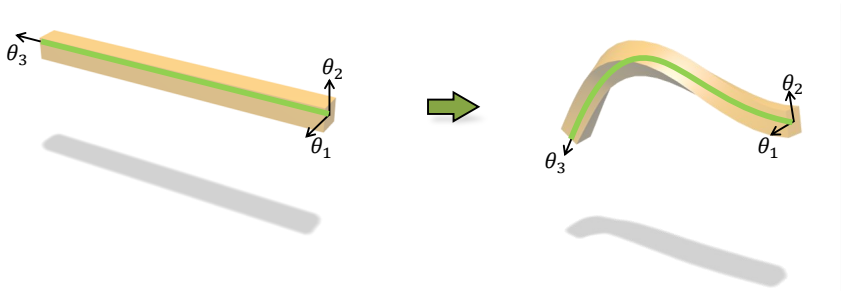


Figure 3.5: *The rod is described only by its centerline (green) and normal information.*

3.2.2 Rods

Consider next a volumetric curve-like solid whose extent along θ_1 (the "tangent direction") is much greater than along θ_2 and θ_3 (the "normal directions" spanning the "cross-sectional plane").

We derive this special case of thin rods analogously, identifying the reduced geometry (the *centerline* curve), linearizing strain about the centerline (this time along the *two* directions spanning the cross-sectional neighborhood), and again omitting the collapse of the strain tensor into a lower dimension.

Strain About Centerline. For notational convenience, let the centerline curve Γ be parameterized by the material-domain curve $\boldsymbol{\theta}_0 = (\theta_1, 0, 0)$ (Fig. 3.5). For small extents along both normals θ_2 and θ_3, we linearly approximate positions and displacements by

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

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

$$\boldsymbol{\epsilon}(\boldsymbol{\theta}) \approx \boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \theta_2 \boldsymbol{\beta}^2(\boldsymbol{\theta}_0) + \theta_3 \boldsymbol{\beta}^3(\boldsymbol{\theta}_0), \tag{3.22}$$

where the bending strains $\boldsymbol{\beta}^2$ and $\boldsymbol{\beta}^3$ are defined as in (3.19).

Remark on Twist. The twist of a rod can be computed as $\mathbf{x}_{,12} \cdot \mathbf{x}_{,3}$ or $\mathbf{x}_{,13} \cdot \mathbf{x}_{,2}$, where the two values are identical (up to their sign) under the Kirchhoff assumptions. The difference in twist between the deformed and undeformed configuration can be measured as $\mathbf{x}_{,12} \cdot \mathbf{x}_{,3} - \bar{\mathbf{x}}_{,12} \cdot \bar{\mathbf{x}}_{,3}$. Using (3.1) and linearizing in $\mathbf{u}$ this becomes $\mathbf{u}_{,12} \cdot \bar{\mathbf{x}}_{,3} + \mathbf{u}_{,3} \cdot \bar{\mathbf{x}}_{,12}$. From (3.19) we see that these

quantities are part of the strain entries $\boldsymbol{\beta}^2_{13}$, $\boldsymbol{\beta}^2_{31}$, $\boldsymbol{\beta}^3_{12}$, and $\boldsymbol{\beta}^3_{21}$, showing that the bending strain also incorporates a measurement for twisting deformations.

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

$$W = \frac{h_2 h_3}{2} \int_\Gamma \boldsymbol{\alpha} : \mathbf{C} : \boldsymbol{\alpha} + \frac{h_2^2}{12} \boldsymbol{\beta}^2 : \mathbf{C} : \boldsymbol{\beta}^2 + \frac{h_3^2}{12} \boldsymbol{\beta}^3 : \mathbf{C} : \boldsymbol{\beta}^3 \, \mathrm{d}\Gamma, \tag{3.23}$$

where h_2 and h_3 denote the thickness of the rod in the two normal directions θ_2 and θ_3, respectively. Again, higher order terms in the thin directions θ_2 and θ_3 are discarded. As for shells, this model is an important step toward the unified formulation stated in Chapter 5.

Kirchhoff Rods. The resultant-based formulation for rods found so far allows still a rich number of deformations that the rod can undergo, e.g. cross-sectional shearing and stretch can be captured by the resulting strain measures. If we now again invoke the Kirchhoff assumption of normals staying normal during deformation and additionally require axial inextensibility, the effectively occurring strains reduce dramatically: The $3 \cdot 6$ strain measures reduce to just 3 strains. What they basically measure is bending in the two normal directions and twist along the tangent direction.

This becomes clear by the following observation: Besides the description of the rod's reduced geometry (the curve $\Gamma(s)$), each point on the curve has now an associated *adapted* orthonormal frame $\{\mathbf{t}, \mathbf{m}_1, \mathbf{m}_2\}$ resulting from the Kirchhoff assumption. Adapted means that the axis $\mathbf{t}$ corresponds to the curve's tangent $\Gamma'(s)$. The rotation of this frame along the curve can be compactly described by the so-called *Darboux vector* $\boldsymbol{\Omega} = m\mathbf{t} - m_2\mathbf{m}_1 + m_1\mathbf{m}_2$ relating the frame to its rate of change along the curve by $\mathbf{t}' = \boldsymbol{\Omega} \times \mathbf{t}$, $\mathbf{m}_1' = \boldsymbol{\Omega} \times \mathbf{m}_1$ and $\mathbf{m}_2' = \boldsymbol{\Omega} \times \mathbf{m}_2$. Using this (now nonlinear) strain measures for the deformed and undeformed curves and adapted frames leads to the following simple elastic energy

$$W_{krod} = \frac{1}{2} \int_\Gamma \alpha_1 (m_1 - \bar{m}_1)^2 + \alpha_2 (m_2 - \bar{m}_2)^2 + \beta (m - \bar{m})^2 ds, \tag{3.24}$$

with material parameters α_1, α_2 and β that can be associated to the usual material parameters of linear materials and thicknesses of the rods.

3.3 Discrete Solution Representation

So far we saw how the behavior of elastic material of different types and geometries can be modeled by the use of continuum mechanical principles. An important step that has to be performed next is to transform these *continuous* problems into their *discrete* equivalents. The spatial discretization will be performed in a *Galerkin*-type manner, being a standard approach having important theoretical properties. This type of discretization will be discussed in some detail since a lot of the presented work in the thesis bases on this type of discretization. For dynamic problems, the solution needs also be discretized along the time dimension. For this part we will just shortly present some of the standard approaches we also applied in our framework.

3.3.1 Space Discretization for Linear Formulations

Solution Space. A first important step when we want to solve a PDE with a Galerkin-type approach is to understand what properties the PDE obliges the solution to take, i.e., what properties a function must have in order to be a candidate solution [Hughes, 2000].

Considering the energy formulation (3.9) or alternatively, the weak form (3.10) of the linear elasticity problem, reveals an important property: Opposed to the strong form (3.6), no differentiation of the stress tensor (or the strain tensor, when using a linear material model) is necessary in this formulation, i.e., the solution function needs only be continuously differentiable once. Instead of formulating the problem in a *point-wise* manner, which puts higher requirements on possible solution functions, the weak form states the problem in a *averaged* sense, allowing to reduce the continuity requirement. Furthermore, differentiation itself can also be formulated weakly (*weak derivatives*, [Hughes, 2000]) which, besides nice theoretical properties, has the practical consequence that the solution function needs not to be continuously differentiable on the whole domain, but is allowed to not be so on a subdomain of size zero (i.e., on lower dimensional subdomains). Altogether, this leads to our solution space which is the (infinite-dimensional) Sobolev space $H^1(\Omega)$, defined by all L^2-integrable functions with L^2-integrable derivatives over the domain Ω.

A very important result from functional analysis is given by the *Lax-Milgram theorem*. Assuming a bilinear form $a(u,v)$, a linear form $f(v)$ and $u,v \in V$ with a Hilbert space V, it states that the problem

$$a(u,v) = f(v) \quad \forall v \in V \tag{3.25}$$

has a unique solution u and fulfills a stability estimate restricting the solution u by the right hand side $f(v)$ [Hughes, 2000]. Important for us, this theorem only assumes a Hilbert space structure for the solution space V, leading to a very general statement. If we revisit our weak form Eq. (3.10), we note that our problem is exactly of this form, with $a(u,v) = \int_\Omega \boldsymbol{\epsilon}(u) : \mathbf{C} : \boldsymbol{\epsilon}(v)$ and $f(v) = (\mathbf{f}_{ext}, v) = \int_\Omega \mathbf{f}_{ext} \cdot v$, such that the theorem applies.

Problem Formulation in Subspace. The mentioned generality of the Lax-Milgram Theorem is an essential part for formulating the discrete problem: Since the Hilbert space V is the infinite dimensional $H^1(\Omega)$ in our formulation, it follows that the theorem is also valid for any *finite* dimensional subspace $V_N \subset H^1(\Omega)$ which actually can be represented in a computer, i.e., with a finite amount of memory. That is, we can restrict the solution space to an arbitrary subspace and still retain the existence and stability guarantees of the theorem.

While the Lax-Milgram theorem gives us these guarantees for a solution in the subspace, it does not yet tell us how good the solution actually will be. Therefore, a second important result from functional analysis is considered, *Cea's Lemma*, that roughly states that the solution u_N of the problem formulated in the subspace

$$a(u_N, v_N) = f(v_N) \quad \forall v_N \in V_N \tag{3.26}$$

is actually the *best* possible approximation (up to a constant factor depending on the error norm — being 1 in case of the energy norm) [Hughes, 2000].

In summary, for linear problems, the Lax-Milgram Theorem and Cea's Lemma guarantee to find the best approximation in a given solution subspace and constitute the theoretical basis for Galerkin methods.

Geometric Viewpoints. We can also consider the presented results from a geometric viewpoint, giving some more intuition over the underlying principle of the Galerkin method. We will provide three (slightly) different views to shed more light on the topic.

We start with noting that the relation stated in Eq. (3.25) is also valid if we only consider test functions v that lie in the subspace $V_N \subset V$. By further subtracting Eq. (3.26), we get the following relation

$$a(u - u_N, v_N) = 0,$$

stating, that in the energy scalar product $a(\cdot,\cdot)$, the error vector between the analytical and the approximated solution is orthogonal to the solution space.

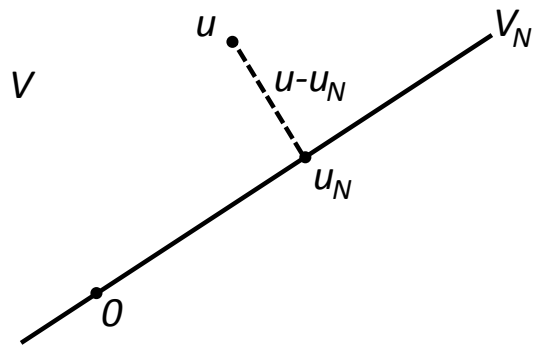


Figure 3.6: *The solution u_N is chosen such that the discretization error $u - u_N$ becomes orthogonal to the solution subspace V_N.*

That is, in the energy norm $|v|_a^2 = a(v,v)$ the found approximate solution is the 'closest' to the analytical one. This is often referred to as *Galerkin orthogonality* (see also Fig. 3.6).

With the same argument of orthogonality, the discrete formulation of the weak form can be derived from the strong form. This is sometimes also referred as the *method of weighted residuals* [Fries and Matthies, 2004]. We can first state the strong form abstractly as $Lu = f$ with L being a linear differential operator. For an approximate solution $u_N \in V_N$, we can consider the residual $r_N = Lu_N - f$ and require that it should be orthogonal to the solution subspace V_N (in the L^2 sense), leading to

$$\int_\Omega r_N v_N = \int_\Omega (Lu_N - f) v_N = 0 \quad \forall v_N \in V_N, \tag{3.27}$$

corresponding exactly to the discrete version of the weak form (after applying again Green's identity). Note that if the same subspaces are chosen for the solution and test space this is called a *Bubnov-Galerkin* method. If the two spaces do not coincide this is referred to as a *Petrov-Galerkin* method [Fries and Matthies, 2004]. Further note that in this second view, orthogonality is stated in the classic L^2 sense between the *residual* and test space, while in the former we stated the orthogonality directly on the solution itself but employed the energy scalar product. Both views are equivalent and are just different illustrations of the same concept.

Of course, also our view from an energy minimization perspective is still valid. The role of the weak form as necessary conditions for an energy minimum is also true in the subspace: The candidate with the smallest energy in the subspace is selected. This can be seen easily by formulating the abstract

energy

$$W(u_N) = \frac{1}{2}a(u_N, u_N) - f(u_N) \tag{3.28}$$

and equaling the directional derivative $\frac{d}{dt}W(u_N + tv_n)$ to zero for all $v_N \in V_N$ which gives us again the discrete weak form Eq. (3.26). This last view is particularly useful in practice since only a system's energy has to be discretized and the following force and Jacobian computations can more easily be performed in the discrete, finite dimensional setting.

Discrete Formulation. To derive the discrete formulation of the problem that can effectively be solved using numerical methods, we first need a *representation* of the finite dimensional space $V_N \subset H^1$. Using a basis of shape functions $N_i(\bar{\mathbf{x}})$, every function in the subspace can be represented as a linear combination of the basis with the actual *degrees of freedom* $\mathbf{u}_i$, giving

$$\mathbf{u}_N(\bar{\mathbf{x}}) = \sum_i^N \mathbf{u}_i N_i(\bar{\mathbf{x}}) \in V_N. \tag{3.29}$$

Note that the basis functions $N_i(\bar{\mathbf{x}})$ (usually associated with a corresponding position $\bar{\mathbf{x}}_i \in \Omega$) must lie in H^1, i.e., must form a proper basis of a subspace of H^1. Geometrically, the basis functions work as interpolation devices defining the spatial influence of each associated DOF on its neighborhood.

Furthermore, basis functions must also fulfill the *completeness* property [Hughes, 2000]. For solid elasticity problems, these correspond to the two following requirements:

- *Constant Reproduction* In order to represent arbitrary translations of the body, the solution space must be able to represent constant vector fields. This property can be fulfilled in different manners. One possibility is that the basis contains a constant basis function 1 (with an associated DOF). However, the global support of such a basis function is corrupting the sparsity of resulting system matrices and therefore has a negative influence on the performance of the linear system solve. The better choice is to "distribute" this property among the basis functions and requiring that (at least a part of) the basis forms a *partition of unity* (PU), i.e., suffices the requirement $\sum_i N_i(\bar{\mathbf{x}}) = 1$.
- *Linear Reproduction* It is also necessary for a basis to represent constant strain fields as well as arbitrary rigid body motions. Next to the translation, it means that also rotations of an object should be perfectly representable. Since a global rotation of an object is a linear function in positions, the basis is also required to reproduce linear functions.

Again, this property can either be enforced by specific global basis functions leading to performance issues or as a property of the entire basis.

As mentioned earlier, there are different possibilities on how to derive the discrete problem, but we will just present the energy-based approach here. Inserting our solution representation of Eq. (3.29) into the energy of Eq. (3.28) and making use of the (bi-) linearity, leads to the discrete energy

$$\begin{aligned} W_N(\mathbf{u}_N) &= \frac{1}{2}a(\sum_i \mathbf{u}_i N_i, \sum_j \mathbf{u}_j N_j) - f(\sum_i \mathbf{u}_i N_i) \\ &= \sum_{ij} \mathbf{u}_i \mathbf{u}_j a(N_i, N_j) - \sum_i \mathbf{u}_i f(N_i) \\ &= \mathbf{u}^T \mathbf{K} \mathbf{u} - \mathbf{u}^T \mathbf{f}_{ext}, \end{aligned} \tag{3.30}$$

where $\mathbf{K}_{ij} = a(N_i, N_j)$, $\mathbf{f}_{ext,i} = f(N_i)$ and $\mathbf{u}$ is a $3N$-dimensional vector of the concatenated 3-vectors $\mathbf{u}_i$. Please note here that we are using a simplified notation to handle the dimensionality of the solution function. Whenever $a(\cdot,\cdot)$ takes two scalar arguments, it still evaluates to a 3×3 matrix $\mathbf{M} = a(s,t)$ with entries $\mathbf{M}_{kl} = a(s\,\mathbf{e}_k, t\,\mathbf{e}_l)$. Likewise, the vector $\mathbf{f} = f(s)$ has entries $\mathbf{f}_i = f(s)\mathbf{e}_i$.

In the same manner as internal forces could be computed from the energy by its variational derivative in the continuous case, the discrete internal forces can simply be computed as $\mathbf{f}_i = -\frac{\partial W_N}{\partial \mathbf{u}_i}$, leading to the linear system

$$\mathbf{K}\mathbf{u} = \mathbf{f}_{ext} \tag{3.31}$$

for the static case and to the system of ODEs

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{f}_{ext} \tag{3.32}$$

for the dynamic case. The *mass matrix* $\mathbf{M}$ in the momentum term stems from the discretization of the weak form of the continuous momentum term $\int_\Omega \rho \ddot{\mathbf{u}} \mathbf{v}$ and consists of 3×3 blocks

$$\mathbf{M}_{ij} = \mathbf{I} \cdot \int_\Omega \rho N_i N_j \, d\Omega\,. \tag{3.33}$$

Alternatively, the mass matrix can also be "lumped" into diagonal matrix, represented as mass vector $\mathbf{m}$, i.e., all matrix rows are condensed to a singular scalar mass value per DOF by computing

$$\mathbf{m}_i = \sum_j M_{ij}.$$

This is a popular choice made in graphics since it allows to considerably increase efficiency for explicit time integration schemes, however can also lead to problems for non-nodal basis functions.

If required, an additional damping term could also be added to model energy dissipation. However, since we will e mainly using a stable implicit or semi-implicit Euler integration schemes [Baraff and Witkin, 1998], there is no need for an additional damping term.

Choice of Subspace. Of course, the actual choice of the subspace is of great importance for practical implementations of the Galerkin method and influences different aspects of the resulting algorithms [Grinspun, 2003]. Let us briefly discuss some of them:

- **Locality and Efficiency** The spatial support of a basis function crucially influences the efficiency of the numerical solution procedures. The larger the support of basis functions are, the denser becomes the coupling between the individual DOFs since more of them define the solution at a given point. As a result, the Hessian of the energy (stiffness matrix $\mathbf{K}$) becomes denser. In graphics, such globally supported bases are sometimes used in model reduction techniques with few DOFs [James and Fatahalian, 2003], but generally bases are preferred to have local support.
- **Resolution, Order and Convergence** The dimensionality of the subspace is another important choice considerably affecting how close the approximated solutions come to the analytical ones. Naturally, this choice also influences the performance of setting up the linear systems and their numerical solving. Therefore a tradeoff between accuracy and performance has to be made. Next to the resolution also the *order* of the basis function plays an important role since it governs the approximation power of the basis. For a polynomial basis, e.g., higher order polynomials lead to better convergence rates, i.e., the approximate solution approaches the analytical one faster when increasing the subspace dimension.
- **Shape of Basis Functions** The shape of a basis function is a further important choice which we explore in Chapter 4. Depending on the problem, *prior* knowledge can be incorporated into the solution space such that accurate solutions can already be computed only with few DOFs. Model reduction techniques [Choi and Ko, 2005; Barbič and James, 2005] or the enrichment textures of [Kaufmann et al., 2009b] are two good examples where such prior knowledge is incorporated into the solution space. Also in absence of such knowledge, proper design of "general-purpose" basis functions allows to

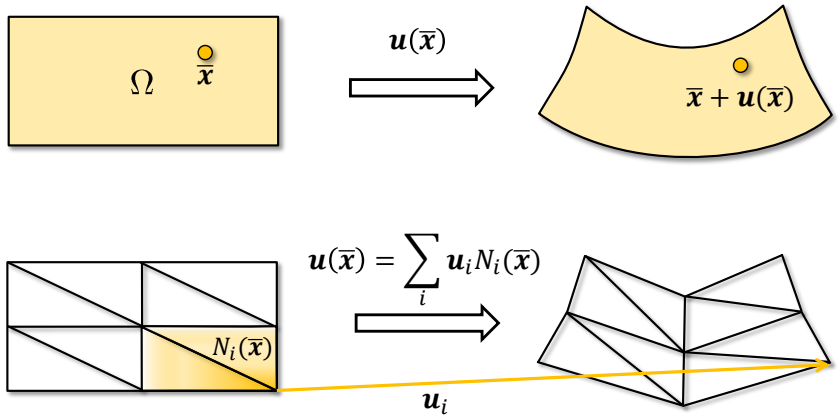


Figure 3.7: *The FEM discretizes the domain Ω into elements upon which the continuous deformation field is approximated in a piecewise manner.*

increase considerably the flexibility of the resulting algorithms and incorporate other geometric properties.

- **Nodal vs. Non-nodal** Basis functions $N_i(\bar{\mathbf{x}})$ are associated to DOFs $\mathbf{u}_i$. If they are *nodal*, it holds that $N_i(\bar{\mathbf{x}}_j) = \delta_{ij}$, i.e., the approximation is actually interpolating the $\mathbf{u}_i$ at their nodal location $\bar{\mathbf{x}}_i$. This property can be quite useful e.g. for simplifying rendering (where the solution vectors $\mathbf{u}_i$ can directly be used to deform the visualization mesh) or when enforcing boundary conditions (where individual DOFs can directly be fixed). For a non-nodal basis these tasks become more involved.

FEM. The de facto standard approach for a Galerkin discretization is the famous *finite element method (FEM)*. The FEM is a specific instantiation of the Galerkin methodology we discussed so far and mainly proposes an approach on how to construct the actual solution space. As the name suggests, the object is partitioned into *finite elements* e, i.e., $\bigcup e = \Omega$, and $\mathbf{u}(\bar{\mathbf{x}})$ is approximated by interpolating the displacements $\mathbf{u}_i$ of the nodes $\bar{\mathbf{x}}_i$ within elements (see Fig. 3.7). The interpolated displacement $\mathbf{u}^e$ in an element e of k nodes is

$$\mathbf{u}(\bar{\mathbf{x}})|_e \approx \mathbf{u}^e(\bar{\mathbf{x}}) := \sum_{i=1}^{k} \mathbf{u}_i N_i^e(\bar{\mathbf{x}}),$$

where the shape functions $N_i^e = N_i|_e$ determine the influence of the nodal displacements $\mathbf{u}_i$ inside the element. From the gradient $\nabla \mathbf{u}^e(\bar{\mathbf{x}})$ one computes

strain and stress such that the elastic energy stored in the deformed element

$$\mathrm{W}_e(\mathbf{u}^e) \;=\; \frac{1}{2}\int_e \boldsymbol{\epsilon}(\mathbf{u}^e) : \mathbf{C} : \boldsymbol{\epsilon}(\mathbf{u}^e)\; \mathrm{d}\mathbf{x},$$

can be accumulated into the total energy $\mathrm{W}(\mathbf{u}) = \sum_e \mathrm{W}_e(\mathbf{u}^e)$.

As we have seen earlier, the basis functions N_i have to be in the Sobolev space H^1. In the special case of linear FEM they have to be C^1 smooth within elements and C^0 continuous across element boundaries. The basis functions also need to exactly reproduce constant and linear functions (e.g., rigid body motions). These conditions are satisfied by linear shape functions for tetrahedra and trilinear shape functions for hexahedra — the element types and basis functions most frequently used in computer graphics. However, since those require complex remeshing for topological changes, we will introduce more flexible shape functions for general polyhedral elements in Chapter 4.

Quadrature. When considering simple basis functions as in the linear FEM, energy integration can be performed analytically. However for more complex basis functions it is often not possible to perform the energy integration analytically such that numerical quadrature schemes of the form

$$W(\mathbf{u}(\bar{\mathbf{x}})) = \int \Psi(\mathbf{u}(\bar{\mathbf{x}}))dV \approx \sum_q \Psi(\mathbf{u}(\bar{\mathbf{x}}_q))V_q \tag{3.34}$$

with $\Psi(\cdot)$ the energy density, $\bar{\mathbf{x}}_q$ the quadrature points, and V_q the quadrature weights have to be considered. Throughout this thesis we usually use Gauss quadrature on background meshes or Monte Carlo-type integration [Press et al., 2007] which we will discuss individually in the next chapters. In terms of implementation, this leads to a quadrature-centric view on the problem where basis functions, deformations, and local stiffness matrices are computed pointwise and the later then assembled into the global system matrices.

3.3.2 Discrete Handling of Boundary Conditions and Collisions

Boundary Conditions. For simulations employing nodal basis functions (i.e., $\mathbf{u}(\bar{\mathbf{x}}_i) = \mathbf{u}_i$) prescribing positional or *Dirichlet* boundary constraints simply corresponds to fixing individual vertices. However, if basis functions are not interpolating, a different approach for imposing boundary conditions is required. In these cases, we employ a penalty method since it is

simple to implement, gives satisfactory results, and does not introduce additional DOFs into the system (opposed to approaches employing Lagrange Multipliers [Fries and Matthies, 2004], for example).

A target displacement $\mathbf{u}_{BC}$ can be imposed on a subdomain $\Gamma_{BC} \subset \Gamma$ by adding the term in Eq. (3.15) to the elastic energy that penalizes the deviation from the prescribed displacement. Using the solution representation $\mathbf{u}(\bar{\mathbf{x}}) = \sum_i \mathbf{u}_i N_i(\bar{\mathbf{x}})$, this leads to a discrete penalty energy

$$\begin{aligned} W_{BC} &= \frac{\gamma}{2}\int_{\Gamma_{BC}} (\mathbf{u} - \mathbf{u}_{BC})^2 \, d\Gamma \\ &= \frac{\gamma}{2}\int_{\Gamma_{BC}} (\mathbf{u}_i N_i - \mathbf{u}_{BC})^2 \, d\Gamma \\ &= \frac{\gamma}{2}\mathbf{u}_i(\int_{\Gamma_{BC}} N_i N_j \, d\Gamma)\mathbf{u}_j - \gamma \mathbf{u}_i(\int_{\Gamma_{BC}} N_i \mathbf{u}_{BC} \, d\Gamma) + \frac{\gamma}{2}\int_{\Gamma_{BC}} (\mathbf{u}_{BC}^2 \, d\Gamma) \\ &= \frac{1}{2}\mathbf{u}^T \mathbf{K}_{BC}\mathbf{u} - \mathbf{u}^T \mathbf{f}_{BC} + c, \end{aligned}$$

where $\mathbf{K}_{BC,ij} = \gamma \mathbf{I} \int_{\Gamma_{BC}} N_i N_j d\Gamma$ and $\mathbf{f}_{BC,i} = \gamma \int_{\Gamma_{BC}} N_i \, \mathbf{u}^c d\Gamma$. Again, for general basis functions, these integrals have to be approximated by a quadrature rule. Since we were enforcing boundary conditions on the triangular visualization mesh, we approximated the integral using the mesh's vertices and their associated area from the neighboring triangles as quadrature weights.

Collisions. In a similar manner we can also find the discrete potential for the collision response forces, as soon as a collision has been detected for a material point at $\mathbf{x}_c = \bar{\mathbf{x}}_c + \mathbf{u}(\bar{\mathbf{x}}_c)$. Similarly as for the boundary conditions, inserting the solution representation into the plane collision potential (Eq. (3.16)) leads to

$$\begin{aligned} W_{col} &= \frac{\gamma}{2}((\mathbf{x}_c - \mathbf{b})^T \mathbf{n})^2 \\ &= \frac{\gamma}{2}((\bar{\mathbf{x}}_c + \mathbf{u} - \mathbf{b})^T \mathbf{n})^2 \\ &= \frac{\gamma}{2}(\mathbf{u}^T \mathbf{n} + (\bar{\mathbf{x}} - \mathbf{b})^T \mathbf{n})^2 \\ &= \frac{\gamma}{2}(\mathbf{u}^T \mathbf{n})^2 + 2(\mathbf{u}^T \mathbf{n})((\bar{\mathbf{x}} - \mathbf{b})^T \mathbf{n}) + ((\bar{\mathbf{x}} - \mathbf{b})^T \mathbf{n})^2 \\ &= \frac{\gamma}{2}((\mathbf{u}_i N_i)^T \mathbf{n})^2 + 2((\mathbf{u}_i N_i)^T \mathbf{n})((\bar{\mathbf{x}} - \mathbf{b})^T \mathbf{n}) + ((\bar{\mathbf{x}} - \mathbf{b})^T \mathbf{n})^2 \\ &= \frac{\gamma}{2}\mathbf{u}_i(\mathbf{n}\mathbf{n}^T N_i N_j)\mathbf{u}_j + \gamma \mathbf{u}_i(\mathbf{n}\mathbf{n}^T(\bar{\mathbf{x}} - \mathbf{b})N_i) + c \\ &= \frac{\gamma}{2}\mathbf{u}^T \mathbf{K}_{col}\mathbf{u} + \gamma \mathbf{u}^T \mathbf{f}_{col} + c, \end{aligned} \tag{3.35}$$

where $\mathbf{K}_{col,ij} = (\mathbf{n}\mathbf{n}^T N_i N_j)$ and $\mathbf{f}_{col,i} = (\mathbf{n}\mathbf{n}^T(\bar{\mathbf{x}} - \mathbf{b})N_i)$. As for boundary conditions, the collision potential can simply be added to the elastic energy.

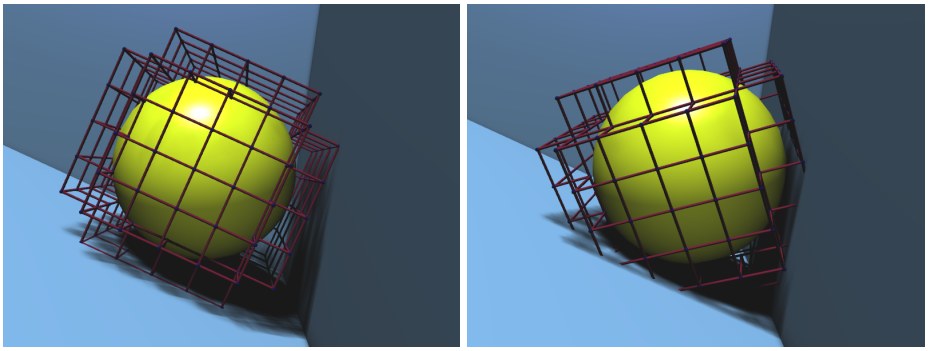


Figure 3.8: *Left: Collision handling on the simulation mesh's nodes. Right: Collision handling on the embedded surface.*

In this manner the static and dynamic solvers are agnostic to the handling of BCs and collisions. Fig. 3.8 shows an example of plane collisions either handled on the simulation mesh or on an embedded mesh.

3.3.3 Corotation and Embedding

Corotation

Linear elasticity is convenient tool for analyzing the deformation behavior of material due to the simple algebraic formulation: energies are simple quadratic forms and the static and dynamical simulations can efficiently be computed by solving single linear systems. While the restriction of being only valid for small deformations is not too limiting for many applications in mechanics, computer graphics applications are often more demanding in this respect since visually interesting simulations mostly involve large deformations. Particularly when accompanied with large rotational components, the linear theory fails to give realistic material descriptions and lead to well-known artifacts [Müller and Gross, 2004]. The main reason for this is that the linear strain measure Eq. (3.2) is not invariant under rotations. In order to still make use of the linear theory, various papers present *corotational* extensions [Veubeke, 1976; Müller et al., 2002; Mezger et al., 2008; Chao et al., 2010], which introduce a correction to these artifacts at increased computation costs.

Idea. The idea of corotation is relatively simple: Since the linear Cauchy strain measure is not invariant under rotations, a deformation consisting of pure rotation results in a non-zero strain leading to unwanted fictive forces. Therefore, before measuring the strain at a given point, the rotational

component $\mathbf{R}$ of the local deformation field $\mathbf{u}$ is factored out and restored after the force computations.

Element-based Corotation. As noted by Müller et al. [2004] and Hauth and Strasser [2004], the original corotational approach [Müller et al., 2002] introduced ghost forces by performing the factorization and force correction directly per vertex. The principal cause of these erroneous forces lies in the fact that their computation does not preserve *symmetry*. For conventional linear FEM, the force contributions of a single element are symmetric — they have the property that

$$\sum_i \mathbf{f}_i^e = \mathbf{1}^T \mathbf{f}^e = \mathbf{1}^T \mathbf{K}^e \mathbf{u}^e = 0,$$

and do therefore not perform *work*. If the same rotation is now applied per element for each vertex force, this symmetry is not broken and no ghost forces do appear. On the other hand, if elemental forces are first assembled at the vertices and then rotated, symmetry is lost and ghost forces appear.

Therefore, instead of computing the elemental internal forces linearly as $\mathbf{f}_{int}^e = \mathbf{K}^e \mathbf{u}^e$, in corotated FEM they are computed as

$$\mathbf{f}_{int}^e = \mathbf{R}^T \mathbf{K}^e (\mathbf{R}(\bar{\mathbf{x}}^e + \mathbf{u}^e) - \bar{\mathbf{x}}^e), \tag{3.36}$$

i.e., the deformed position $\bar{\mathbf{x}}^e + \mathbf{u}^e$ is first rotated by $\mathbf{R}$ and then the new displacement vector is used to compute the forces that are finally rotated back. In all our element-based corotational implementations we use this approach and compute the rotation by polar decomposing of the deformation gradient $\mathbf{F} = \mathbf{QR}$ [Hauth and Strasser, 2004].

Quadrature-based Corotation. In case where no analytical integration is possible, there is also the possibility to apply the corotational approach when applying a quadrature scheme. However, again symmetry preservation has to be taken into account in order to not introduce ghost forces. This is possible if corotation is applied on the quadrature points themselves [Mezger et al., 2008; Martin et al., 2010].

For each quadrature point q, the rotational part $\mathbf{R}^q$ of the local deformation gradient $\mathbf{I} + \nabla \mathbf{u}(\bar{\mathbf{x}}^q)$ is again extracted using polar decomposition. Then the internal force contribution per quadrature point can be computed in the same manner as in element-based corotation:

$$\mathbf{f}_{int}^q = \mathbf{R}^{qT} \mathbf{K}^q (\mathbf{R}^q (\bar{\mathbf{x}}^q + \mathbf{u}^q) - \bar{\mathbf{x}}^q),$$

Again, we have symmetry of forces since $\mathbf{1}^T \mathbf{K}^q \mathbf{u}^q = 0$ holds and we are rotating all the force contributions with the same rotation matrix.

Embedding

A technique that has a long tradition in graphics is the use of embedded surface meshes for representing high-resolution geometry without actually requiring the core algorithms to work at these resolutions [Faloutsos et al., 1997; Capell et al., 2002; Müller and Gross, 2004; James et al., 2004; Sifakis et al., 2007b; Kim et al., 2008; Wojtan et al., 2009; Thürey et al., 2010]. The idea is to deform the embedded geometry by a continuous deformation field that is spanned by another much coarser discrete structure (e.g. a FEM mesh or meshless deformation field). See Fig. 3.8 for an example of a sphere embedded into a coarse FEM mesh.

In our implementations we also make heavy use of this technique: Assume that the surface geometry is given by undeformed vertices $\bar{\mathbf{x}}_j^s$ which is embedded in a coarse discrete structure that generates a deformation field $\mathbf{u}(\bar{\mathbf{x}}) = \sum_i \mathbf{u}_i N_i(\bar{\mathbf{x}})$. For each surface point we can precompute the weights $N_i(\bar{\mathbf{x}}_j^s)$ and at runtime compute the current position as

$$\mathbf{x}_j^s = \bar{\mathbf{x}}_j^s + \sum_i \mathbf{u}_i N_i(\bar{\mathbf{x}}_j^s).$$

This computation is very cheap since only a linear combination of a small number of supporting DOFs needs to be computed.

3.3.4 Space Discretization for Nonlinear and Resultant-Based Formulations

While we shed light on different important concepts of the discretization procedures for linear elasticity problem, we will now shortly turn to the discrete formulations of the nonlinear and resultant-based models. For these, the central Galerkin discretization procedure is the same as for linear problems — differences arise in the type of discrete equations (nonlinear algebraic) and the requirements for representing of the discretization subspace. Obviously, in each of these fields again a vast number of approaches exist such that we will again only focus on the relevant concepts for this thesis.

Nonlinear FEM

Applying the FEM for non-linear elasticity problems follows the same lines as linear elasticity. Let us discuss the simplest approach building up on piecewise linear elements that form a proper subspace of the $H^1(\Omega)$, which is again the solution space for nonlinear solid problems.

In the case of linear elements, the deformation gradient $\mathbf{F}$ becomes piecewise constant inside the elements and can be computed efficiently as

$$\mathbf{F} = \mathbf{x}\bar{\mathbf{x}}^{-1},$$

where $\mathbf{x} = [\mathbf{x}_2 - \mathbf{x}_1, \mathbf{x}_3 - \mathbf{x}_1, \mathbf{x}_4 - \mathbf{x}_1]$ and $\bar{\mathbf{x}} = [\bar{\mathbf{x}}_2 - \bar{\mathbf{x}}_1, \bar{\mathbf{x}}_3 - \bar{\mathbf{x}}_1, \bar{\mathbf{x}}_4 - \bar{\mathbf{x}}_1]$ [Irving et al., 2004], where the $\mathbf{x}_i$ and $\bar{\mathbf{x}}_i$ describe an element's deformed and undeformed vertex positions, respectively. As a consequence, the Green strain $\mathbf{E}$ becomes also piecewise constant (these elements are also called *constant strain elements* - CSE) as well as the energy density Ψ. It follows that the total energy can then be integrated analytically simply as

$$W = \sum_e \Psi_e V_e,$$

where V_e describes an element's volume. Since the energy is now fourth-order in the DOF, the resulting force are cubic, requiring nonlinear solvers for static and dynamic problems (Section 3.3.5).

Resultant-Based Formulations

Discretizing resultant-based formulations is more involved than solid elasticity problems due to the introduction of bending energies. In order to compute these, second derivatives of the solution field need to be computed, leading to the requirement of C^1-continuous basis functions spanning a subspace of the Sobolev space $H^2(\Omega)$ [Hughes, 2000]. The construction of such basis function is in general more involved and requires specialized approaches (see [Cirak et al., 2000; Chapelle and Bathe, 2003] for shells, [Spillmann and Teschner, 2007] for rods).

A popular alternative are *discrete formulations*, where energies are formulated directly using discrete measures on the underlying representation structures, building up discrete counterparts of the continuous mechanical principles (see [Grinspun et al., 2003; Garg et al., 2007; Wardetzky et al., 2007] for discrete shell models, [Bergou et al., 2008; 2010] for discrete rods). In these formulations, a discretization of continuous equations is not necessary anymore.

3.3.5 Time Discretization

So far we discussed on how to discretize the PDE in space and how to describe the spatial behavior of the material in a discrete manner. For statics, this leads to a system of linear or nonlinear equations, while for dynamics a system of ODEs has to be solved. Next, we will briefly discuss the numerical solution procedures that were followed in this thesis.

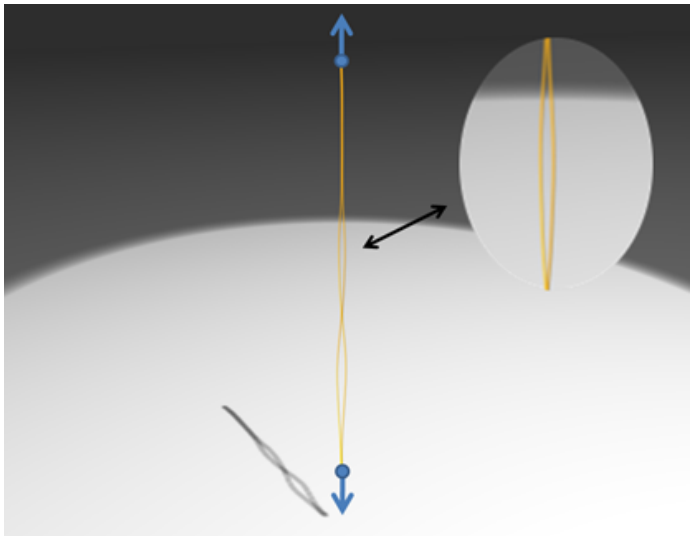


Figure 3.9: *Opposing forces are applied to both ends of a straight rod. The new rest state should be computed with an iterative corotated static solver which however fails to converge and oscillates due to stability issues of the fixed point iteration.*

Statics

Linear Elasticity. In the case of linear elasticity the resulting linear system in Eq. (3.31) can be solved efficiently using high-performance linear solvers. Since the system matrix $\mathbf{K}$ is symmetric positive definite (spd), we commonly use sparse direct solver based on incomplete Cholesky factorization [Toledo et al., 2003; Chen et al., 2008].

Corotated Linear Elasticity. In case of corotated linear elasticity, we need to use an iterative solving procedure for the (now nonlinear) equation

$$\mathbf{f}_{int}(\mathbf{u}) = \mathbf{f}_{ext},$$

where $\mathbf{f}_{int}(\mathbf{u})$ is given by Eq. (3.36). To solve this problem efficiently, a common approach lies in the following update procedure:

$$\mathbf{u} \Leftarrow \mathbf{K}(\mathbf{u})^{-1}(\mathbf{f}_{ext} - \mathbf{K}(\mathbf{u})\bar{\mathbf{x}} + \mathbf{R}^T(\mathbf{u})\mathbf{K}\bar{\mathbf{x}}), \tag{3.37}$$

where the last known solution $\mathbf{u}$ is used in the right hand side and we used the short notation $\mathbf{K}(u) = \mathbf{R}^T(\mathbf{u})\mathbf{K}\mathbf{R}(\mathbf{u})$. This update procedure can easily be recognized as a *fix point iteration*.

A commonly known result from numerical optimization [Kreyszig, 2005] states that a fixed point iteration $x \Leftarrow F(x)$ can only converge, if $|F'(x)| < 1$

holds. However, if we consider Eq. (3.37) under this light, it is not clear at all that convergence can be expected. Moreover, convergence should become even more difficult if the external force $\mathbf{f}_{ext}$ increases. This expected stability problems from the fixed point iteration theory point of view is covered by the observed behavior of such static simulations, where the corotational approach becomes instable for large deformations (due to large external forces), as depicted in Fig. 3.9[1]. Increasing the stability of corotational approaches is still an active research topic where recent publications [McAdams et al., 2011; Chao et al., 2010] show up interesting principled ways to improve these methods.

Nonlinear Elasticity. In the case of nonlinear elasticity the discrete systems are fully nonlinear and therefore standard iterative methods from nonlinear numerical optimization are used [Nocedal and Wright, 2006]. In our framework we used a Newton-Raphson method with linesearch and Hessian correction (for indefinite Hessians) — a standard procedure in numerical optimization.

Assume we are given (in index notation) a general nonlinear potential energy function $W(x_k)$ with gradient $g_i = \frac{\partial W(x_i)}{\partial x_i}$ and Hessian $H_{ij} = \frac{\partial^2 W(x_k)}{\partial x_i \partial x_j}$ which we sought to minimize. A *Newton-Raphson* correction step $\Delta\mathbf{x}^n$ for a intermediate solution $\mathbf{x}^n$ is computed by solving the linear system

$$\mathbf{H}(\mathbf{x}^n)\Delta\mathbf{x}^n = \mathbf{g}(\mathbf{x}^n) \tag{3.38}$$

and the solution is updated as

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \alpha\Delta\mathbf{x}^n.$$

The stepsize α is determined with a linesearch procedure in order to guarantee sufficient decrease of the energy for the step [Nocedal and Wright, 2006]. Depending on the local geometry of the energy function, the Hessian $\mathbf{H}$ might not be positive-definite in which case a descent direction is not guaranteed to be found by Eq. (3.38). This case can efficiently be detected by applying a Cholesky-based solver that fails to solve in these cases. Then, we perform a simple Hessian correction step

$$\mathbf{H} \Leftarrow \mathbf{H} + \beta\mathbf{I},$$

where $\beta > 0$ is a small value, but sufficiently large to make $\mathbf{H}$ positive definite. This loop is iterated until the iteration converges. In our implementation we used the simple test

$$|\mathbf{g}(\mathbf{x}^n)| < \epsilon,$$

[1] This figure is part of Jeronimo Bayer's master thesis [2011].

for some threshold value ϵ to determine if the iteration converged.

Dynamics

There exists a large variety of time integrators for numerically solving equations of motion of the type Eq. (3.8) and since the focus of this thesis did not lie on this topic we do not give a thorough introduction in this large field. Therefore, we will just quickly recapitulate the standard explicit, symplectic and (semi-)implicit Euler integrators that were used in this thesis. These schemes are basically the same for linear and nonlinear elasticity problems since only the actual force and Jacobian computation differ from each other.

Explicit Euler. In order to formulate the simplest time integration scheme we first introduce the velocity vector $\mathbf{v}$ as a state variable and transform the second order Eq. (3.8) into the system of first order equations

$$\begin{aligned} \dot{\mathbf{u}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= \mathbf{M}^{-1}(\mathbf{f}_{ext} - \mathbf{f}_{int}(\mathbf{u})), \end{aligned}$$

which we can write alternatively as

$$\dot{\mathbf{y}}(t) = \mathbf{F}(\mathbf{y}, t)$$

with

$$\mathbf{y}(t) = \begin{pmatrix} \mathbf{u}(t) \\ \mathbf{v}(t) \end{pmatrix} \quad \text{and} \quad \mathbf{F}(\mathbf{y}, t) = \begin{pmatrix} \mathbf{v}(t) \\ \mathbf{M}^{-1}(\mathbf{f}_{ext}(t) - \mathbf{f}_{int}(\mathbf{u})) \end{pmatrix}.$$

By developing $\mathbf{y}(t)$ in a first order Taylor approximation $\mathbf{y}(t+h) \approx \mathbf{y}(t) + h\dot{\mathbf{y}}(t)$ and using the relationship stated above, we get the update rule

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\, \mathbf{F}(\mathbf{y}_n, t).$$

Or, by using again position and velocity state variables, we can formulate the *explicit Euler* integration scheme as

$$\begin{aligned} \mathbf{u}_{n+1} &\Leftarrow \mathbf{u}_n + h\, \mathbf{v}_n \\ \mathbf{v}_{n+1} &\Leftarrow \mathbf{v}_n + h\, \mathbf{M}^{-1}(\mathbf{f}_{ext} - \mathbf{f}_{int}(\mathbf{u}_n)). \end{aligned}$$

While this is the simplest scheme possible, is has serious stability issues for larger timesteps h. Without incorporating additional damping, this explicit integration increases the total system energy continuously, making it not suitable for dynamical simulations.

Symplectic Euler. By modifying the explicit Euler scheme only marginally, we arrive at the *symplectic Euler* scheme: If we first evaluate velocity with the current acceleration and then use the updated velocity for updating positions we arrive at a first order method that is able to preserve the total energy and results in better stability:

$$\begin{aligned} \mathbf{v}_{n+1} &\Leftarrow \mathbf{v}_n + h\,\mathbf{M}^{-1}(\mathbf{f}_{ext} - \mathbf{f}_{int}(\mathbf{u}_n)) \\ \mathbf{u}_{n+1} &\Leftarrow \mathbf{u}_n + h\,\mathbf{v}_{n+1}. \end{aligned}$$

(Semi-)Implicit Euler. If one is willing to trade improved stability against additional computational costs, *implicit Euler* integration is the favorable scheme due to its unconditional stability property.

We arrive at the implicit Euler scheme if we perform the first order Taylor expansion around time $t + h$ an approximate the solution *backwards* at time t, i.e., we assume

$$\mathbf{y}(t) \approx \mathbf{y}(t+h) - h\dot{\mathbf{y}}(t+h).$$

By rearranging terms and using the ODE we get

$$\mathbf{y}_{n+1} \Leftarrow \mathbf{y}_n + h\mathbf{F}(\mathbf{y}_{n+1}, t+h),$$

i.e., we get the update rule in an implicit form where the unknown state $\mathbf{y}_{n+1}$ appears on both sides such that a (non-)linear system needs to be solved. To concretize this, we formulate the implicit scheme for our problem and with respect to the unknown positions $\mathbf{u}_{n+1}$ as

$$\mathbf{M}\left(\frac{\mathbf{u}_{n+1} - \mathbf{u}_n}{h^2} - \frac{\mathbf{v}_n}{h}\right) + \mathbf{f}_{int}(\mathbf{u}_{n+1}) = \mathbf{f}_{ext}. \tag{3.39}$$

Following [Baraff and Witkin, 1998], this nonlinear equation in $\mathbf{u}_{n+1}$ is solved using a conventional Newton-Raphson procedure. If only one step of this iterative solve is performed per timestep, the resulting *semi-implicit* method is still stable. While being less accurate, it is more efficient since assembly and solve have to be performed only once. After each iteration, the velocity is updated as $\mathbf{v}_{n+1} = \frac{\mathbf{u}_{n+1} - \mathbf{u}_n}{h}$.

3.4 Discussion and Outlook

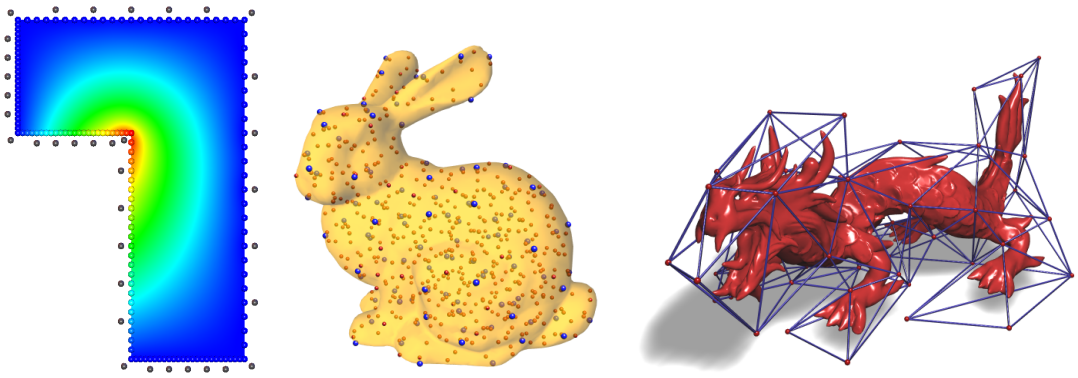
In this chapter we recapitulated the essential concepts required for simulating deformable objects. We divided them into three main blocks: *Continuum mechanics for solids*, where we introduced the mathematical foundation for

simulating elastic solids, *resultant-based formulations*, where we introduced alternative models for simulating objects featuring thin geometries, and *discrete solution representation*, where we presented the Galerkin method for the numerical treatment of the continuous problems.

In the following chapter we will now in turn present how to extend these basic simulation principles. First, we will have a closer look at the discretization subspaces and present new forms for spanning them in order to achieve more flexible FEM discretizations (Chapter 4). Then, we will revisit the specialized continuous models for rods and shells and reconcile them into a single model for all three types of geometries (Chapter 5). Last, we will get back to the elastic potentials and present an example-based approach to "upgrade" them for additional directability (Chapter 6).

CHAPTER 4

Tailoring Solution Subspaces



In this chapter, we will study how to make advantage of choosing specific solution subspaces in order to simplify and enhance the conventional FEM and to make it more amenable for graphics applications. We will focus on solid geometries and will pursue two different goals: First, we will present two methods to soften the strict meshing requirements of previous tetra- and hexahedral FEM approaches by introducing more flexible discretization structures. Second, we will present a method to increase preservation of geometric features at sub-element scales.

4.1 Overview

Polyhedral Elements. Finite element simulations in computer graphics are typically based on tetrahedral or hexahedral elements which enable simple and efficient implementations, but in turn require complicated remeshing in case of topological changes or adaptive refinement. However, by extending the FEM to also handle more flexibly arbitrary polyhedral elements and therefore adapting the solution subspace accordingly, remeshing can be avoided completely. By making use of harmonic coordinates, we are able to design an according solution subspace that allows the use of generalized element shapes. These satisfy all necessary conditions for FEM simulations and seamlessly generalize both linear tetrahedral and trilinear hexahedral elements.

Meshfree Representation. In order to span a suitable solution subspace, the FEM makes use of an underlying mesh data structure for defining local shape functions. For simulations requiring frequently and dynamically changing discretizations, e.g., adaptive simulations or simulations involving topological changes (like fracturing or cutting), the generation of high quality meshes is a demanding task. As an alternative to polyhedral element shapes, we can also make use of moving least squares (MLS) interpolation to span solution spaces that require only point information without explicit connectivity, simplifying the discretization task considerably. While the presented approach is a variant of the *Element-Free Galerkin* method commonly known in mechanics literature [Belytschko et al., 1994], we will present a suitable extension in Chapter 5, capable of also handling different resultant-based formulations.

Feature Preservation. When embedding high-resolution surface geometry into coarser finite element meshes, there is a priori no guarantee that the shape of fine features in the embedded geometry is preserved during deformation — due to simple interpolation schemes they are usually unnaturally distorted. By choosing a suitable discretization subspace, we can however achieve feature preservation also at sub-element scales: By making use of quasi-conformal Green Coordinates (GC) for the representation of the deformation field, we are able to get sub-element shape-preservation 'by construction' in a cage-based setting.

Since the designs of the discretization subspaces presented in this chapter are independent of the (non-)linearity of the underlying continuous model, we will demonstrate the versatility of the approaches within a simulation

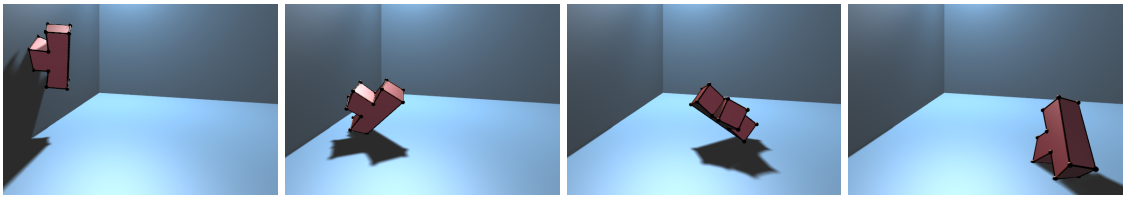


Figure 4.1: *Using harmonic basis functions, even non-convex polyhedral elements can be used directly in FEM simulations.*

framework for corotated linear elasticity. Note however, that the same spaces can equivalently be used for nonlinear elasticity.

4.2 Polyhedral Elements

Traditionally, FEM simulations in computer graphics rely on strictly tetrahedral or hexahedral meshes, which simplify the finite element approximation and significantly speeds up the involved computations. However, allowing a single element shape only can be too restrictive, since it requires complex remeshing in case of topological changes, for instance due to cutting, fracture, or adaptive refinement.

One class of approaches [Molino et al., 2004; Sifakis et al., 2007a; 2007b] therefore avoids remeshing after cutting by embedding each resulting cut part into an individual copy of the original tetrahedron. This is conceptually similar to the XFEM method [Jeřábková and Kuhlen, 2009; Kaufmann et al., 2009b] where discontinuities are effectively introduced into the basis functions such that the resulting subspace is able to represent the disconnection of the material at these locations. An interesting alternative is the approach of Wicke et al. [Wicke et al., 2007]: They directly support more general convex polyhedral elements in finite element simulations by employing mean-value coordinates as a generalization of linear barycentric FEM shape functions. That is, discontinuities are still represented using (disconnected) mesh geometry, but by allowing convex polyhedral elements, remeshing can be omitted due to more flexible subspace bases.

In this section we extend their approach to arbitrary convex and non-convex polyhedral elements using *harmonic coordinates* [Joshi et al., 2007] as FEM basis functions (see Fig. 4.1). Harmonic basis functions naturally generalize linear basis functions for tetrahedral elements and trilinear basis functions for hexahedral elements. Hence, this approach seamlessly integrates into existing FEM frameworks, such that standard tetrahedral or hexahedral elements can

be used in regular parts of the model, whereas irregular polyhedral elements are used in regions of cutting or adaptive refinement.

Harmonic coordinates, as the solution of a Laplace PDE with Dirichlet boundary constraints, have an analytic solution for simple element shapes only. For general polyhedra they therefore have to be approximated numerically, using for instance finite differences, finite elements, or the boundary element method. As a simple and flexible alternative we will present an approximation using radial basis functions, which guarantees the resulting basis functions to be harmonic and to furthermore exactly satisfy important physical conservation properties.

The use of general polyhedral elements significantly increases the flexibility in generating and manipulating the simulation mesh. We demonstrate this versatility in examples of cutting and adaptive refinement.

4.2.1 Harmonic Basis Functions

Shape Function. We propose to use *harmonic basis functions* as a generalization of linear barycentric basis functions to general polyhedral elements. A shape function $N_i^e : e \to \mathbb{R}$ is *harmonic* if its Laplacian vanishes in e, in which case it is uniquely determined by Dirichlet boundary constraints $b(\bar{\mathbf{x}})$ on ∂e:

$$\Delta N_i^e(\bar{\mathbf{x}}) = 0\,, \qquad \text{for} \quad \bar{\mathbf{x}} \in e\,, \tag{4.1}$$

$$N_i^e(\bar{\mathbf{x}}) = b_i(\bar{\mathbf{x}})\,, \qquad \text{for} \quad \bar{\mathbf{x}} \in \partial e\,. \tag{4.2}$$

For a straightforward finite element simulation we require nodal basis functions N_i^e to interpolate quantities within each element e. If these functions are chosen to be harmonic, they are fully determined by the values $b_i(\bar{\mathbf{x}})$ on the element boundary ∂e, which we set up following Joshi et al. [2007]: First, in order to interpolate nodal quantities, the basis function N_i^e of node i has to equal 1 at the node $\bar{\mathbf{x}}_i$ and 0 at all others, i.e.,

$$N_i^e\big(\bar{\mathbf{x}}_j\big) \;=\; \delta_{ij} \quad \forall\, i,j = 1,\ldots,k\,. \tag{4.3}$$

Additionally, in order to ensure continuity across element boundaries, the values of the basis function for node i defined in neighboring elements e_1 and e_2 should coincide on the face or edge shared by the two elements:

$$N_i^{e_1}(\bar{\mathbf{x}}) \;=\; N_i^{e_2}(\bar{\mathbf{x}}) \quad \text{for} \quad \bar{\mathbf{x}} \in e_1 \cap e_2\,. \tag{4.4}$$

Recursive Definition. Property (4.4) can be guaranteed by choosing the values on the faces of a d-dimensional element to be $(d-1)$-dimensional harmonic coordinates. For a trivariate harmonic basis function N_i^e on a 3D element e the boundary conditions are bivariate harmonic coordinates on its faces, which themselves are determined by univariate harmonic (i.e., linear) interpolation of the nodal values $N_i^e(\bar{\mathbf{x}}_j) = \delta_{ij}$ along the edges.

It follows from these recursively defined boundary constraints and the uniqueness of harmonic functions for fixed Dirichlet constraints, that harmonic shape functions reproduce linear triangles and bilinear quads in 2D, as well as linear tetrahedra and trilinear hexahedra in 3D. The harmonic basis for a more complex 2D element is shown in Fig. 4.2.

Basis Function Properties. Harmonic basis functions satisfy all requirements for admissible FEM basis functions [Joshi et al., 2007]:

- Since 3D harmonic shape functions degenerate to 2D harmonic coordinates on element faces, they are continuous across element boundaries: $N_i \in \mathrm{C}^0(\Omega)$.
- As solution to Laplace's equation (4.1) they are smooth within elements: $N_i^e \in \mathrm{C}^\infty(e)$.
- For fixed constraints b_i, (4.1) characterizes the minimizer of the Dirichlet energy $\int_e \left\|\nabla N_i^e\right\|^2$. Hence, the gradients of harmonic functions are square integrable: $\nabla N_i^e \in \mathrm{L}^2(e)$. Combining the last three points we get $N_i \in \mathrm{H}^1(\Omega)$.
- Constant reproduction is given by the partition of unity property $\sum_i N_i(\bar{\mathbf{x}}) = 1$, which follows from the (recursively) defined boundary condition: Since $\sum_i b_i(\bar{\mathbf{x}}) = 1$ holds, $\sum_i N_i(\bar{\mathbf{x}}) = 1$ follows immediately from the maximum principle.
- Linear reproduction is given by the fact that linear functions are also harmonic. When a Laplace problem $\Delta u = 0$ takes linear boundary conditions $u(\bar{\mathbf{x}}) = l(\bar{\mathbf{x}}), \bar{\mathbf{x}} \in \partial\Omega$ (with $l(\bar{\mathbf{x}})$ being a linear function), the *unique* solution u to the problem is also linear. Therefore, since the linear combination $\sum_i u_i N_i$ is harmonic by construction, it is sufficient to check that the boundary function $\sum_i u_i N_i(\bar{\mathbf{x}})|_{\bar{\mathbf{x}}\in\partial\Omega}$ is also linear. By the recursive definition of harmonic coordinates we know, that a harmonic problem has to be solved on each face individually. However, if we consider the simple case of a one-dimensional boundary (recursion root), we can directly conclude that the linear hat function will generate the linear function interpolating the nodal values. In this

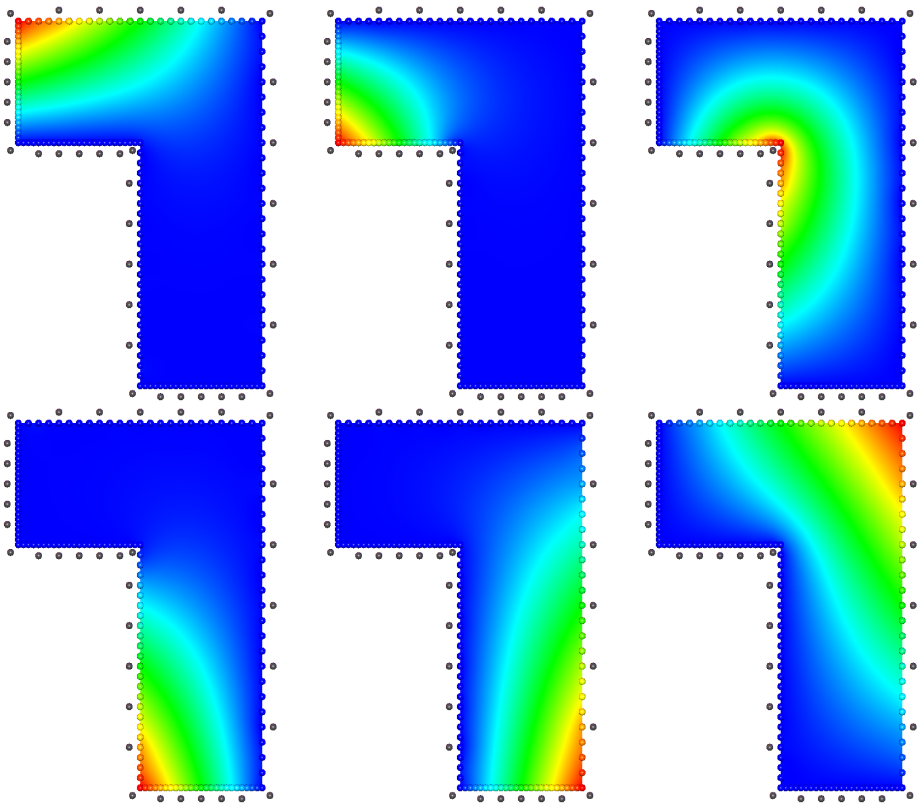


Figure 4.2: *Harmonic basis functions for the six nodes of a non-convex 2D element. The constraint collocation points* $\mathbf{c}_i$ *are visualized as small spheres along the element boundary, the kernel centers* $\mathbf{k}_i$ *are shifted slightly outside and are shown in gray.*

manner, linearity can then be propagated up from edges to faces up to volumes.

4.2.2 Numerical Approximation

Closed form expressions for harmonic basis functions exist for simple element shapes only, such as tetrahedra or hexahedra. For more general elements, harmonic basis functions N_i^e have to be computed numerically as the solution of (4.1), (4.2), which is valid for both 2D faces and 3D elements. To this end, several well-established techniques for solving the 2D and 3D Laplacian PDEs exist, each having their own respective advantages and drawbacks.

Approximation Schemes

Finite Differences. Overlaying the element by a regular 3D grid and using a finite difference discretization leads to the solution of a sparse linear system for a piecewise trilinear approximation of N_i^e [Joshi et al., 2007]. While this method is comparatively easy to implement, an accurate solution requires a sufficiently dense grid in order to resolve the smallest edges/faces of the element. In particular for cutting, where small edges occur frequently, the cubic growth of volumetric grids leads to very complex systems, which require advanced multi-grid methods for their solution [Joshi et al., 2007].

Finite Elements. They could be used to solve (4.1) on an adaptive tessellation of polyhedral elements, thereby overcoming the limitations of regular grids. However, since the major goal of our approach is to enable adaptive FEM computations *without* complex remeshing of elements, the recursive application of adaptive FEM to each polyhedral element is a chicken-and-egg problem and contradicts our goals.

Boundary Element Method. The boundary element method (BEM) is also well suited to solve the PDE (4.1). By formulating the solution as an integral of fundamental solutions over the element's boundary, it avoids a volumetric tessellation and therefore needs a boundary discretization only. Due to our experiments the major drawback of BEM is performance: In its exact formulation, each function evaluation requires a full integral over the element's boundary, which makes the numerical integration of (3.34) very expensive.

Fundamental Solutions. While all the above methods can be employed for solving (4.1), (4.2) we found the method of fundamental solutions (MFS) [Fairweather and Karageorghis, 1998] to be a more flexible, easier-to-implement, yet sufficiently accurate alternative.

Method of Fundamental Solutions

MFS is closely related to BEM: It is also a boundary method, and thus does not require a volumetric tessellation, and it also represents the approximate solution in terms of fundamental solution kernels. However, instead of the boundary integrals in BEM, MFS employs a simple, meshless collocation.

Representation. A shape function $N_i^e(\bar{\mathbf{x}})$, simply denoted by $N(\bar{\mathbf{x}})$ in the following, is represented in the following form:

$$N(\bar{\mathbf{x}}) \;=\; \sum_{j=1}^{n} w_j \cdot \psi\big(\|\bar{\mathbf{x}} - \mathbf{k}_j\|\big) \;+\; \mathbf{a}_1^T \bar{\mathbf{x}} + a_0\,, \tag{4.5}$$

where the first part is a superposition of n weighted radial basis functions ψ (RBFs), centered at $\mathbf{k}_j$, and the second part is a linear polynomial in $\bar{\mathbf{x}}$. The kernel function ψ is chosen as fundamental solution of the Laplace PDE, which is $\psi(r) = \log r$ in 2D and $\psi(r) = 1/r$ in 3D. As a consequence, the function (4.5) is harmonic by construction [Duchon, 1977], in the whole domain except at the kernels' singularities $\mathbf{k}_j$.

Kernel Placement. Hence, the kernel $\mathbf{k}_j$ have to be placed outside the element. A standard method is to first sample the boundary by $\mathbf{s}_j \in \partial e$, and to move the kernels outward in (interpolated) normal direction by a small fraction ξ of the element size:

$$\mathbf{k}_j \;=\; \mathbf{s}_j \;+\; \xi \cdot \mathrm{size}(e) \cdot \mathbf{n}\big(\mathbf{s}_j\big)\,. \tag{4.6}$$

For non-convex elements one additionally has to take care that this simple offsetting does not generate centers in the element's interior. For generating the n samples $\mathbf{s}_j$, we select the element's nodes, about 3–5 samples on each edge, and a uniform sampling of its faces of about the same density.

Approximating Dirichlet Conditions. The function (4.5) satisfies (4.1) by construction, thus we solve for the best approximation of the Dirichlet constraints (4.2). To this end, we approximate the boundary integral of the L^2 error by a sum of m collocation points $\mathbf{c}_i$:

$$\int_{\partial e} |N(\bar{\mathbf{x}}) - b(\bar{\mathbf{x}})|^2 \approx \frac{1}{m} \sum_{i=1}^{m} |N(\mathbf{c}_i) - b(\mathbf{c}_i)|^2 \to \min\,. \tag{4.7}$$

These collocation points $\mathbf{c}_i \in \partial e$ are generated equivalently to the samples $\mathbf{s}_j$, but at a higher resolution of $m \approx 3n$. The distribution of kernel centers $\mathbf{k}_i$ and collocation constraint points $\mathbf{c}_i$, as well as the resulting basis functions are shown for a non-convex L-shaped 2D element in Fig. 4.2.

Least Squares Problem. Given the kernel centers $\mathbf{k}_i$, the minimization of the L^2 error (4.7) amounts to solving an overdetermined linear least squares

system for the coefficients of (4.5):

$$\begin{pmatrix} \psi_{11} & \dots & \psi_{1n} & \mathbf{c}_1^T & 1 \\ \vdots & & \vdots & \vdots & \vdots \\ \psi_{m1} & \dots & \psi_{mn} & \mathbf{c}_m^T & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ \vdots \\ w_n \\ \mathbf{a}_1 \\ a_0 \end{pmatrix} = \begin{pmatrix} b(\mathbf{c}_1) \\ \vdots \\ b(\mathbf{c}_m) \end{pmatrix}, \tag{4.8}$$

where $\psi_{ij} = \psi(\|\mathbf{c}_i - \mathbf{k}_j\|)$. This least squares system can be solved in a numerically robust manner using the QR factorization or the SVD pseudo-inverse [Golub and Loan, 1989].

In order to compute a 3D basis function N_i^e for an element e, we first solve the above linear system on each of its faces. The resulting 2D harmonic functions constitute the boundary constraints for the final 3D linear system, which yields the coefficients of N_i^e. Notice that in order to compute all k basis functions $N_1^e, \dots, N_k^e$ of an element e with k nodes, the same 2D and 3D systems are solved for k different right-hand sides $b_i(\bar{\mathbf{x}})$. After factoring each matrix once, these systems can be solved efficiently by back-substitution.

Discussion

FEM Requirements. As described in Section 4.2.1, *exact* solutions of (4.1), (4.2) satisfy the conditions for admissible FEM shape functions. The numerical approximation $N(\mathbf{x})$ from (4.5) satisfies all but one exactly, and one up to small numerical errors.

Since the singularities $\mathbf{k}_j$ are located outside of e, we have $N \in \mathrm{C}^1(e)$ and $\nabla N \in \mathrm{L}^2(e)$. However, the C^0 continuity across elements is only satisfied approximately through the Dirichlet conditions (4.2), resp. (4.7). Our typical choice of 5 edge samples for generating kernels through (4.6) leads to L^2 boundary errors of about 2–3%. More accurate results can be achieved by using more kernels in (4.5). However, as we show in Section 4.2.4, the accuracy of the global solution is not limited by the individual basis functions' errors.

Linear Reproduction. Typically, MFS approximations are based on fundamental solution kernels $\psi(\|\cdot - \mathbf{k}_j\|)$ only, and do not include a linear polynomial as in (4.5). This polynomial, however, is crucial in our case, since it guarantees *exact* reproduction of linear functions, independent of the number n of kernels used. Due to our experiments even small errors in the linear

reproduction (L^2 error around 10^{-3}) would cause ghost forces, thereby destroying the preservation of linear and angular momenta and resulting in counter-intuitive behavior.

Let us give a short proof of this important property. We use the following notation: First we can write the i-th basis functions as $N_i(\bar{\mathbf{x}}) = \mathbf{p}(\bar{\mathbf{x}}) \cdot \mathbf{w}_i$ with $\mathbf{p}(\bar{\mathbf{x}}) = [\psi_1(\bar{\mathbf{x}}),\dots,\psi_n(\bar{\mathbf{x}}),\bar{\mathbf{x}},1]^T$ and $\mathbf{w}_i = [w_1,\dots,w_n,\mathbf{a}_1^T,a_0]^T$. If we let $\mathbf{b}_i$ denote the vector of boundary condition values for this basis function (right hand side of Eq. (4.8)), and the matrix $\mathbf{A}$ hold all evaluations of the basis functions at the collocation points (system matrix in Eq. (4.8)), we can write the least squares solution as $\mathbf{w}_i = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}_i$. Therefore, we can also write the basis function i in the following form

$$N_i(\bar{\mathbf{x}}) = \mathbf{p}(\bar{\mathbf{x}}) \cdot (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}_i.$$

For proofing the PU property for this approximate representation of the true harmonic basis function we can just plug this definition into the sum of basis functions. By further making use of linearity and the property that the boundary values need to sum up to 1, i.e., $\sum_i \mathbf{b}_i = \mathbf{1}$, we get

$$\begin{aligned}
\sum_i N_i(\bar{\mathbf{x}}) &= \sum_i \mathbf{p}(\bar{\mathbf{x}}) \cdot (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}_i \\
&= \mathbf{p}(\bar{\mathbf{x}}) \cdot (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\sum_i \mathbf{b}_i \\
&= \mathbf{p}(\bar{\mathbf{x}}) \cdot (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{1} \\
&= 1.
\end{aligned}$$

In summary, this derivation states that it is equivalent to first approximate individual basis functions and to sum them up, or to reconstruct directly the constant 1 function with the chosen *ansatz* (which it is of course perfectly capable to do).

The proof for linear reproduction works similarly. In this case, we have to show that the basis can represent any linear function $l(\bar{\mathbf{x}})$, i.e., that $\sum_i l(\bar{\mathbf{x}}_i)N_i(\bar{\mathbf{x}}) = l(\bar{\mathbf{x}})$. In the same way as for the PU, we can formulate

$$\begin{aligned}
\sum_i l(\bar{\mathbf{x}}_i)N_i(\bar{\mathbf{x}}) &= \sum_i l(\bar{\mathbf{x}}_i)\mathbf{p}(\bar{\mathbf{x}}) \cdot (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}_i \\
&= \mathbf{p}(\bar{\mathbf{x}}) \cdot (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\sum_i l(\bar{\mathbf{x}}_i)\mathbf{b}_i, \\
&= \mathbf{p}(\bar{\mathbf{x}}) \cdot (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T[l(\mathbf{c}_1),\dots,l(\mathbf{c}_m)]^T, \\
&= l(\bar{\mathbf{x}}),
\end{aligned}$$

i.e., an arbitrary linear function $l(\bar{\mathbf{x}})$ can indeed be computed exactly by the approximate representation. Note that from line two to three we used the fact

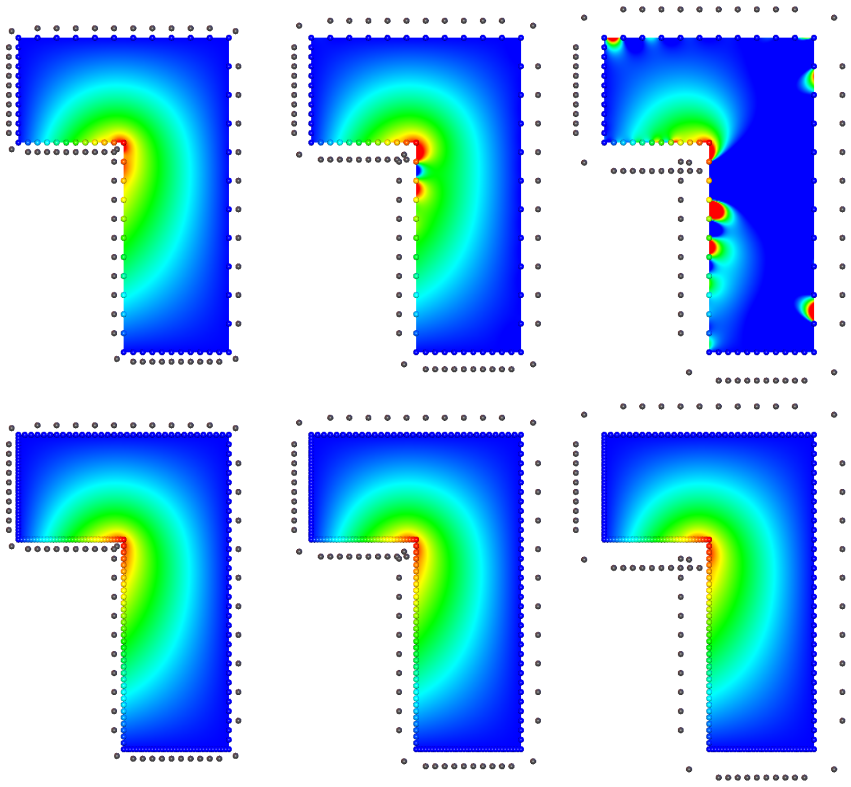


Figure 4.3: *Top: If we enforce the boundary conditions exactly, the solution oscillates and is very sensitive to the offset distance ($\xi = 0.05, 0.1, 0.15$). Bottom: In contrast, dense least-squares boundary conditions are more robust and considerably less affected by different parameter choices.*

that the boundary values $\mathbf{b}_i$ are samples of the harmonic boundary conditions of each basis function. The linear combination $\sum_i l(\bar{\mathbf{x}}_i)\mathbf{b}_i$ therefore holds the samples of the linear function $l(\bar{\mathbf{x}})$ at exactly these sample locations. It follows then that the third line states the reconstruction problem of a linear function using our chosen *ansatz* with constant and linear monomials which is of course exactly solvable.

Kernel Placement. The offset distance ξ in (4.6) has to be chosen heuristically. In our experiments, we found $\xi = 0.1$ to be a reliable setting, as similarly stated in [Li et al., 2007]. Moreover, due to our dense sampling of collocation points $\mathbf{c}_i$ ($m \approx 3n$) the solution of the least squares system (4.8) is hardly influenced by ξ. In contrast, an exact interpolation ($m = n + 3$) cannot prevent oscillations on the boundary between the $\mathbf{c}_i$ and would be much more sensitive to the offset distance (Fig. 4.3).

Degenerate Elements and Edges. Degenerate elements cause numerical problems for harmonic shape functions, just as they do for standard shape functions. Two (almost) *coincident kernels* $\mathbf{k}_i$, $\mathbf{k}_j$ lead to linearly dependent rows i, j in (4.8), yielding a rank deficient matrix. Simple cases, like the clustered kernels near the concave corner in Fig. 4.3, can be resolved explicitly by merging kernels, or implicitly through the SVD pseudo-inverse.

For *degenerate edges* with coincident nodes $\bar{\mathbf{x}}_i$, $\bar{\mathbf{x}}_j$, we conceptually merge $\bar{\mathbf{x}}_i$ and $\bar{\mathbf{x}}_j$ by computing one joint basis function ϕ^e_{ij} from the constraints $b_i + b_j$ in (4.2) and using a joint nodal displacement $\mathbf{u}_{ij} = \mathbf{u}_i = \mathbf{u}_j$. Note that this does not change the simulation mesh, and therefore preserves, e.g., planarity of faces. Almost planar *sliver elements* can be merged with their neighbors as proposed in [Wicke et al., 2007].

4.2.3 Simulation

After introducing harmonic basis functions and their numerical approximation, we insert them into the Galerkin discretization of Section 3.1.1 and set up the matrix equations for the FEM simulation. At this point, let us quickly recapitulate the required steps to arrive at the resulting equations and give a more specific formulation of the polyhedral element based approach using the practical *Voigt notation*.

Discrete Displacement. Once the basis functions N^e_i are computed as described in the previous section, the displacement $\mathbf{u}^e$ within e can be approximated as in (3.34), which can also be written in matrix notation

$$\mathbf{u}^e(\bar{\mathbf{x}}) \;:=\; \sum_{i=1}^{k} N^e_i(\bar{\mathbf{x}})\,\mathbf{u}_i \;=\; \mathbf{H}_e(\bar{\mathbf{x}})\,\mathbf{u}_e\,,$$

with a $3 \times 3k$ matrix $\mathbf{H}_e(\bar{\mathbf{x}})$ of basis function values $N^e_i(\bar{\mathbf{x}})$ and a vector $\mathbf{u}_e = [\mathbf{u}_1^T, \ldots, \mathbf{u}_k^T]^T$ of e's nodal displacements.

Discrete Energy. Since the Cauchy strain is linear in the displacements, it can also be written in Voigt notation as the 6×1 vector

$$\boldsymbol{\epsilon}(\mathbf{u}^e(\bar{\mathbf{x}})) \;=\; \mathbf{B}_e(\bar{\mathbf{x}})\,\mathbf{u}_e\,,$$

holding the six distinct values of the corresponding strain tensor. The $6 \times 3k$ matrix $\mathbf{B}_e(\bar{\mathbf{x}})$ is built from gradients $\nabla N^e_i(\bar{\mathbf{x}})$. Using this matrix notation of

each element's strain, the global elastic energy (3.34) of the deformed model is then formulated as

$$\mathrm{E}(\mathbf{u}) \;=\; \sum_e \mathbf{u}_e^T \underbrace{\left(\frac{1}{2} \int_e \mathbf{B}_e^T \mathbf{C} \mathbf{B}_e d\bar{\mathbf{x}} \right)}_{=: \mathbf{K}_e} \mathbf{u}_e \;=\; \frac{1}{2} \mathbf{u}^T \mathbf{K} \mathbf{u}\,, \tag{4.9}$$

with $\mathbf{K}_e$ denoting the $3k \times 3k$ stiffness matrix of element e, which are assembled into the global stiffness matrix $\mathbf{K}$.

Quadrature. For a general polyhedral element e, the computation of its stiffness matrix $\mathbf{K}_e$ requires numerical integration, since the derivative matrix $\mathbf{B}_e(\bar{\mathbf{x}})$ is not constant, as in the special case of linear tetrahedra. We employ either bounding box subdivision and Gaussian quadrature or Monte Carlo integration instead of the heuristic integration proposed by [Wicke et al., 2007], since the latter degrades for non-convex elements. For linear elasticity, the matrices $\mathbf{K}_e$ can be precomputed, such that the integration has to be performed only once for each element. As a consequence, the run-time complexity of our simulation is not higher than that of a simulation using a tetrahedral or hexahedral discretization.

Corotation. Since the linear strain is not rotation-invariant, even rigid-body motions will give rise to strain, which in turn causes ghost forces. This can be remedied by adapting one of the corotation approaches we have seen in Section 3.3.3 to general polyhedral elements [Wicke et al., 2007]: The rotation $\mathbf{R}_e$ of the element's displacement $\mathbf{U}_e$ is extracted using shape matching, and is factored out by correcting the stiffness matrix as $\mathbf{K}_e \leftarrow \mathbf{R}_e \mathbf{K}_e \mathbf{R}_e^T$. This correction has to be performed in each time step, and the global stiffness matrix $\mathbf{K}$ needs to be updated accordingly. Again, the complexity for these computations is of the same order as for tetrahedral or hexahedral simulations. Alternatively, in order to prevent rotation artifacts for large elements that can undergo large deformations (and rotations), corotation could also be performed at the quadrature point level (Section 3.3.3), making the cost of assembly depending on the total number of quadrature points instead of elements.

Equations of Motion. With the discrete energy (4.9), the governing equation for a dynamically deforming elastic solid becomes

$$\mathbf{M}\ddot{\mathbf{u}} \;+\; \mathbf{D}\dot{\mathbf{u}} \;+\; \mathbf{K}\mathbf{u} \;=\; \mathbf{f}_{\text{ext}}\,, \tag{4.10}$$

with mass matrix $\mathbf{M}$, damping matrix $\mathbf{D}$, and the vector $\mathbf{f}_{\text{ext}}$ containing external forces. In all our tests we used a standard semi-implicit Euler method (3.39) for the robust time-integration of (4.10). We also employ the simple nodal collision detection and handling approach discussed in Section 3.1.3 and Section 3.3.2.

4.2.4 Results

In this section we demonstrate the versatility of our polyhedral finite element framework on adaptive refinement and progressive cutting, and give statistics and comparisons of our method. Most examples show individual frames of simulations that are also included in the accompanying video.

Non-Convex Elements. As a proof of concept, Fig. 4.1 shows a simulation of a simple, non-convex element, whose shape functions are computed using $n = 97$ RBF centers in (4.5). While non-convex elements undoubtedly increase the meshing flexibility, we also note that in practice a restriction to convex elements might be preferable, for instance for efficient collision detection.

2D Adaptive Refinement. Fig. 4.4 shows a quantitative analysis of our method based on a 2D Poisson problem $-\Delta u = f$ with known analytic solution. We compare convergence behavior and condition numbers of uniform refinement of standard bilinear FEM, adaptive refinement of bilinear basis functions with CHARMS [Grinspun et al., 2002], and our quadtree-like adaptive element refinement (Fig. 4.5).

While condition numbers increase similarly with the number of DOFs for all methods, adaptive refinement makes better use of the DOFs than uniform refinement. When comparing CHARMS to our refinement technique, the former can also be used for higher order basis functions, whereas our method is a generalization of linear shape functions only. However, it allows for more flexible splits without the need to balance neighboring elements' refinement levels (see below). The plots also show that the number of sources $\mathbf{s}_i$ (resp. of RBF kernels $\mathbf{k}_i$) used for individual basis functions N_i^e has basically no effect on the global approximation error.

Embedding and Adaptive Mesh Generation. With the method described so far, we can simulate deformable objects that are discretized by general polyhedral elements. However, a straightforward volume tessellation works

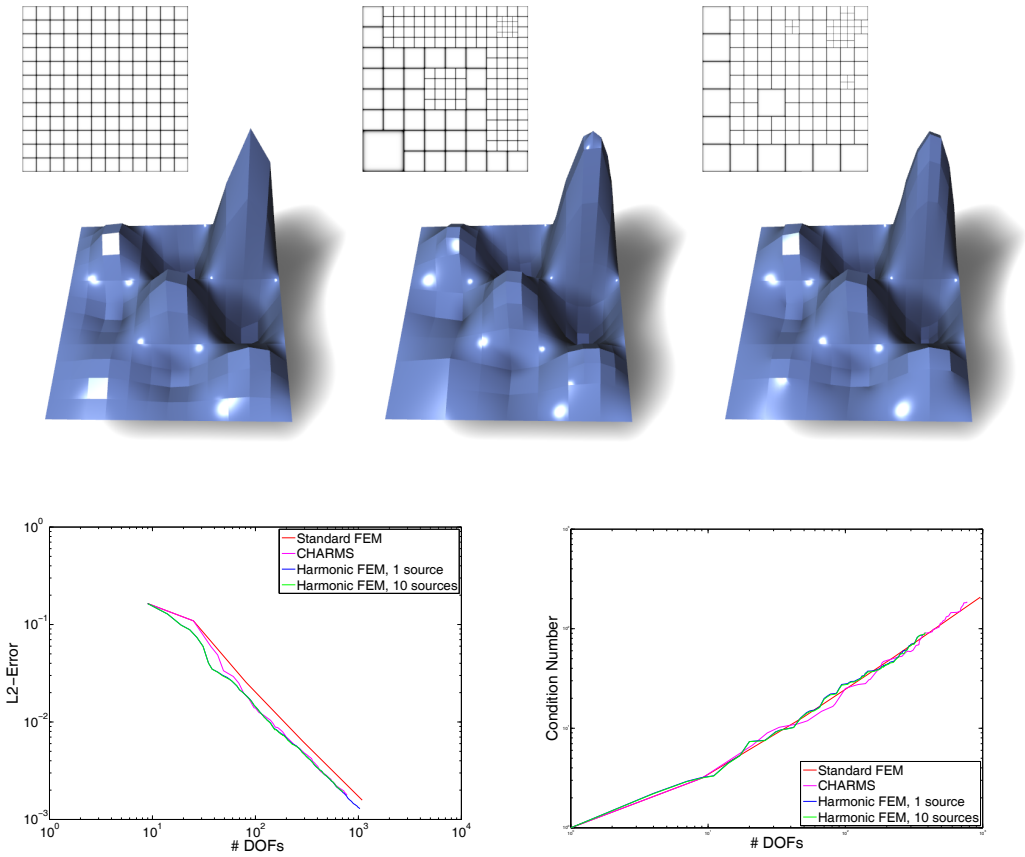


Figure 4.4: *For a 2D Poisson problem with known solution, we compare bilinear FEM (top left), adaptive basis function refinement of CHARMS (top middle), and our adaptive element refinement (top right). The graphs (bottom) compare approximations for grids of about the same number of DOFs. The plots show* L^2 *errors and condition numbers of* $\mathbf{K}$ *for increasing numbers of DOFs.*

for clean, moderately complex objects only, but becomes problematic for highly complex or topologically inconsistent models (e.g., scanned data, point-based models).

To be able to also handle such objects, we adapt a space embedding technique discussed in Section 3.3.3. In a preprocessing step, we voxelize the object into hexahedral elements, and then simulate the elastic deformation on the resulting voxels only. The high resolution surface mesh is deformed by interpolating the displacement within the voxels according to (3.34). However, since the complexity of regular grids grows cubically under refinement, this approach can handle moderate grid resolutions only.

Exploiting the flexibility we gain from arbitrary element shapes, a hierarchical, *adaptive* refinement is very easy to implement. Similar to Botsch et al. [2007],

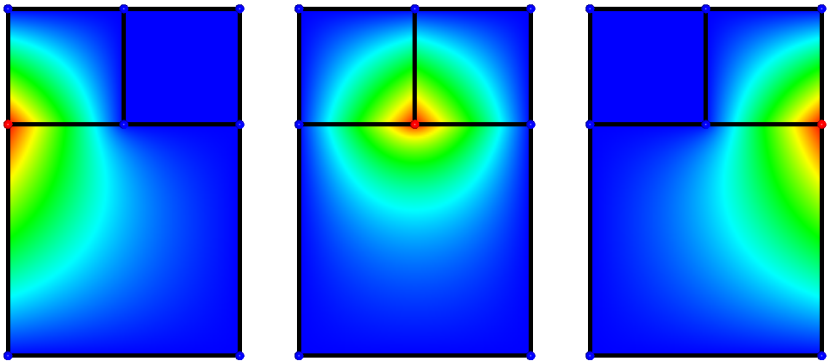


Figure 4.5: *A quadtree element with neighbors at a higher refinement level. Shown are the basis functions for the shared nodes, where the center one is not a hanging node, but instead is part (and DOF) of all three elements.*

we employ an octree-like discretization that refines nodes near the embedded surface. Note that this does not lead to hanging nodes in our discretization. Since the elements need not be strictly hexahedral, faces between octree cells of different depth do not require special handling, as illustrated in Fig. 4.5.

In Fig. 4.6, the Stanford bunny is embedded in an adaptive octree-like simulation mesh, which concentrates the DOFs at the more interesting boundary surface. Supporting general polyhedral elements makes this kind of adaptive embedding both easy to implement and to maintain, thereby enabling the efficient simulation of highly complex or topologically inconsistent meshes.

For embedded simulations we perform collision handling on the vertices of the embedded surface, instead of on the simulation nodes (Fig. 3.8). Similar to [Sifakis et al., 2007b], (penalty) forces applied to an embedded vertex $\bar{\mathbf{x}} = \sum_i \bar{\mathbf{x}}_i N_i(\bar{\mathbf{x}})$ simply have to be distributed to the simulation nodes $\bar{\mathbf{x}}_i$, weighted by the (generalized) barycentric coordinates $N_i^e(\bar{\mathbf{x}})$.

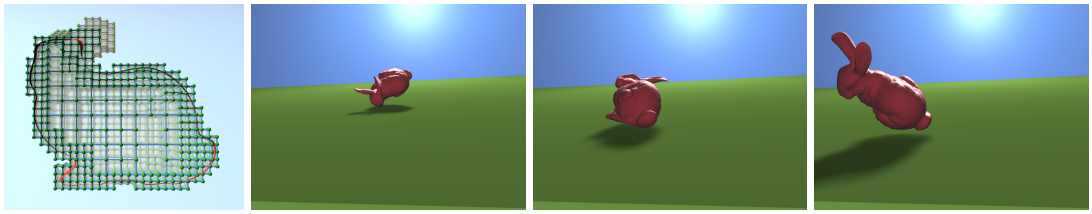


Figure 4.6: *The bunny model is embedded into an adaptively refined, octree-like simulation mesh, shown on the left. The degrees of freedom are concentrated on the surface, wasting little computing power on the less interesting, invisible interior.*

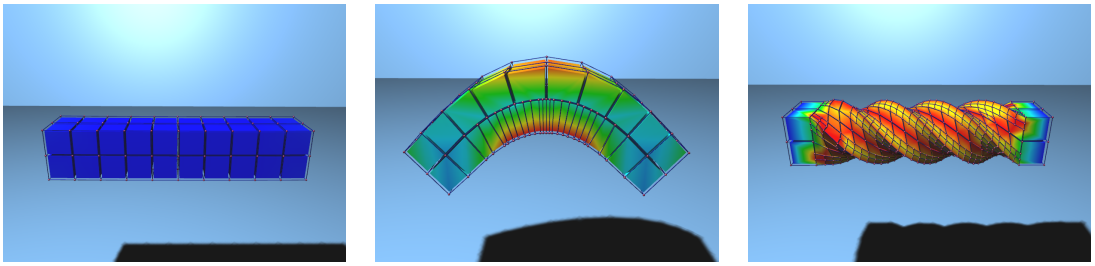


Figure 4.7: *Dynamic, stress-based refinement of a hexahedral bar model, using 1-to-2 splits for the bending deformation and 1-to-8 subdivision for twisting. The bar is constrained at both ends, the color visualizes the maximum principal stress.*

Stress-Based Dynamic Refinement. Polyhedral elements allow for dynamic element refinement: Similar to fracture [O'Brien and Hodgins, 1999; O'Brien et al., 2002], we sample the stress tensor $\sigma(\bar{\mathbf{x}})$ at a few points within each element, compute the principal stress as the largest absolute eigenvalue, and refine an element once a certain threshold is reached. Fig. 4.7 illustrates this for two kinds of adaptive refinement: A uniform 1-to-8 subdivision of voxels, and the more flexible 1-to-2 splitting perpendicular to the maximum stress direction, which results in fewer elements for the same refinement threshold.

Progressive Cutting. As proposed in [Wicke et al., 2007], supporting general polyhedra in FEM simulations effectively avoids the need for complex remeshing during cutting and thus considerably simplifies the implementation. Our harmonic basis functions seamlessly integrate into both tetrahedral

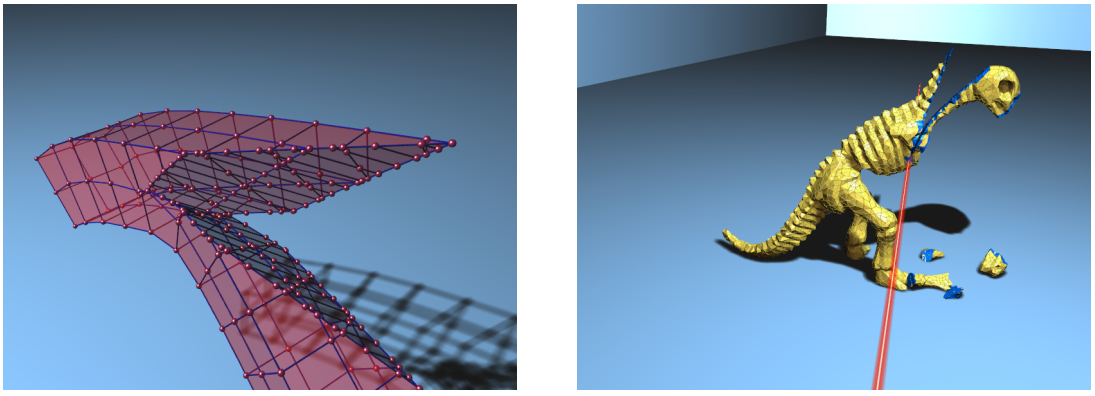


Figure 4.8: *Left: Progressive cutting of a hexahedral bar model. Right: Cutting a tetrahedral dinosaur mesh. Tetrahedra are visualized in yellow, general polyhedra in blue.*

and hexahedral simulations, where then only the cut elements have to be computed as harmonic polyhedral elements (Fig. 4.8). Arbitrary cuts can lead to non-convex elements with small opening angles, which complicate off-setting RBF centers. To avoid this problem, and to simplify the actual element splitting and collision detection, our cutting algorithm generates convex elements only, following [Wicke et al., 2007].

Timings. For the examples shown, Table 4.1 summarizes model complexities and timings, taken on an Intel Core2 Duo, 2.4 GHz. Solving the linear systems takes about the same time as for standard FEM, with only a slight increase in matrix density in case of complex polyhedra with high vertex count. Solving for and numerically integrating shape functions N_i^e is considerably more expensive than for simple linear tetrahedra or trilinear hexahedra. Note, however, that general polyhedral elements are employed in irregular regions of adaptivity and cutting only, whereas in regular regions we can use standard elements. Our approach therefore trades the combinatorial complexity of remeshing for the computational complexity of polyhedral elements.

Scene	Start #N/#E	End #N/#E	t_{init}	t_{solve}	t_{total}
Collision (Fig. 3.8)	274 / 153	274 / 153	105	5.2	20
Bunny (Fig. 4.6)	4.8k / 3k	4.8k / 3k	59	221	247
Bending (Fig. 4.7)	99 / 40	256 / 88	302	1.6	60
Twisting (Fig. 4.7)	99 / 40	1392 / 719	170	37	276
Bar Cut (Fig. 4.8)	99 / 40	391 / 63	688	3	235
Dino Cut (Fig. 4.8)	5.6k / 19k	9.3k / 21k	48	113	752

Table 4.1: *Statistics and timinigs for the shown examples. We list initial and final number of nodes (#N) and elements (#E), avg. time to compute N_i^e and setup $\mathbf{K}_e$ per* polyhedral *element, and for the linear solve per time-step (both [ms]), and the total simulation time [s].*

4.3 Meshfree Representations

So far we saw how moving from simple representations of the subspace using (tri-) linear basis functions to more advanced representation bases such as harmonic coordinates enables much simpler handling of adaptivity, cutting and fracturing. However, this formulation of advanced basis functions requires always an underlying mesh structure. Requiring a mesh for discretization is not always the optimal choice: for example when a material undergoes large, possibly plastic, deformations, the distorted discretization structure needs to be adapted in order to maintain quality and accuracy leading to frequent remeshing operations [Wojtan and Turk, 2008].

In these cases it is advantageous to have a discretization structure which is as simple as possible. For this reason, purely point-based discretization techniques have been pursued in order to maximally ease discretization: the simple placement of points in space [Müller et al., 2004a; Pauly et al., 2005; Adams et al., 2008; Gerszewski et al., 2009]. Common to all these methods is the use of *moving least squares (MLS)* as scattered data interpolation procedure to spatially interpolate the pointwise defined DOF values [Fries and Matthies, 2004].

While most these mentioned methods use MLS directly to interpolate deformation gradients in collocation-based schemes, we will use MLS to represent the discretization subspace, similar to Adams et al. [Adams et al., 2008], to perform again a Galerkin discretization — a procedure commonly seen in meshfree methods in computational mechanics [Belytschko et al., 1994; Krysl and Belytschko, 1996; Fries and Matthies, 2004].

Advantages of Meshfree Methods. Meshfree methods allow for the simplest possible discretization structure – only point distribution is required. While this leads to considerably simpler *h-refinement* than with mesh-based methods, also changing the interpolation order, i.e., *p-refinement* is conceptually simpler in this class of methods since increasing the polynomial order only affects the number of required neighboring particles but is enclosed in the general formulation of MLS interpolation.

While using different polynomial orders in classic FEM is well-established and allows for higher order solution continuity inside the elements, it is still a difficult task to construct suitable basis functions satisfying global continuity requirements [Li et al., 2004]. Contrary, meshfree methods allow to easily have *any desired order of continuity* which is only determined by the continuity of the employed weight function.

4.3.1 MLS Basis Functions

Scattered Data Interpolation. Orginally, MLS was introduced by Lancaster and Salkauskas [1981] as interpolation scheme for scattered data. Only later its generality has been discovered and has been applied in many scientific fields (see Fries and Matthies [2004] for a thorough discussion).

Assume that sample points $\bar{\mathbf{x}}_1,\ldots,\bar{\mathbf{x}}_n$ are given inside the domain Ω, each with an associated function value (in our case the displacement DOFs) $\mathbf{u}_i \in \mathbb{R}^3$. Then, around a given point $\hat{\bar{\mathbf{x}}}$, the displacement field $\mathbf{u}(\bar{\mathbf{x}})$ can locally be approximated by a polynomial $\mathbf{a}(\hat{\bar{\mathbf{x}}})^T\mathbf{p}(\bar{\mathbf{x}})$ with a vector of monomials $\mathbf{p}(x,y,z) = (1,x,y,z)^T$ (for the linear case) and coefficients $\mathbf{a}(\hat{\bar{\mathbf{x}}})$. These coefficients are determined by a *local* weighted least squares fit to the displacement values $\mathbf{u}_i$. That is, for the point of interest $\hat{\bar{\mathbf{x}}}$, an error function is constructed measuring the approximation error between every data $\mathbf{u}_i$ and its polynomial approximation $\mathbf{a}(\hat{\bar{\mathbf{x}}})^T\mathbf{p}(\bar{\mathbf{x}}_i)$ weighted by the sample distance to the evaluation point. Using a weight function of the form $w(\bar{\mathbf{x}} - \bar{\mathbf{x}}_i)$, we can define this error function as

$$J(\mathbf{a}(\hat{\bar{\mathbf{x}}})) \;=\; \frac{1}{2}\sum_{i=1}^{n} w(\hat{\bar{\mathbf{x}}} - \bar{\mathbf{x}}_i)\left\|\mathbf{a}^T\mathbf{p}(\bar{\mathbf{x}}_i) - \mathbf{u}_i\right\|^2. \tag{4.11}$$

In our implementations, we usually applied the commonly used weighting kernel $w(\mathbf{d}) = (1 - \|\mathbf{d}\|^2)^3$ (see Fries and Matthies [2004] for more discussion on weighting kernels). Fig. 4.9 gives a simple example of a one dimensional MLS fit.

The next step then consists in finding the best polynomial approximation to the data, i.e., to minimize the error function $J(\mathbf{a}(\hat{\bar{\mathbf{x}}}))$. The derivative of the error function is

$$\frac{\partial J}{\partial \mathbf{a}(\hat{\bar{\mathbf{x}}})} = \sum_{i=1}^{n} w(\hat{\bar{\mathbf{x}}} - \bar{\mathbf{x}}_i)\mathbf{p}(\bar{\mathbf{x}}_i)(\mathbf{a}(\hat{\bar{\mathbf{x}}})^T\mathbf{p}(\bar{\mathbf{x}}_i) - \mathbf{u}_i).$$

Setting it to zero and solving the system for the coefficient vector yields

$$\mathbf{a}(\hat{\bar{\mathbf{x}}}) = (\sum_{i=1}^{n} w(\hat{\bar{\mathbf{x}}} - \bar{\mathbf{x}}_i)\mathbf{p}(\bar{\mathbf{x}}_i)\mathbf{p}(\bar{\mathbf{x}}_i)^T)^{-1}\sum_{i=1}^{n} w(\hat{\bar{\mathbf{x}}} - \bar{\mathbf{x}}_i)\mathbf{p}(\bar{\mathbf{x}}_i)\mathbf{u}_i.$$

That is, with this coefficient vector we get the optimal polynomial approximation $\mathbf{a}(\hat{\bar{\mathbf{x}}})^T\mathbf{p}(\bar{\mathbf{x}})$ for the point $\hat{\bar{\mathbf{x}}}$. If we now use this local fit only for evaluating the exact point $\hat{\bar{\mathbf{x}}}$ and define the same approximation for arbitrary $\hat{\bar{\mathbf{x}}}$, we can simplify notation and use $\bar{\mathbf{x}} = \hat{\bar{\mathbf{x}}}$ giving us the continuous MLS approximation to the data as

$$\mathbf{u}(\bar{\mathbf{x}}) = \mathbf{p}(\bar{\mathbf{x}})^T(\sum_{i=1}^{n} w(\hat{\bar{\mathbf{x}}} - \bar{\mathbf{x}}_i)\mathbf{p}(\bar{\mathbf{x}}_i)\mathbf{p}(\bar{\mathbf{x}}_i)^T)^{-1}\sum_{i=1}^{n} w(\hat{\bar{\mathbf{x}}} - \bar{\mathbf{x}}_i)\mathbf{p}(\bar{\mathbf{x}}_i)\mathbf{u}_i. \tag{4.12}$$

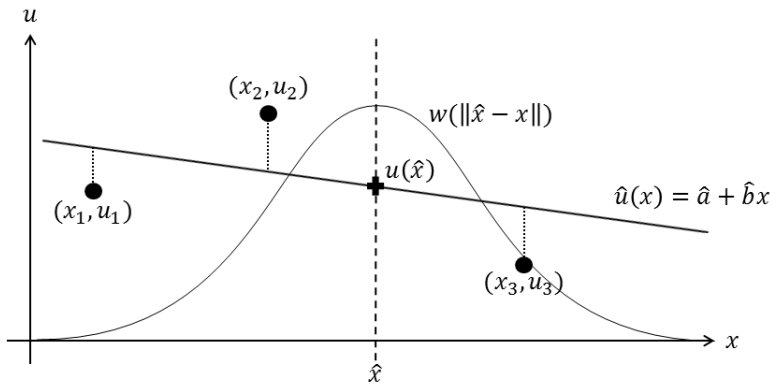


Figure 4.9: *A simple example of a linear 1D MLS fit: For position $\hat{x}$, a linear polynomial $\hat{a}x + \hat{b}$ is fitted to the three data points (x_1, u_1), (x_2, u_2) and (x_3, u_3) by minimizing the weighted sum of vertical errors (dotted lines): $\sum_1^3((\hat{a}x_i + \hat{b}) - u_i)^2 w(|\hat{x} - x_i|)$.*

Subspace Representation. Having a closer look at the continuous approximation Eq. (4.12) reveals a very important structure of the approximation. By rearranging terms, we can divide the data vectors $\mathbf{u}_i$ from the interpolation specific terms and arrive to the well-known form of a linear combination of DOF deformation vectors $\mathbf{u}_i$ with basis functions $N_i(\bar{\mathbf{x}})$, i.e., MLS also spans a linear subspace of a infinite dimensional function space (for C^∞-continuous weight functions, a subspace of $C^\infty(\Omega)$ is spanned). That is, we can write

$$\begin{aligned} \mathbf{u}(\bar{\mathbf{x}}) &= \sum_{i=1}^{n} \mathbf{u}_i N_i(\bar{\mathbf{x}}), \\ N_i(\bar{\mathbf{x}}) &= \mathbf{p}(\bar{\mathbf{x}})^T \mathbf{G}^{-1}(\bar{\mathbf{x}})\, \mathbf{p}(\bar{\mathbf{x}}_i)\, w(\bar{\mathbf{x}} - \bar{\mathbf{x}}_i), \\ \mathbf{G}(\bar{\mathbf{x}}) &= \sum_{i=1}^{n} w(\bar{\mathbf{x}} - \bar{\mathbf{x}}_i)\, \mathbf{p}(\bar{\mathbf{x}}_i)\, \mathbf{p}(\bar{\mathbf{x}}_i)^T, \end{aligned} \tag{4.13}$$

where $\mathbf{G}(\bar{\mathbf{x}})$ is called *moment matrix* which should be non-singular in order to be invertible. This is not the case if either too few data points support the evaluation point $\bar{\mathbf{x}}$ or the data points are located co-linearly or co-planarly which makes intuitive sense since in this case no information on how the data evolves in the missing directions is given. In Chapter 5 we will see a variant of this approach being less stringent on the actual sample requirements.

In contrast to FEM basis functions, one can see that MLS basis functions do actually have a more complex algebraic form: Instead of simple polynomials they are of rational nature. This allows them to adapt nicely to almost arbitrary point samplings and giving them the good continuity and approximation properties.

Basis Function Properties. In Section 3.3 we have seen that suitable basis functions for the Galerkin discretization of the elasticity PDE need to fulfill a handful of requirements:

- **Continuity** Basis functions need to lie in Sobolev space $H^1(\Omega)$. As just discussed, MLS basis function have very good continuity properties that depend on the continuity of the weighting function $w(\cdot)$. Since we choose them to be C^∞-continuous, so are the basis functions, i.e., $N_i \in C^\infty(\Omega) \subset H^1(\Omega)$.
- **Consistency of order** n MLS basis functions using n-th order polynomials are called consistent of order n. That is, a n-th order MLS basis can reproduce polynomials *exactly* up to order n. In order to get the constant and linear reproduction property it is therefore sufficient to use at least linear polynomials in the construction of the MLS basis (see also Section 4.2.3 in [Fries and Matthies, 2004] for a proof for consistency of MLS).

4.3.2 Simulation

For actually performing a solid simulation with MLS basis functions we can mostly rely on the techniques presented in Chapter 3. Let us discuss some choices that we made for implementing a straightforward meshless simulator:

- **Sampling** In order to perform the discretization we basically require two point sets: the DOF and the quadrature point locations. In our implementations we choose the following simple procedure to generate them (see also [Adams et al., 2009]).

  First, we cover the domain with a regular grid, whose resolution is usually chosen large enough to get also a good sampling of thin geometric features of the domain. This generates us the *candidate* samples. Then, we perform *furthest distance sampling* on these candidates in order to determine the DOF locations: The first point is chosen randomly, the following ones by choosing the point among the candidates which is furthest away from the previously chosen ones. Such a strategy leads to a regular sample distribution and allows increasing the number of samples progressively.

  For the quadrature points we have different possibilities: Either we perform a Gauss quadrature on a background regular mesh, as performed when integrating the polyhedral elements. Or we can apply

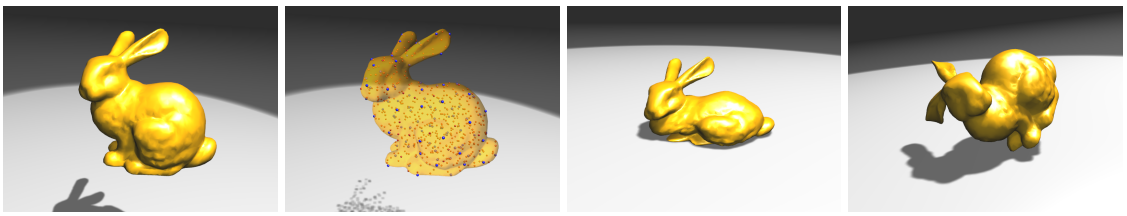


Figure 4.10: *The described approach allows for very simple discretization (second) by just placing DOF (red) and quadrature (blue) samples inside the computation domain. The resulting simulation performance is comparable to corotated FEM while giving visually pleasing results due to the high smoothness property of MLS shape functions.*

the same technique as just described to choose also the quadrature point locations. By choosing the same number of quadrature points as DOFs we arrive at a simple collocation scheme (and need also to determine these samples only once). While such an approach is very simple it bears accuracy issues (see Section 4.3.3). We can however also progressively increase the quadrature density in order to capture the energy coupling between the DOFs better and better – at increasing costs however.

- **Precomputation** As soon as the DOF and quadrature point locations are known, the precomputation procedure is pretty similar to the one known from corotated FEM simulations. However, since there is no notion of elements, precomputation and assembly move from being element-centric to quadrature-centric: For each quadrature point, the basis functions, their derivatives as well as the local stiffness matrix are precomputed and stored individually (Eq. (3.34)). Furthermore, if an embedded surface mesh is used for visualization, the basis functions are also precomputed at the vertex locations such that only fast linear combinations have to be computed at runtime to get the deformed surface mesh (Eq. (3.37)).
- **Runtime** During runtime, almost the same steps are performed as for corotated FEM except that again the *assembly* is quadrature-centric. In each timestep, rotations need to be first determined per quadrature point by performing a polar decomposition [Hauth and Strasser, 2004] of the deformation gradient $\mathbf{F} = \mathbf{I} + \nabla\mathbf{u}$. Using these, we can compute the actual pointwise internal forces and Jacobians in a corotated fashion (see Section 3.3.3). These (quadrature-) pointwise quantities are then assembled into a global force vector and Jacobian matrix, which are then in turn used to perform time integration (see Section 3.3.5).

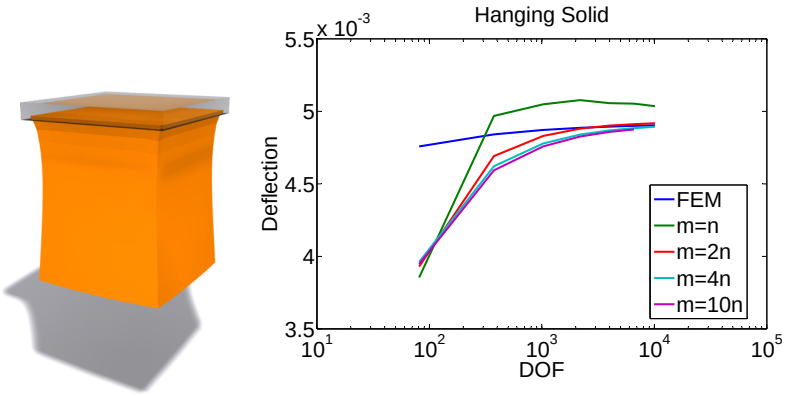


Figure 4.11: *A solid cube fix on its top side and hanging under gravity. Right: The convergence plot compares the maximal displacement for classic FEM and the MLS-based meshfree approach for different ratios of #DOFs(n) to #quadrature points(m).*

4.3.3 Results

Dynamics. Substituting classic FEM basis functions by the more flexible MLS basis functions leads to a very simple simulation framework. In contrast to corotated FEM [Müller et al., 2002; Müller and Gross, 2004], the presented approach allows for simpler discretization without the need of generating qualitative volumetric meshes, but just requires the distribution of individual points inside the computation domain. Due to the high smoothness property of MLS, the visual results are superior to the piecewise linear solution obtained by linear FEM. As for FEM, the main computational tasks consist in preprocessing, assembly and linear solves. While preprocessing is more demanding due to the computation of MLS basis functions, the assembly is comparable or even superior depending on the chosen integration accuracy. Also the density of the linear systems and therefore the solving performance is comparable to FEM. Fig. 4.10 shows the approach applied to a bunny model.

Quadrature. The accuracy of the resulting method depends strongly on the accuracy of the quadrature for the energy integration. While realistic results can already be achieved with a simple collocation scheme (where DOFs and quadrature points are collocated), more accurate results are achieved by a better energy integration. Fig. 4.11 shows a comparison of the accuracy for different quadrature sampling densities.

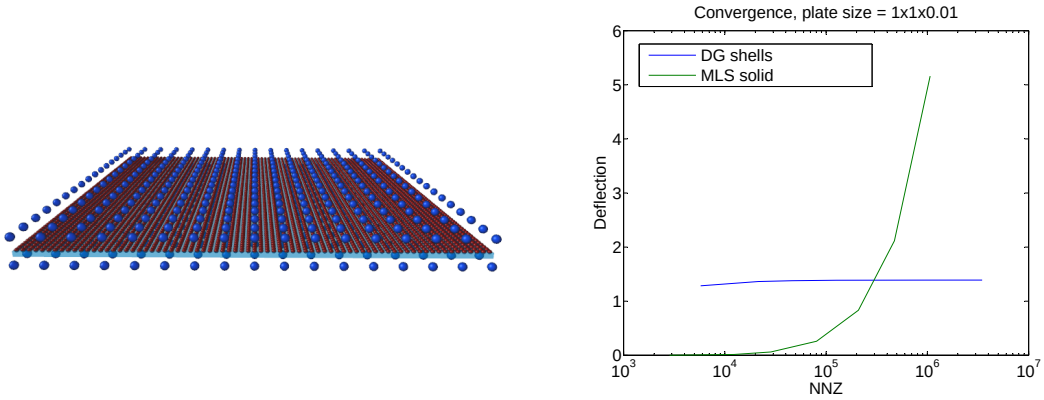


Figure 4.12: *When applying the MLS-based approach to a clamped thin plate under gravity (two DOF layers in red, single quadrature layer in blue), the presented approach is not able give convergent results, as opposed to the DG shell model that converges to the correct analytic solution.*

Close Independent Features. While in the FEM the individual elements sharply describe the influence domain of the basis functions, the support is not determined that flexibly in the basic meshless approach described so far. While it is possible to choose complex influence domains per DOF sample [Fries and Matthies, 2004; Adams et al., 2009], our basic approach chosen here just uses spherical support domains with adaptive sphere radii. This simple choice however can lead to unwanted artifacts when unconnected regions fall into the same support of a DOF and get unintentionally coupled. Such artifacts can be prevented in different ways: Increasing the sampling density (with decreased support radii) resolves the problem to some point, however depends on the actual feature size in the problematic regions and can introduce unwanted high number of DOFs. A better choice is the one taken by [Pauly et al., 2005] or [Adams et al., 2009]) who introduce a *material distance metric* that elegantly creates complex DOF influence domains. In the next chapter, we will see an alternative solution that introduces a *meshless virtual node algorithm* [Molino et al., 2004] to resolve the problem.

Thin Structures. As the FEM, the meshless approach presented so far is not well suited to handle geometries with thin-walled structures, due to numerical stability problems. Moreover, as shown in Fig. 4.12, the presented approach is not even convergent: a single quadrature layer on the structure's midsurface together with a double layer of DOFs (required by the volumetric MLS sampling) is not capable to converge to the static (analytic) solution of the problem, as opposed to the DG shell model of [Kaufmann et al., 2009a].

As we will see in the next chapter, it is possible to generalize the basic method presented here to also support shell- and rod-like structures in a simple and general way.

4.4 Feature Preservation

Due to its sound theoretical guarantees the FEM is widely applied in engineering applications where stability and accuracy of the solution are crucial. Typical graphics applications have somewhat different requirements. High accuracy is less important, while efficiency, visual plausibility and aesthetics are paramount. Often, these goals can be achieved by a larger variety of approaches than in engineering. As an example, the purely geometric shape matching approach of Müller et al. [2005] shows excellent efficiency and stability properties, while just mimicking physical behavior.

For the task of geometric modeling, a large variety of different techniques exist, too. Space deformation techniques define the deformation of a 3D space, causing the embedded geometry to deform accordingly. Coordinate-based techniques, where the deformed space is restricted to a closed domain, have been investigated extensively in the past years. In particular, Lipman et al. [2008] presented Green coordinates (GC) that have the property of being shape preserving, in the sense that mappings possess minimal angular distortion. This helps to greatly improve the visual quality of the deformed geometry as shown in Fig. 4.13.

While these space deformation techniques are directly applicable to static geometric modeling, the automatic generation of full animation sequences is more difficult to realize. In this section we therefore present an approach on how to include the particular (non-linear) subspace spanned by Green coordinates into a Galerkin discretization in order to animate deformable objects in a shape-preserving manner — analogous to how the GC approach preserves shape for modeling. Independently of the discretization resolution, such an approach allows to preserve small features. In contrast, coarse linear

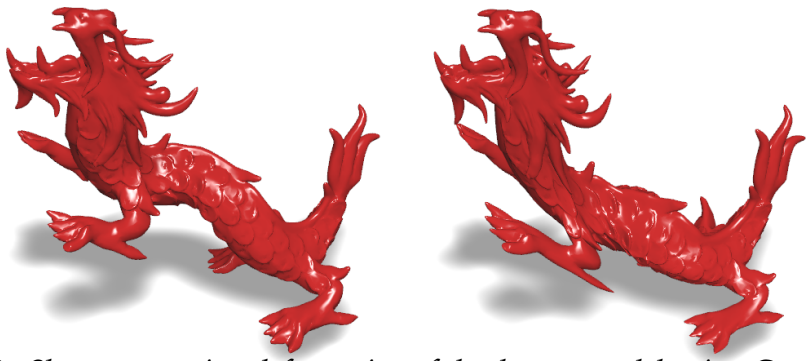


Figure 4.13: *Shape preserving deformation of the dragon model using Green coordinates as basis functions (left, 156 DOFs) compared to FEM-based deformation (right, 162 DOFs).*

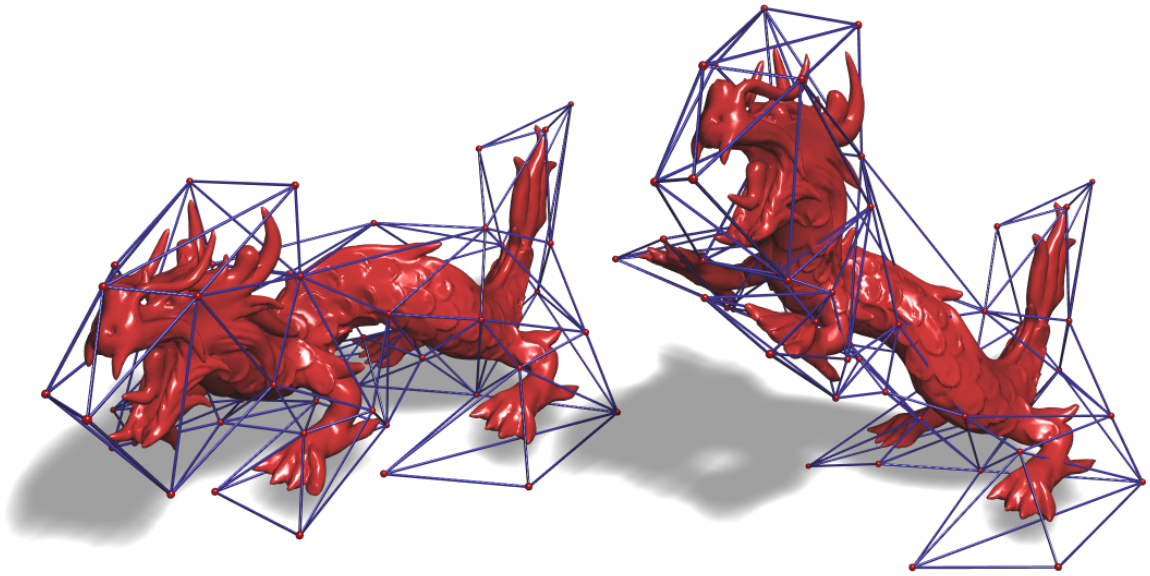


Figure 4.14: *Embedded dragon model in undeformed (left) and deformed cage (right).*

FEM approaches result in locally affine deformations that distort features at sub-element scales.

4.4.1 Green Coordinates

Let us first shortly review the important definitions and properties of GC, presented originally in [Lipman et al., 2008]. Similar to other commonly used coordinates, GC are also cage-based and thus defined inside a closed polytope. Classic barycentric coordinates describe a point inside the cage as a linear combination of cage vertices. Deforming the cage's shape then leads to a deformation of the enclosed space (Fig. 4.14). GC additionally incorporate the faces' normals in their formulation to achieve shape-preserving deformations, i.e., deformations that locally only allow a limited amount of change in angles. Achieving this property is not possible using mean value coordinates or harmonic coordinates [Floater, 2003; Joshi et al., 2007], since they are affine invariant [Lipman et al., 2008]; affine transformations like shearing and anisotropic scaling violate shape-preservation.

Subspace Representation. More formally, GC define how a point $\bar{\mathbf{x}}$ in the interior of the undeformed cage is deformed to a new position $\mathbf{x}$ by a linear combination of vertex positions $\mathbf{x}_i$ and face normals $\mathbf{n}_j$ of the deformed cage as

$$\mathbf{x}(\bar{\mathbf{x}}) = \sum_{i \in I_V} \mathbf{x}_i N_i(\bar{\mathbf{x}}) + \sum_{j \in I_T} s_j \mathbf{n}_j M_j(\bar{\mathbf{x}}), \tag{4.14}$$

where I_V is the set of all vertices and I_T the set of faces. While GC can be defined on general cage types we will focus on cages represented as triangle meshes for which they can be described analytically. In order to achieve shape preserving deformations, the scaling factor s_j has to be chosen as

$$s_j = \frac{\sqrt{|\bar{\mathbf{v}}|^2 |\mathbf{w}|^2 - 2(\bar{\mathbf{v}} \cdot \bar{\mathbf{w}})(\mathbf{v} \cdot \mathbf{w}) + |\mathbf{v}|^2 |\bar{\mathbf{w}}|^2}}{\sqrt{8}\,\mathrm{area}(t_j)}, \tag{4.15}$$

where $\bar{\mathbf{v}}$ and $\bar{\mathbf{w}}$ are two arbitrary undeformed edges of the triangle t_j, with corresponding deformed edges $\mathbf{v}$ and $\mathbf{w}$. It is important to note here that the normals nonlinearly depend on the cage vertex positions which will be important for the discretization.

Please note that the subspace represented by GC is spanned by basis functions $N_i(\bar{\mathbf{x}})$ and $M_j(\bar{\mathbf{x}})$ that are global, i.e., having support on the entire cage. While the chosen subspace is linear in conventional Galerkin discretizations, the subspace spanned by this GC representation is not: Since the face normals depend nonlinearly on the cage vertices (the actual DOFs), the mapping from *DOF* to *candidate functions* is nonlinear, forming a more general manifold as subspace. This is different from standard Galerkin approaches where this map is linear and leads to a linear subspace of H^1. Instead of a linear combination we have a non-linear combination of basis functions — a fact also becoming important when we will set up the discrete problem. Furthermore, we cannot rely anymore on the existence and convergence guarantees we had for linear subspaces. Nevertheless, we did not observe any problems due the lack of these properties.

Basis Functions. Using Green's function $G(\bar{\mathbf{x}}, \bar{\mathbf{x}}') = \frac{1}{4\pi|\bar{\mathbf{x}} - \bar{\mathbf{x}}'|}$ and the piecewise linear hat function $\Gamma_i(\bar{\mathbf{x}}')$ defined on the triangle mesh and centered at vertex i, the coordinates/basis functions are defined as

$$N_i(\bar{\mathbf{x}}) = \int_{\bar{\mathbf{x}}' \in \Omega_i} \Gamma_i(\bar{\mathbf{x}}') \frac{\partial G(\bar{\mathbf{x}}, \bar{\mathbf{x}}')}{\partial \mathbf{n}(\bar{\mathbf{x}}')} dS_{\bar{\mathbf{x}}'} \tag{4.16}$$

and

$$M_j(\bar{\mathbf{x}}) = -\int_{\bar{\mathbf{x}}' \in t_j} G(\bar{\mathbf{x}}, \bar{\mathbf{x}}') dS_{\bar{\mathbf{x}}'}, \tag{4.17}$$

where the integration domain Ω_i consists of all triangles adjacent to vertex i. Representing the cage by a triangle mesh allows analytic computation of these coordinates and their derivatives. For pseudocode of the actual coordinate computation we refer to [Lipman et al., 2008]. The computation of their derivatives, which we need additionally for strain computations, can be derived straightforwardly by taking the derivatives in the according lines of their pseudocode [Huber, 2009].

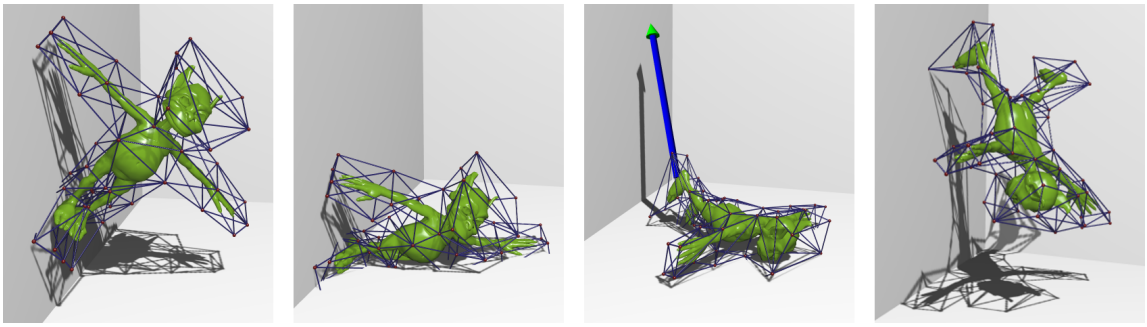


Figure 4.15: *Animation of the goblin model showing external forces and collisions.*

Subspace Properties. Although the spanned space is not linear, it can easily be checked that all other necessary conditions for a Galerkin discretization are fulfilled otherwise:

- Inside the cage, the coordinates are C^∞, which is sufficient since we will choose the computation domain to lie inside the cage.
- The hat functions Γ_i used in the definition of the vertex-based coordinates N_i guarantee the partition of unity property such that constant deformations can be represented.
- GC also reproduce linear functions as required for the representation of rotations [Lipman et al., 2008].

4.4.2 Numerical Approximation

Let us next have a look on how we discretize the elastic solid model with GC. In a first step, we formulate the problem nonlinearly and then show in a second step how we can introduce a suitable linearization to arrive at a simple simulation method.

Nonlinear Formulation

Domain Definition. In the last section we have seen that GC fulfill all requirements for a Galerkin discretization as long as the problem domain Ω resides inside the cage. The problem domain can be represented by a high-resolution triangle mesh embedded in the volume of the cage. We do not discuss how to generate cages for given domains since they can easily be designed by hand using arbitrary modeling tools. In the remainder we will assume that our domain is equipped with a suitable triangular mesh representing the cage.

Displacement Field. In order to formulate the Galerkin discretization for linear elasticity in the classical displacement-based manner, we first need a representation of the deformation field $\mathbf{u}(\bar{\mathbf{x}})$ in terms of GC. Representing both deformed and undeformed positions with the representation given in Eq. (4.14) and considering their difference, we get

$$\mathbf{u}_G(\bar{\mathbf{x}}) = \sum_i \mathbf{u}_i N_i(\bar{\mathbf{x}}) + \sum_j \mathbf{m}_j M_j(\bar{\mathbf{x}}), \tag{4.18}$$

where $\mathbf{u}_i = \mathbf{x}_i - \bar{\mathbf{x}}_i$ and $\mathbf{m}_j = \mathbf{n}_j s_j - \bar{\mathbf{n}}_j$ (since $s_j = 1$ for the undeformed state).

Energy and Forces. Following the same line as classic Galerkin discretizations that we have seen in Eq. (3.30), we can formulate the elastic energy using the new deformation field $\mathbf{u}_G$ as

$$W_G = \frac{1}{2} a(\mathbf{u}_G, \mathbf{u}_G) - f(\mathbf{u}_G). \tag{4.19}$$

In order to derive internal forces, we again take the derivative of W_G with respect to the degrees of freedom $\mathbf{u}_i$ (note that the $\mathbf{m}_j$ are nonlinear functions of the $\mathbf{u}_i$'s). Taking advantage of the linearity of $a(\cdot,\cdot)$ and $f(\cdot)$, the energy gradient reads as

$$\begin{aligned} \frac{\partial W_G}{\partial \mathbf{u}_i} &= \sum_j a(N_i, N_j)\mathbf{u}_j \\ &+ \sum_j a(N_i, M_j)\mathbf{m}_j \\ &+ \sum_{jk} \frac{\partial \mathbf{m}_j}{\partial \mathbf{u}_i}^T a(M_j, N_k)\mathbf{u}_k \\ &+ \sum_{jk} \frac{\partial \mathbf{m}_j}{\partial \mathbf{u}_i}^T a(M_j, M_k)\mathbf{m}_k \\ &- \sum_j \frac{\partial \mathbf{m}_j}{\partial \mathbf{u}_i}^T f(M_j) - f(N_i) \end{aligned} \tag{4.20}$$

which is obviously nonlinear in the DOFs. As mentioned before, we aim at using implicit Euler as time integration scheme, where also computationally more involved second order derivatives of the energy are required [Baraff and Witkin, 1998]. In order to simplify the resulting system and reduce the computational burden, we will therefore perform a linearization described in the following section.

Linearization

The complexity of the Galerkin discretization using GC results mainly from the nonlinear dependency between triangle normals and vertex positions. In order to simplify the discrete equations we make the following 0-th order approximation: We assume that the change in normals between two sufficiently small timesteps is negligible such that it can be ignored during the force computation. This means that we will treat the $\mathbf{m}_i$ as constant and update them only after each timestep using the updated cage vertices.

Using this linearization of the deformation field, Eq. (4.20) simplifies to

$$\begin{aligned} \frac{\partial W_G}{\partial \mathbf{u}_i} &= \sum_j a(N_i, N_j)\mathbf{u}_j \\ &+ \sum_j a(N_i, M_j)\mathbf{m}_j - f(N_i), \end{aligned} \tag{4.21}$$

which we can rewrite as

$$\frac{\partial W_G}{\partial \mathbf{u}} = \mathbf{K}\mathbf{u} + \mathbf{H}\mathbf{m} - \mathbf{f}, \tag{4.22}$$

where the additional stiffness matrix $\mathbf{H}$ relates normal differences to nodal forces and consists of 3×3 blocks $\mathbf{H}_{ij} = a(N_i, M_j)$.

Comparing Eq. (4.22) to Eq. (3.31) or Eq. (3.32), we note that we now have additional normal-based forces $\mathbf{Hm}$ next to the classic vertex-based forces $\mathbf{f}$. This is the only change that needs to be made to the basic simulation framework we have seen in Section 3.1.1.

4.4.3 Simulation

Algorithm Overview. Algorithm 1 summarizes the actual simulation loop. The blocks $\mathbf{K}_{ij}^q$ and $\mathbf{H}_{ij}^q$ can be precomputed since they remain constant throughout the simulation.

4.4.4 Results

Comparisons. In Fig. 4.16 and Fig. 4.13 we compare the presented approach to linear FEM where we embed the surface geometry into a hexahedral element mesh. For the conventional FEM simulations, the boundary conditions were applied in the same manner as described in Section 3.3.2. For the common situation where the simulation mesh is much coarser than the mesh

Set $\mathbf{K}$, $\mathbf{H}$ and $\mathbf{f}$ to zero
for all $i, j \in I_V$: // Stiffness matrix and BC assembly
  $\mathbf{K} \Leftarrow \mathbf{K}_{ij}^c$
  $\mathbf{f} \Leftarrow \mathbf{f}_i^c$
  for all quadrature points q:
    extract rotation $\mathbf{R}^q$
    $\mathbf{K} \Leftarrow \mathbf{R}^q \mathbf{K}_{ij}^q \mathbf{R}^{qT}$
    $\mathbf{f} \Leftarrow (\mathbf{R}^q \mathbf{K}_{ij}^q - \mathbf{R}^q \mathbf{K}_{ij}^q \mathbf{R}^{qT}) \bar{\mathbf{x}}_j$
  end
end
for all $i \in I_V, j \in I_T$:
  $\mathbf{H} \Leftarrow \mathbf{H}_{ij}^c$
  for all quadrature points q:
    $\mathbf{f} \Leftarrow -\mathbf{R}^q \mathbf{H}_{ij}^q (\mathbf{R}^{qT} \mathbf{n}_j s_j - \bar{\mathbf{n}}_j)$
  end
end
compute forces according to Eq. (3.32)
implicit integration of $\mathbf{M}\frac{\partial^2 \hat{\mathbf{u}}}{\partial t^2} + \mathbf{K}\hat{\mathbf{u}} = \mathbf{f}$
update $\mathbf{m}$ with new positions of cage vertices

Algorithm 1: Summary of the simulation loop.

of the embedded surface, our method clearly shows better handling of the fine-scale details. Fig. 4.15 shows an animation sequence demonstrating the handling of boundary conditions, collisions and external forces.

Modeling. The use of boundary conditions in combination with solving the static problem gives also raise to a simple and intuitive GC modeling tool. In contrast to cage-based modeling, where a shape is deformed indirectly over cage manipulations, our approach allows direct modeling of the shape. Fig. 4.17 shows the armadillo model with boundary constraints at its extremities and how they can be manipulated for modeling the geometry. Since boundary conditions are imposed vertex-wise, different effects can be achieved by fixing one, two, or more vertices. Constraining only a single vertex prescribes only its position but no orientation. Constraining two vertices results in one remaining rotational degree of freedom, while fixing more than two vertices constrains position and orientation. We constrained one vertex in each hand of the armadillo and all feet vertices. Furthermore, solving the

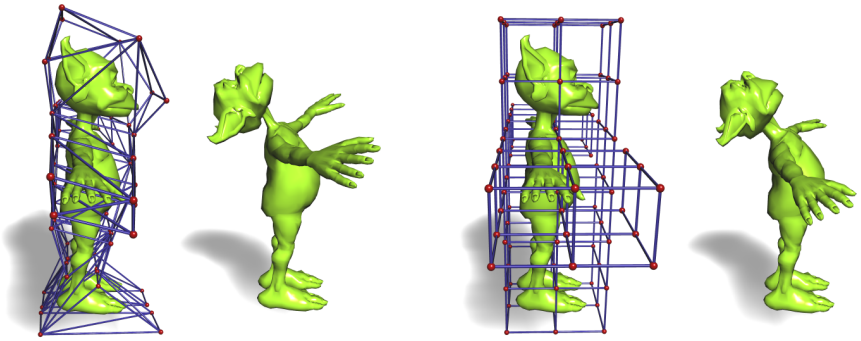


Figure 4.16: *The goblin's head is pulled back using Dirichlet BCs. Note the preservation of the head's overall shape using our approach (left, 168 DOFs) compared to hexahedral FEM (right, 297 DOFs).*

dynamics instead of the static problem allows to model animation sequences including secondary motions due to inertia.

Timings. For the examples shown, Table 4.2 summarizes timing information, taken on an Intel Core2 Duo, 2.4 GHz. Note that the computation time is approximately linear in the number of quadrature points and quadratic in the number of DOFs due to the dense stiffness matrices.

4.5 Discussion and Outlook

In this chapter, we have seen that switching from simple basis functions (and induced solution spaces) to more general shapes can improve the applicability and handling of FEM methods considerably. We presented approaches to simplify the actual remeshing task using polyhedral elements and to abandon remeshing completely by switching to a meshfree representations. Further, we also presented a cage-based approach to enable small-scale feature preservation.

Scene	# DOFs	# Q	t_{init}	t_{step}
Animation (Fig. 4.15)	168	200	3132	529
Goblin (Fig. 4.16)	168	200	3236	503
Dragon (Fig. 4.13)	156	300	4615	653
Armadillo (Fig. 4.17)	159	300	4451	784

Table 4.2: *Statistics and timings for the examples shown. We list number of DOFs, quadrature points, time to set up all $\mathbf{K}_{ij}^q$ and $\mathbf{H}_{ij}^q$ and the time to perform one timestep, in milliseconds.*

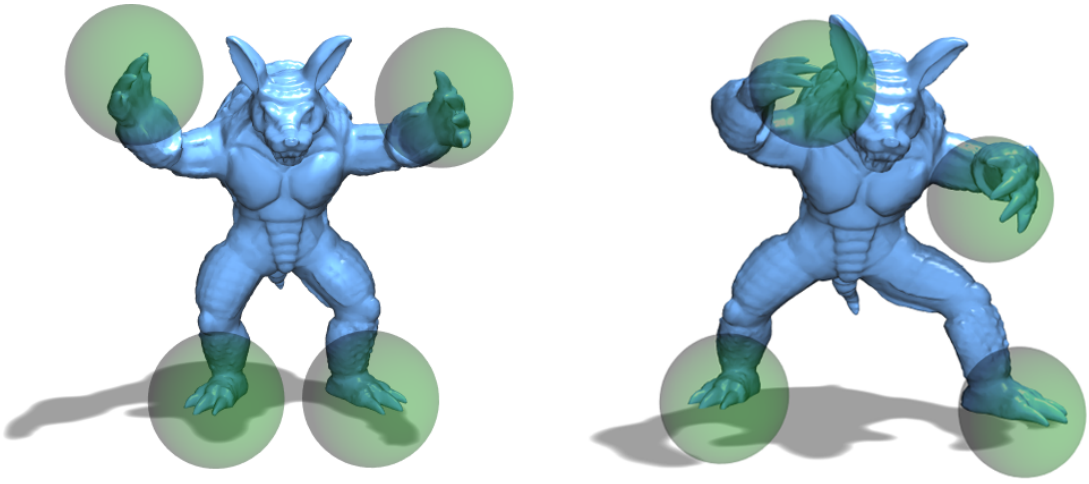


Figure 4.17: *By setting boundary conditions on few vertices and using the static solution, our method can also be used as a modeling tool.*

Polyhedral Elements. We introduced arbitrary polyhedral elements based on harmonic basis functions, and proposed the method of fundamental solutions as a simple and flexible method for computing these basis functions. Being able to use general polyhedral elements in FEM simulations considerably simplifies topological changes of the simulation domain, as illustrated for adaptive mesh generation, dynamic refinement, and progressive cutting. Extending our approach to both adaptive and hierarchical discretizations and solvers has further potential to improve runtime performance of the simulation.

Meshfree Representation. Building up on the element-free Galerkin method [Belytschko et al., 1994], we presented a corotated extension that is simple to implement and, due to the flexible point sampling, allows to continuously trade efficiency with a collocation approach against accuracy with a more accurately integrated Bubnov-Galerkin scheme. As we have shown, the presented formulation is only suitable for thick geometries and restricts the discretization points to be arranged volumetrically. In the next chapter, we will build up on this attractive meshfree approach and tackle the problems of thin geometries and inherent sampling requirements. Moreover, we will also focus on taking advantage of the simple discretization scheme by presenting two approaches to handle large plastic flows as well as the cutting of material.

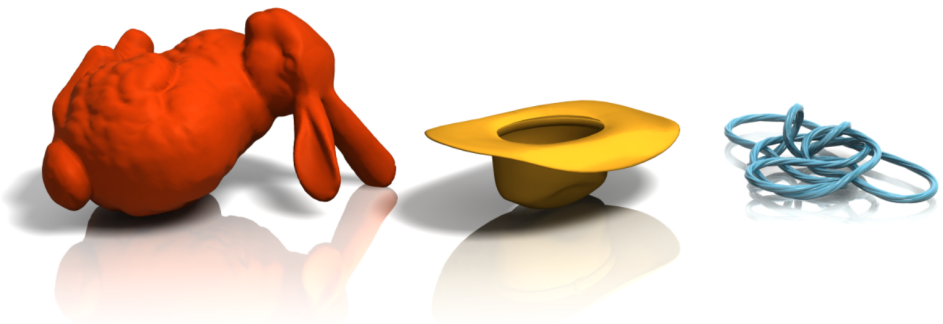
Feature Preservation. Simulations resulting from this approach are necessarily different from simulations performed with classical FEM since the

deformations are tied to a different subspace. Contrary to the approximation spaces chosen in classical FEM, the subspace implied by GC does not allow for convergence under refinement since analytic solutions to elasticity are not shape preserving. Nevertheless, with respect to graphics requirements, the presented method is able to give novel and visually pleasing results. Due to the globally supported basis functions, both stiffness matrices are dense, restricting the total number of DOFs as well as the runtime performance since the dense matrix assembly needs to be performed in each timestep (see Algorithm 1). While our approach works with single cages, extensions to multiple cages (i.e. multiple elements) are thinkable, leading to GC-based FEM with sparse matrices and improved performance. Assuming the normals to be constant in each timestep introduces damping in the angular momentum, especially for large timesteps, restricting us from taking large steps in the simulation. We expect that taking higher order Taylor approximations should be able to reduce this artifact.

When considering the presented approaches in the light of the chosen discretization structures, we observe that the line between classic tetra- or hexahedral FEM approaches and their meshless counterparts becomes blurred: Introducing arbitrary polyhedral element shapes leads to very flexible basis function shapes that can easily adapt to various geometric circumstances, reducing the amount of meshing to represent a given discretization domain. The step from these basis functions to completely meshless discretizations without any meshing requirement is then a rather small one, even if different data interpolation schemes are used. Furthermore, while we demonstrated these different basis function types in the context of corotated linear elasticity, it is important to note that they can as well be used with nonlinear strain measures and nonlinear material behavior, which therefore also constitutes an interesting direction for future work.

CHAPTER 5

Unifying Resultant-based Models



While the last chapter focused on generalizing classic approaches for solid simulations, this chapter we will focus on enlarging the range of possible object geometries. In order to allow the simulation of fine structures that cannot be captured by classic solid approaches, we are going to investigate into *resultant-based models* — thin shell and rod models — that are specialized for such characteristic geometries.

In contrast to these specific resultant-based methods being only valid for single types of geometry (shell or rod), we however aim at developing an approach being able to handle all three types in an *unified* manner. The advan-

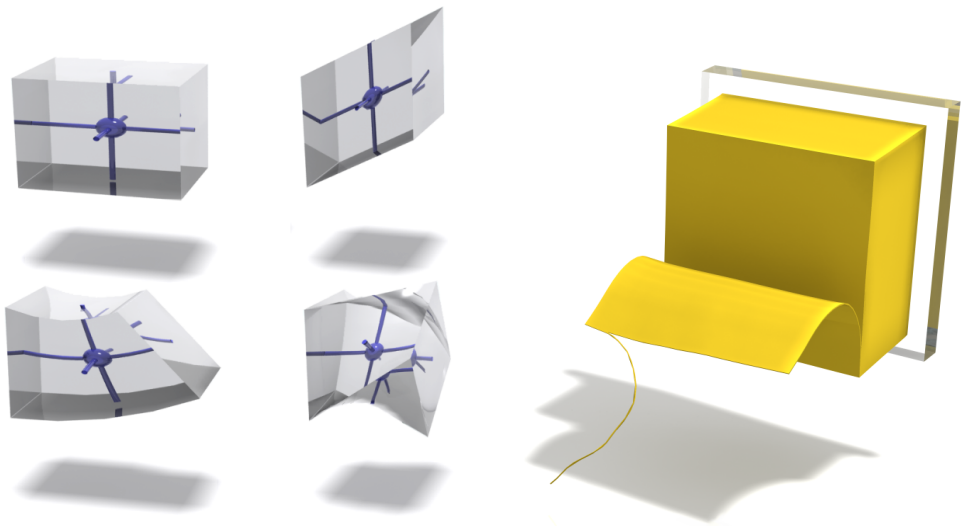


Figure 5.1: *An elaston measures stretching, shearing, bending, and twisting along any axis. An assembly of elastons accurately captures the behavior of elastic materials of any dimension, manifold or not, such as this rod cut out of a shell cut out of a cube.*

tages of such an approach are significant. While the coding task is simplified considerably (a single code instead of three different simulation codes plus coupling), also the spatial transition between non-manifold geometries or the temporal transition occurring during topological changes or elasto-plastic deformation can all be captured by a single, unified approach. With previous methods, these would be formidable tasks, but, as we will show, it is straightforward with the concept of *elastons* that we will present.

5.1 Overview

We achieve the goal of unifying the three distinct modeling approaches by deriving a simple quadrature rule for volumetric deformation fields that stably and accurately resolves the stored deformation energy regardless of the (local) form. By evaluating this quadrature rule at points we call *elastons*, we obtain elastoplastic forces acting on the simulation degrees of freedom.

The concept of elastons is independent of the representation of the volumetric deformation field, or choice of DOFs. In the light of the meshless nature of the elastons and the discretization ease we have seen in Section 4.3 by using MLS basis functions, we however choose a suitable extension: *generalized moving least squares* (GMLS) — a meshless generalization of Hermite interpolation

to three dimensions that also allows unproblematic sampling of, possibly co-linear, reduced geometries.

In order to further extend the range of different materials and effects, we also present approaches to include plastic deformations as well as topological changes into the elaston-based framework. To achieve this we apply the additive plasticity model of O'Brien et al. [1999] as well as a resampling technique for the discretization structures to faithfully handle the resulting large deformations. To handle the topological changes occur for example during cutting or fracturing, an extension of the virtual node algorithm [Molino et al., 2004] to meshless discretizations is presented.

5.2 Volumetric Resultant-Based Models

In order to understand and set up the necessary requirements we want to impose on a unified approach, let us first revisit the main steps that lead to the classic Kirchhoff theories for shells and rods in Section 3.2.

The derivation of Kirchhoff-Love shells and Kirchhoff rods in Section 3.2.1 and Section 3.2.2, respectively, followed basically the following steps: Using a suitable curvilinear description of the shell or rod material we were able to give a linearized description of the strain field by just using first and second order deformation information on the *reduced* geometry (midsurface or centerline) of a given object. Using this linearized description then allowed us to reduce the volumetric elastic potential to a surface- or curve-based description of the deformation energy. Typical resultant-based formulations would then invoke the Kirchhoff assumption and impose orthogonality of normals during deformation, leading to vanishing normal components of the strain. Although not our primary concern at this point, we note this assumption is by nature restrictive, prohibiting the tangential shearing, being an effect visually important for thicker materials.

While the transition from a volumetric to a purely reduced geometry-based representation would be advantageous for an exclusive treatment of shells or rods, it is also the primary source of aches in the effort to consistently unite specialized models: combining different dimensionalities of the solution representation is a difficult task in general. By following the classical resultant-based derivation — but only halfway, i.e., without invoking the Kirchhoff assumption — we gain the notions of measuring stretching separately from bending (and twisting), while keeping the displacement field *volumetric*. We estimate this as a key requirement for formulating an unified method. By keeping the solution description volumetric in all regimes, the

representation becomes trivial and no combining and coupling of different discrete descriptions is necessary.

However, this also requires that the resultant-based energy formulation not only measures deformation of the reduced geometry itself but also along the normal direction(s). If this is not the case, additional (geometric) constraints need to be introduced to give a well-defined volumetric solution. Otherwise, problems arise due to vanishing zero-energy modes.

So far, we have seen the stored deformation energy of solids, shells, and rods deforming under the volumetric displacement field $\mathbf{u}$. Since the energy expressions (3.4), (3.20), and (3.23) are distinct, they do not "agree" on a single unified implementation. Hypothetically, we could try to classify each region of the material domain as solid, shell, or rod. This is both complicated and unlikely to be effective: manual classification would fail for materials that drastically deform under cutting or plastic deformation; automatic classification is not a well-posed problem, e.g., not all shapes can be divided into pieces that are unambiguously solids, shells, or rods. Perhaps more fundamental is the observation that because classifications are discrete decisions, a change in classification (e.g., due to cutting or extreme deformation) could result in a jump in elastic forces and consequently popping artifacts. These reasons lead to the second requirement that also energy measurement should take place in a unified, regime-independent fashion. We avoid classification altogether by following the pattern of derivations of solids (zero normal directions), shells (one normal), and rods (two normals) to its logical conclusion, *elastons* (three normals).

5.3 Elastons

Consider a volumetric point-like solid whose extent along *all three* directions is small. This time the reduced geometry is a point, or elaston, and our notion of strain in the vicinity of this point will measure the linear deformations of stretch and shear at the center, as well as the quadratic deformations of bending and twist along all three "normal" directions. Figure 5.1, left, depicts these four essential deformation modes.

Linearizing Strain. Analogous as in Section 3.2, we employ again curvilinear coordinates and describe an elaston centered at $\boldsymbol{\theta}_0 = (0,0,0)$. We perform a first-order Taylor approximation of positions and displacements, this time

in all *three* normal directions θ_1, θ_2, θ_3:

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

Substituting into (3.2) yields the strain centered about the elaston

$$\boldsymbol{\epsilon}(\boldsymbol{\theta}) \approx \boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \sum_{k=1}^{3} \theta_k \boldsymbol{\beta}^k(\boldsymbol{\theta}_0)\,. \tag{5.1}$$

Observe that this expression naturally generalizes its shell (3.17) and rod (3.22) analogues; in particular, it captures stretching, shearing, bending, and twisting along all three axes.

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

Correspondingly, we compute the elaston's energy by substituting (5.1) into (3.4) to obtain an integral over the elaston's volume Ω_e,

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

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

$$W = \frac{V}{2}\Big(\boldsymbol{\alpha}(\boldsymbol{\theta}_0) : \mathbf{C} : \boldsymbol{\alpha}(\boldsymbol{\theta}_0) + \sum_{k=1}^{3} \frac{h_k^2}{12}\boldsymbol{\beta}^k(\boldsymbol{\theta}_0) : \mathbf{C} : \boldsymbol{\beta}^k(\boldsymbol{\theta}_0)\Big)\,. \tag{5.2}$$

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

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

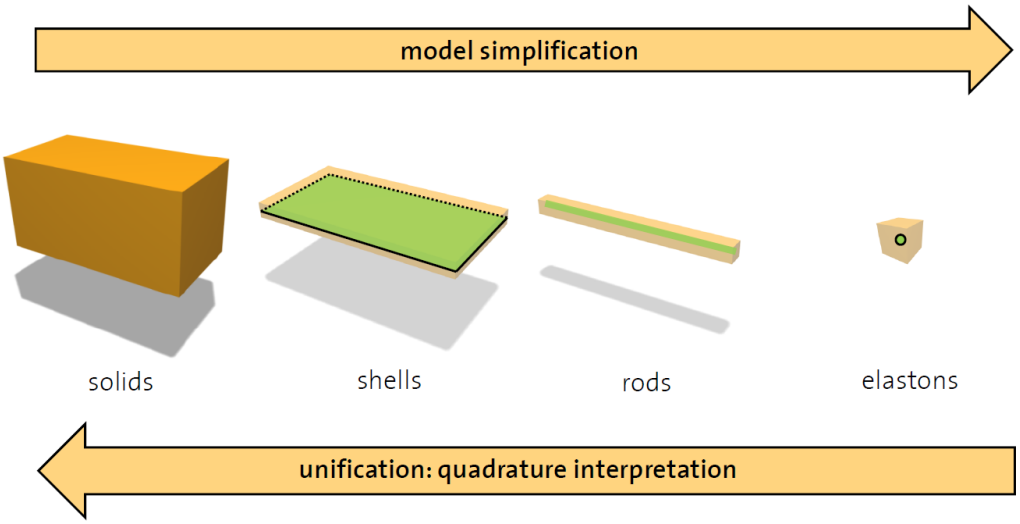


Figure 5.2: *For thin materials (transparent orange) resultant-based models (green) deliver good descriptions of their mechanics. Alternatively, these models can also be used to describe materials of "higher dimensionality" (on their left). Ultimately, the elaston is able to describe rods, shells and solids.*

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

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

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

Formally, a set of elastons $e \in \mathcal{E}$ form an integration rule approximating the elastic energy (3.4) as

$$W = \sum_{e \in \mathcal{E}} \frac{V^e}{2} \left(\boldsymbol{\alpha}^e : \mathbf{C} : \boldsymbol{\alpha}^e + \sum_{k=1}^{3} \frac{(h_k^e)^2}{12} \boldsymbol{\beta}^{ke} : \mathbf{C} : \boldsymbol{\beta}^{ke} \right), \tag{5.3}$$

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

- The elastons have to sample the material Ω sufficiently densely, such that all relevant deformations are measured and no undesired modes can appear because they would not be captured by the integration rule. A dense sampling avoids these problems and at the same time guarantees an accurate integration of the elastic energy (see Section 5.5).
- We also have requirements for an admissible basis of the solution space, as seen in the last section. Since we are now dealing with second order quantities, the basis functions must now be twice differentiable in order to be able to measure bending strains. As before, they need to reproduce constant and linear functions for accurate preservation of linear and angular momenta.

Let us first discuss the second point.

5.4 Generalized MLS

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

Motivation for GMLS. If we recall the MLS-based discretization in Section 4.3, we note an important limitation of the classical MLS shape functions $N_i(\bar{\mathbf{x}})$: To guarantee an invertible matrix $\mathbf{G}(\bar{\mathbf{x}})$, there have to be sufficiently many samples $\bar{\mathbf{x}}_i$ supporting the point of evaluation $\bar{\mathbf{x}}$ (determined by $w(\bar{\mathbf{x}} - \bar{\mathbf{x}}_i)$), and *these samples* $\bar{\mathbf{x}}_i$ *must not be coplanar.*

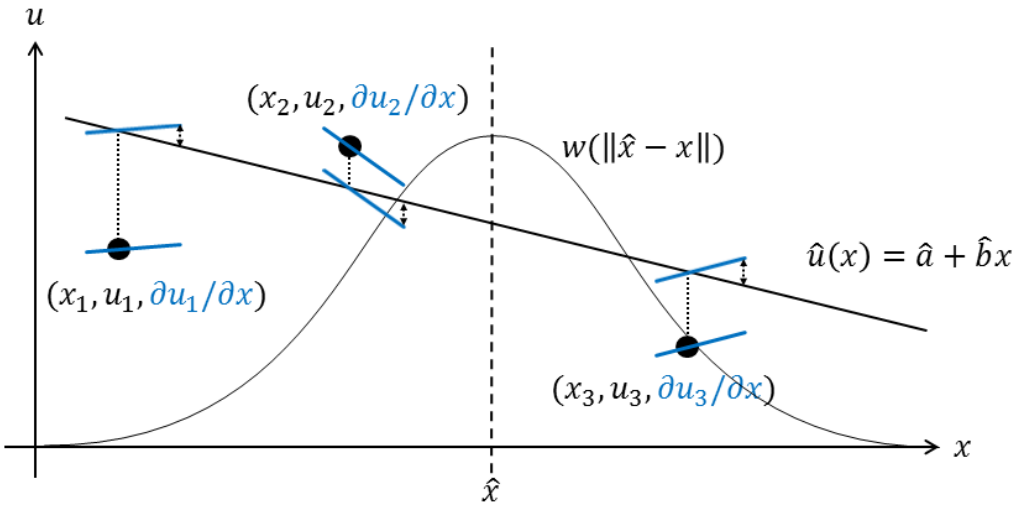


Figure 5.3: *A simple example of a linear 1D GMLS fit: For position $\hat{x}$, a linear polynomial $\hat{a}x + \hat{b}$ is fitted to the three data points (x_1, u_1), (x_2, u_2) and (x_3, u_3) by minimizing the weighted sum of value (black dots) and derivative (blue lines) errors: $\sum_1^3([(\hat{a}x_i + \hat{b}) - u_i)^2 + (\hat{a} - \frac{\partial u_i}{\partial x})^2]w(|\hat{x} - x_i|)$.*

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

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

Linear GMLS. In addition to displacement DOF $\mathbf{u}_i$, as we had for linear MLS in Section 4.3.1, each sample point $\bar{\mathbf{x}}_i$ is now also associated with derivative DOFs $\mathbf{u}_{i,j} \in \mathbb{R}^3$ $(1 \leq j \leq 3)$. The coefficients $\mathbf{a}(\bar{\mathbf{x}})$ are determined by fitting the polynomial $\mathbf{a}^T\mathbf{p}(\bar{\mathbf{x}})$ not only to the value DOF, but now also to the additional derivative DOF (as depicted in Fig. 5.3). Formally, this simply amounts to

add the error term

$$\sum_{i=1}^{n}\sum_{j=1}^{3} w(\bar{\mathbf{x}}-\bar{\mathbf{x}}_i)\left\|\mathbf{a}^T\mathbf{p}_{,j}(\bar{\mathbf{x}}_i)-\mathbf{u}_{i,j}\right\|^2$$

to $J(\mathbf{a})$ in (4.11). Minimizing the resulting H^p-like norm leads to

$$\mathbf{u}(\bar{\mathbf{x}}) = \sum_{i=1}^{n}\left[\mathbf{u}_i N_i(\bar{\mathbf{x}}) + \sum_{j=1}^{3}\mathbf{u}_{i,j}N_i^j(\bar{\mathbf{x}})\right], \tag{5.4}$$

with additional basis functions for derivative information

$$N_i^j(\bar{\mathbf{x}}) \;=\; \mathbf{p}(\bar{\mathbf{x}})^T\,\mathbf{G}^{-1}(\bar{\mathbf{x}})\,\mathbf{p}_{,j}(\bar{\mathbf{x}}_i)\,w(\bar{\mathbf{x}}-\bar{\mathbf{x}}_i) \tag{5.5}$$

and an augmented, generalized matrix $\mathbf{G}$ used in the construction of $N_{i,j}(\bar{\mathbf{x}})$ in (5.5) *and* $N_i(\bar{\mathbf{x}})$ in (4.13):

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

Compared to (4.13), the outer products of monomial derivatives $\mathbf{p}_{,j}$ guarantee a regular matrix $\mathbf{G}$ in any case—for coplanar and colinear samplings, even for a single sample point. This independence of the sampling makes GMLS the ideal choice for discretizing deformation fields of solids, shells, and rods, where each point $\bar{\mathbf{x}}_i$ now has the 12 DOFs $\mathbf{u}_i$, $\mathbf{u}_{i,1}$, $\mathbf{u}_{i,2}$, $\mathbf{u}_{i,3} \in \mathbb{R}^3$.

Quadratic GMLS. If higher accuracy and faster convergence are desired, we can alternatively consider second order derivative DOFs $\mathbf{u}_{i,jk}$ $(1 \le j,k \le 3)$ and additionally use a quadratic polynomial $\mathbf{p}(\bar{\mathbf{x}}) = (1,\bar{x},\bar{y},\bar{z},\bar{x}\bar{x},\bar{x}\bar{y},\bar{x}\bar{z},\bar{y}\bar{y},\bar{y}\bar{z},\bar{z}\bar{z})^T$, providing 30 DOFs per sample point $\bar{\mathbf{x}}_i$. Analogously, this adds a third error term

$$\sum_{i=1}^{n}\sum_{j,k=1}^{3} w(\bar{\mathbf{x}}-\bar{\mathbf{x}}_i)\left\|\mathbf{a}^T\mathbf{p}_{,jk}(\bar{\mathbf{x}}_i)-\mathbf{u}_{i,jk}\right\|^2$$

to $J(\mathbf{a})$ and leads to

$$\mathbf{u}(\bar{\mathbf{x}}) = \sum_{i=1}^{n}\left[\mathbf{u}_i N_i(\bar{\mathbf{x}}) + \sum_{j=1}^{3}\mathbf{u}_{i,j}N_i^j(\bar{\mathbf{x}}) + \sum_{j,k=1}^{3}\mathbf{u}_{i,jk}N_i^{jk}(\bar{\mathbf{x}})\right], \tag{5.6}$$

with additional second-order derivative shape functions

$$N_i^{jk}(\bar{\mathbf{x}}) \;=\; \mathbf{p}(\bar{\mathbf{x}})^T\,\mathbf{G}^{-1}(\bar{\mathbf{x}})\,\mathbf{p}_{,jk}(\bar{\mathbf{x}}_i)\,w(\bar{\mathbf{x}}-\bar{\mathbf{x}}_i) \tag{5.7}$$

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

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

In our simulation framework we integrated both first and second order GMLS. While the first order scheme allows for faster simulations (12 DOFs/sample), the second order discretization (30 DOFs/sample) provides more accurate results and converges to the physically correct solution (see Section 5.7). The first and second order methods have the same structure, so that a single code can offer an easy trade-off between speed and accuracy.

5.5 Implementation

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

Input: Point-based material representation $\mathcal{M}$

Precomputation
- Generate GMLS points $\mathbf{x}_i$ by sampling $\mathcal{M}$ (Sec. 5.5.1)
- Generate elastons e by sampling $\mathcal{M}$ (Sec. 5.5.1)
- Compute elaston stiffness matrices $\mathbf{K}^e$ (Sec. 5.5.2)
- Compute mass matrix $\mathbf{M}$ (Sec. 5.5.2)

Simulation loop
- Assemble global stiffness matrix $\mathbf{K}$ (Sec. 5.5.2)
- Boundary conditions assembly (Sec. 5.5.3)
- Collision detection and handling (Sec. 5.5.3)
- Time integration (Sec. 5.5.3)

Algorithm 2: The individual steps of our simulation, divided in precomputation and per-frame computations.

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

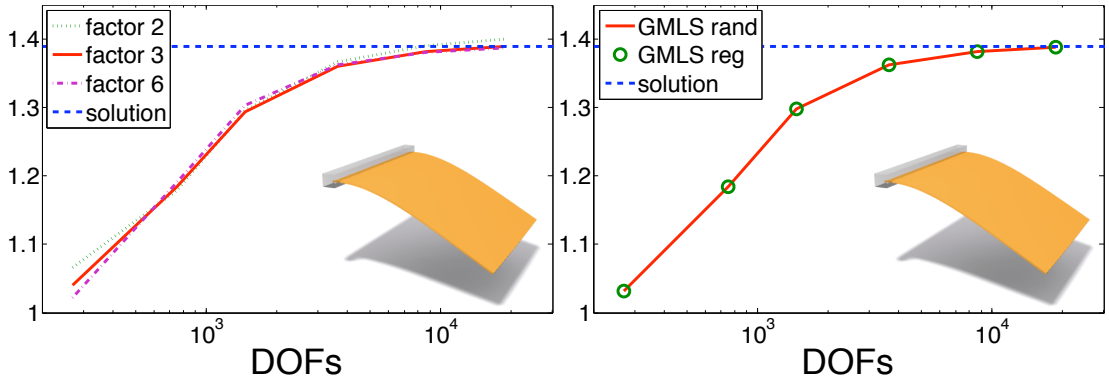


Figure 5.4: *Left: Convergence behavior for a clamped shell under gravity, for several ratios of elaston density to GMLS sample density. Moving from a double to triple ratio improves convergence, but higher ratios do not lead to further improvements. Right: Randomly rotating the elastons' tangent axes around the normal vector does not change the result; the two curves are coincident. Both plots show the displacement at the rightmost point vs. DOFs. The reference solution is computed using a high-resolution Kirchhoff-Love shell.*

5.5.1 Sampling

Given an input cloud, we generate the positions $\{\bar{\mathbf{x}}_1, \ldots \bar{\mathbf{x}}_n\}$ of GMLS sample points and elaston centers $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$ by subsampling the dense material point set $\mathcal{M}$. For DOF positions $\bar{\mathbf{x}}_i$, we use farthest point sampling of the material $\mathcal{M}$, similar to Adams et al. [2008]. Starting from $\bar{\mathbf{x}}_1 = \mathbf{m}_1$, subsequent samples $\bar{\mathbf{x}}_{i+1}$ are picked in a greedy manner to maximize the distance to the points $\{\bar{\mathbf{x}}_1, \ldots, \bar{\mathbf{x}}_i\}$ already selected. This sampling strategy results in a uniform distribution, but favors samples at the boundary of the material. In our context, however, samples should ideally be located at the mid-surface or centerline.

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

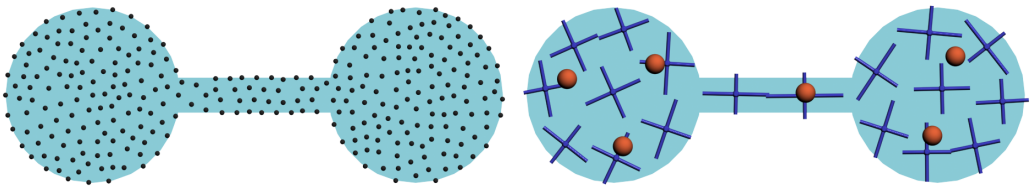


Figure 5.5: *A simple 2D example of our sampling algorithm. Starting from material point representation (left) our algorithm computes DOF locations (right,red dots) and elastons (right, blue crosses).*

Simulation accuracy depends on the ratio of densities of elastons $\mathbf{e}_i$ and of GMLS samples $\bar{\mathbf{x}}_i$ (see Figure 5.4). To capture all relevant deformation modes, elastons should be sampled at 2–3 times the axial density of $\bar{\mathbf{x}}_i$. We begin with elaston positions $\mathbf{e}_i = \bar{\mathbf{x}}_i$ $(1 \leq i \leq n)$ and add more elastons $\{\mathbf{e}_{n+1}, \ldots, \mathbf{e}_m\}$ by farthest point sampling, followed by Lloyd relaxation to improve the sampling.

For theoretical completeness, our derivation of elastons in Section 5.3 assumed curvilinear coordinates $\bar{\mathbf{x}}(\boldsymbol{\theta})$ for strain computations. In implementation we find it simpler to construct a per-elaston local flat parameterization, so that each undeformed elaston is a small cuboid. Given the elaston center $\mathbf{e}_i$ and Voronoi region $\mathcal{M}_i$, covariance analysis of the spheres $B(\mathbf{m}_i, r_i)$ yields an orthogonal local frame and eigenvalues $\{\lambda_j\}$. These axes serve as the undeformed first derivatives $\bar{\mathbf{x}}_{,j}$ ("tangent vectors") for the computation of the strains in (3.18) and (3.19). Note that rotating two tangents of equal length around their common normal does not change the simulation accuracy (Fig. 5.4,right).

Fig. 5.5 shows material points $\mathbf{m}_i$ (left), GMLS samples $\bar{\mathbf{x}}_i$ (right, red), and elastons $\mathbf{e}_i$ with their local frames (right, blue). The relative lengths of the cuboid sides are given by $\sqrt{\lambda_1} : \sqrt{\lambda_2} : \sqrt{\lambda_3}$ and their absolute value is chosen such that the elaston's volume matches that of the Voronoi region $\mathcal{M}_i$, i.e., the exact volume to be approximated by the elaston, ensuring exact representation of total mass independent of the sampling pattern.

As a consequence of our assumption of a locally flat parameterization the second derivatives $\bar{\mathbf{x}}_{,jk}$ used in the computation of the bending strain (3.19) vanish. The error introduced by this depends on the curvature of the undeformed state, and therefore can be reduced by a curvature-adaptive sampling of elastons, i.e., by adjusting the grading field of the Lloyd clustering. Our experiments have shown that the remaining errors of our discrete elaston integration scheme are insignificant compared to the model discretization error (see Figs. 5.4, 5.11). Furthermore, in the limit case the energy of a sin-

gle elaston approaches the exact continuous energy density at that point, guaranteeing vanishing integration error under refinement.

5.5.2 System Matrices

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

Stiffness Matrix. To integrate membrane and bending strains (see (3.18) and (3.19)), we need first and second order derivatives $\bar{\mathbf{x}}_{,j}$, $\bar{\mathbf{x}}_{,jk}$ and $\mathbf{u}_{,j}$, $\mathbf{u}_{,jk}$. By construction of our local elaston parameterization, $\bar{\mathbf{x}}_{,j}$ are the elaston's axes and $\bar{\mathbf{x}}_{,jk}$ vanish. Writing the deformation field as $\mathbf{u}(\bar{\mathbf{x}}(\boldsymbol{\theta}))$, we have the first derivatives

$$\mathbf{u}_{,i} \;=\; \frac{\partial \mathbf{u}(\bar{\mathbf{x}}(\boldsymbol{\theta}))}{\partial \theta_i} \;=\; \nabla \mathbf{u}\, \frac{\partial \bar{\mathbf{x}}(\boldsymbol{\theta})}{\partial \theta_i} \;=\; \nabla \mathbf{u}\, \bar{\mathbf{x}}_{,i}\,, \tag{5.8}$$

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

$$\mathbf{u}_{,ij} \;=\; \bar{\mathbf{x}}_{,i} \cdot \mathbf{H}_{\mathbf{u}}\, \bar{\mathbf{x}}_{,j}\,, \tag{5.9}$$

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

We perform a classic Galerkin discretization and replace the continuous solution $\mathbf{u}(\bar{\mathbf{x}})$ by our GMLS approximation $\sum_i \mathbf{u}_i\, N_i(\bar{\mathbf{x}})$ of Section 5.4. For each elaston $e \in \mathcal{E}$, this leads to a local stiffness matrix $\mathbf{K}^e$, which we construct using Voigt notation [Hughes, 2000]. We identify all basis functions N_i that have support at the elaston's position; for each of these basis functions we compute 6×3 matrices $\mathbf{A}_i$ and $\mathbf{B}_i^k$ corresponding to the membrane strain $\boldsymbol{\alpha}$

and the bending strains $\boldsymbol{\beta}^k$ ($k = 1,2,3$), given in terms of their rows

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

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

Using the 6×6 constitutive tensor $\mathbf{C}$ for linear material response and considering (5.2), we compute 3×3 blocks $\mathbf{K}_{ij}^e$ of the elaston's stiffness matrix $\mathbf{K}^e$ as

$$\mathbf{K}_{ij}^e = V^e \left[\mathbf{A}_i^T \mathbf{C} \mathbf{A}_j + \sum_{k=1}^{3} \frac{\left(h_k^e\right)^2}{12} \mathbf{B}_i^{k\,T} \mathbf{C} \mathbf{B}_j^k\right]. \tag{5.10}$$

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

Corotation. As mentioned earlier, we employ linear strain measures and constitutive relations, such that the Hessian of the energy becomes constant, which enables fast and stable simulations by means of corotation [Müller et al., 2002; Hauth and Strasser, 2004]. As seen in Sections 3.3.3 and 4.3.2, for meshless simulations we can apply corotation per quadrature point, i.e., per elaston in our case.

We start by estimating the local rotation matrix $\mathbf{R}^e$ at each elaston again by polar decomposition of the deformation gradient $(\mathbf{I} + \nabla \mathbf{u})$ and replace the blocks $\mathbf{K}_{ij}^e$ by their rotated versions $\mathbf{R}^e \mathbf{K}_{ij}^e \mathbf{R}^{e\,T}$. The global stiffness matrix $\mathbf{K}$ can then be assembled from the rotated elaston stiffness matrices $\mathbf{K}^e$. Moreover, the vector $\mathbf{R}^e \mathbf{K}_{ij}^e (\mathbf{I} - \mathbf{R}^{e\,T}) \bar{\mathbf{x}}_j^0$ has to be added to the i-th vector component of the external force $\mathbf{f}$. The $\bar{\mathbf{x}}_j^0$ denote the coefficients of the basis functions N_j when representing the undeformed configuration. They are equal to DOF locations for positional DOFs, equal to the unit vectors $\mathbf{e}_k$ for the first derivative DOFs in direction $\mathbf{e}_k$, and vanish for second order derivative DOFs.

Mass Matrix. The mass matrix can also be precomputed, since it does not change as long as the undeformed shape of the material remains unchanged. Unlike the computation procedures described in the last two chapters, a

slightly different approach has to be taken here. First, because we have non-nodal basis functions we avoid simplifications such as mass lumping and instead compute the mass matrix classically by assembling 3×3 blocks

$$\mathbf{M}_{ij} \;=\; \mathbf{I} \cdot \int_{\Omega} \rho(\bar{\mathbf{x}})\, N_i(\bar{\mathbf{x}})\, N_j(\bar{\mathbf{x}})\, \mathrm{d}\Omega\,. \tag{5.11}$$

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

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

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

Computing the integral in (5.11) as a sum of elaston-like integrals of the linearized shape functions—which can again be evaluated analytically—results in the following approximation:

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

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

5.5.3 Implementation Details

Time Integration. As in the previous chapter, we also employ simple semi-implicit Euler integration to advance the simulation in time (see Section 3.3.5). The linear systems resulting from the discussed discretizations are sparse, symmetric, and positive definite, which makes them ideally suited for efficient linear solvers.

Boundary Conditions. As we have seen in Section 3.3.2, prescribing Dirichlet constraints is more demanding for simulations employing non-nodal basis functions (i.e., $\mathbf{u}(\bar{\mathbf{x}}_i) \neq \mathbf{u}_i$) such as GMLS. We therefore employ the penalty-based approach described in Section 3.3.2 since it is simple to implement, gives satisfactory results, and does not introduce additional DOFs into the system.

Collisions. We also perform collision detection and handling in a point-based manner. Since the object is represented by a dense set of spheres $B(\mathbf{m}_j, r_j)$, we detect collisions between material spheres $B(\mathbf{m}_j + \mathbf{u}(\mathbf{m}_j), r_j)$ in the deformed configuration. We also detect collisions with analytically defined objects such as planes and cylinders. We respond to collisions again using penalty forces as described in Section 3.3.2.

5.6 Extensions

Objects do not only transition between solid, shell, and rod in space, regime changes can also develop over time. In plastic and viscous deformations, for instance, material can be stretched into thin sheets or strands, whose elastic behavior can be correctly captured by our approach. The topological changes induced by cutting and fracturing can also generate material with thin features that should be captured by the simulation system. In this section we therefore extend the basic elaston model to support plastic flow and topological changes.

5.6.1 Plasticity

In order to increase the applicability of the elaston approach we first adapt the additive plasticity model presented by O'Brien et al. [2002] to our method.

Plasticity Model

In a first step, we define plastic membrane and bending strain variables $\boldsymbol{\alpha}_p, \boldsymbol{\beta}_p^k$ for each elaston. By taking the difference between the measured geometric strains $\boldsymbol{\alpha}_g, \boldsymbol{\beta}_g^k$ (corresponding to Eq. (3.18) and Eq. (3.19)) and the stored plastic strains, we can define the effective elastic strain as

$$\boldsymbol{\alpha}_e = \boldsymbol{\alpha}_g - \boldsymbol{\alpha}_p, \quad \boldsymbol{\beta}_e^k = \boldsymbol{\beta}_g^k - \boldsymbol{\beta}_p^k, \tag{5.12}$$

Then the new elastic energy stored in a single elaston becomes

$$W_{\mathrm{e}} = \frac{V}{2}\Big(\boldsymbol{\alpha}_e : \mathbf{C} : \boldsymbol{\alpha}_e + \sum_{k=1}^{3} \frac{(h_k^e)^2}{12}\, \boldsymbol{\beta}_e^k : \mathbf{C} : \boldsymbol{\beta}_e^k\Big).$$

The conservative forces resulting from this energy then become

$$\frac{\partial W_{\mathrm{e}}}{\partial \mathbf{u}_i} = \mathbf{A}_i^T \mathbf{C}(\boldsymbol{\alpha}_g - \boldsymbol{\alpha}_p) + \sum_k (\mathbf{B}_i^k)^T \mathbf{C}(\boldsymbol{\beta}_g^k - \boldsymbol{\beta}_p^k),$$

i.e., additionally to the conventional elastic forces of the original approach, the gradient of the energy now also contains the elaston's "plastic force" contribution

$$\mathbf{f}_i^p = \mathbf{A}_i^T \mathbf{C}\boldsymbol{\alpha}_p + \sum_k (\mathbf{B}_i^k)^T \mathbf{C}\boldsymbol{\beta}_p^k.$$

This force can simply be incorporated into the right-hand side of the equation of motion Eq. (3.8). For updating the plastic strains we rely on the same criteria as proposed by O'Brien et al. [2002] and adapt it straightforwardly to our strain representation.

Large Deformations

While this simple model already allows for the simulation of plastic deformation to some extent, it is not suitable for larger deformations when combined with a linear strain measure. We therefore perform a periodic resampling step similar to the approach of Wojtan and Turk [2008] such that the plastic deformation is represented by the geometry and does not need to be carried along anymore by the plastic strain variables. Similar to Müller et al. [2004a], we do however not transfer the entire plastic strain into the rest state geometry which would remove any remaining elastic component, but perform the resampling by the three steps discussed next. Note that we will always use superscripts ^{n} for referring to quantities computed in the new rest configuration. We use $\bar{\mathbf{x}}$ to denote positions in the old rest state and $\mathbf{x}$ for the deformed configuration (being identical to the new rest state).

Step 1: Rest State Update

In order to get a continuous transition of the material position, without having to worry about popping artifacts, we take the current configuration as the new rest state, which will guarantee C^0-continuity of the solution across the resampling [Wojtan and Turk, 2008]. Material points, GMLS samples, and elastons can all three be generated from scratch for the new rest state

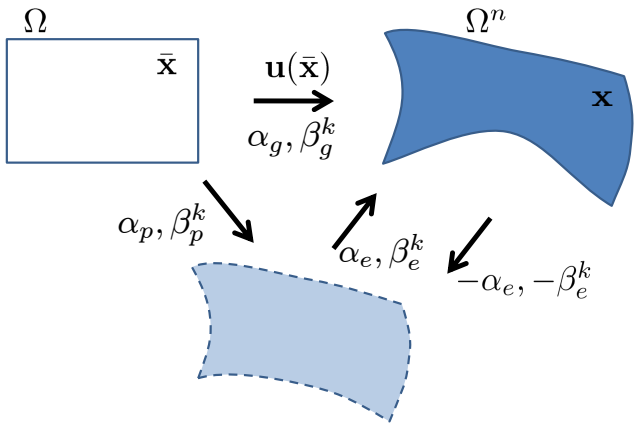


Figure 5.6: *Illustration of geometric, elastic and plastic strain, as well as the new plastic strain after resampling.*

as described in Section 5.5.1. Since the new rest state entirely describes the current positional state of the simulation, the new displacement field $\mathbf{u}$ can be set to zero.

Step 2: Velocity Field Update

In order to also have a smooth transition of the motion across the resampling, i.e., approximate C^1-continuity in time, we find the new velocity field $\mathbf{v}^n(\mathbf{x})$ by minimizing the least squares error between the new and the old velocity field, defined as

$$\int_{\Omega^n} \|\mathbf{v}^n(\mathbf{x}) - \mathbf{v}(\bar{\mathbf{x}}(\mathbf{x}))\|^2 d\Omega^n.$$

Note that these two fields are defined on separate domains. However, these domains are mapped one-to-one by $\mathbf{x}(\bar{\mathbf{x}}) = \bar{\mathbf{x}} + \mathbf{u}(\bar{\mathbf{x}})$. Since the two fields are represented in their corresponding bases as $\mathbf{v}^n(\mathbf{x}) = \sum_i \mathbf{v}_i^n N_i^n(\mathbf{x})$ and $\mathbf{v}(\bar{\mathbf{x}}) = \sum_i \mathbf{v}_i N_i(\bar{\mathbf{x}})$, we can solve the least squares problem by the following linear system

$$\mathbf{G}^n \mathbf{v}^n = \mathbf{G}\mathbf{v},$$

where $\mathbf{G}_{ij}^n = \int_\Omega N_i^n N_j^n$ is the Gram matrix of the new basis, and $\mathbf{G}_{ij} = \int_{\Omega^n} N_i^n N_j$ is the matrix transferring the old velocity field into the new configuration. In order to numerically evaluating these integrals in an accurate manner we do again take advantage of the higher-order integration scheme used for the elaston derivation and mass matrix computation and perform this integration up to second order. That it, we perform a quadratic local approximation around the integration point which we integrate analytically.

Step 3: Plasticity Update

The last remaining step is the transfer of the plastic variables from the old to the new rest state. In order to understand the reasoning for the following procedure, we quickly recapitulate the three states involved (depicted in Fig. 5.6). First, we have the undeformed configuration $\bar{\mathbf{x}}$ in which the discretization variables live. Second, we have a (virtual) physical rest state, which is described implicitly by the plastic strain variables $\boldsymbol{\alpha}_p, \boldsymbol{\beta}_p^k$ (dashed region), and third, we have the current configuration $\mathbf{x}$ described by the actual displacement field as $\mathbf{x} = \bar{\mathbf{x}} + \mathbf{u}(\bar{\mathbf{x}})$. During resampling, we update the undeformed configuration to the current state. The current configuration then coincides with the new undeformed configuration. However, the physical rest state (dashed region), which can change due to plastic deformation, should stay the same during the resampling process. We achieve this by defining the new plastic strain as the negative old elastic strain [Müller et al., 2004a].

Plasticity Transfer for Mapped Elastons. This idea is realized in two steps. First, the new plastic strain can be defined in the old rest configuration. This is done by evaluating the total membrane and bending strains contained in the deformation field $\mathbf{u}(\bar{\mathbf{x}})$. Note that since we are using a corotational approach, actually the rotated deformation field at the elaston's position is taken for the strain evaluation. Then (5.12) can just be used to define the new plastic strains in the undeformed configuration as $-(\boldsymbol{\alpha}_e + \sum_k \boldsymbol{\theta}_k \boldsymbol{\beta}_e^k)$. Note that this strain is still living in the old undeformed configuration. Given a direction $\mathbf{d}$ in the old configuration, the stretch in this direction can be measured as

$$-\mathbf{d}^T(\boldsymbol{\alpha}_e + \sum_k \boldsymbol{\theta}_k \boldsymbol{\beta}_e^k)\mathbf{d},$$

where $\boldsymbol{\theta}$ is the evaluation point in coordinates of the elaston's local frame.

In a second step, in order to use this updated plastic strain in the new rest state, it has to be transformed to the new configuration (Fig. 5.7). We assume the local frames of two elastons in the old and new rest state configurations are given as columns of the 3×3 matrices $\mathbf{T}$ and $\mathbf{T}^n$. Then we can use the deformation gradient $\mathbf{F} = \mathbf{I} + \nabla\mathbf{u}$ to define a matrix $\mathbf{M}$ that transforms a material direction $\mathbf{d}^n$ in the local frame of the deformed configuration to a direction $\mathbf{d}$ in the local frame of the undeformed configuration as

$$\mathbf{M} = \mathbf{T}^T\mathbf{F}^{-1}\mathbf{T}^n.$$

Using this map, we can compute stretch along $\mathbf{d}^n$ in the new rest state by mapping it back and evaluating it in the undeformed configuration using the

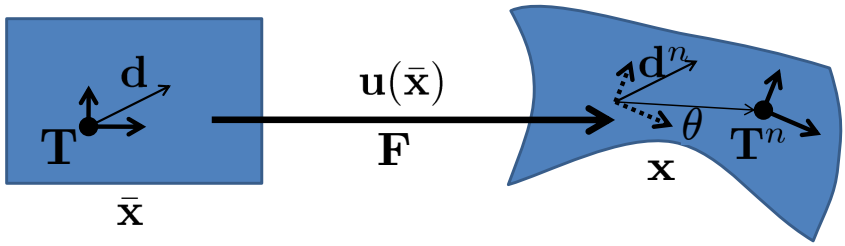


Figure 5.7: *Illustration of an elaston's local frame in the rest state, its mapped position in the new rest state, as well as the frame of a nearby elaston whose plastic strain is extrapolated during resampling.*

old strain. This gives us

$$(\mathbf{d}^n)^T(\boldsymbol{\alpha}_p^n + \sum_k \boldsymbol{\theta}_k \tilde{\boldsymbol{\beta}}_p^{kn})\mathbf{d}^n = -(\mathbf{d}^n)^T\mathbf{M}^T(\boldsymbol{\alpha}_e + \sum_k \boldsymbol{\theta}_k \boldsymbol{\beta}_e^k)\mathbf{M}\mathbf{d}^n,$$

leading to corresponding plastic strains in the new rest state as

$$\boldsymbol{\alpha}_p^n = -\mathbf{M}^T\boldsymbol{\alpha}_e\mathbf{M}$$

and

$$\tilde{\boldsymbol{\beta}}_p^{kn} = -\mathbf{M}^T\boldsymbol{\beta}_e\mathbf{M}.$$

While these two strains now allow for measuring stretch in directions defined in the new rest state, there are still two things missing.

First, the strain in the vicinity of the elaston's center is measured as $\boldsymbol{\alpha}_p^n + \sum_k \boldsymbol{\theta}_k \boldsymbol{\beta}_p^{kn'}$ where $\boldsymbol{\theta}$ describes the evaluation point in the elaston's local frame. However, in the definition given above, $\boldsymbol{\theta}$ is still given in the old rest state's local frame and needs to be transformed with the mapping $\mathbf{M}$ as well:

$$\boldsymbol{\theta} = \mathbf{M}\boldsymbol{\theta}^n.$$

Incorporating this transformation into the new bending strains gives then

$$\boldsymbol{\beta}_p^{kn} = \sum_j \mathbf{M}_{jk}\tilde{\boldsymbol{\beta}}_p^{jn},$$

such that the strain at a location $\boldsymbol{\theta}^n$ in the new rest state can finally be evaluated as

$$\boldsymbol{\alpha}_p^n + \sum_k \boldsymbol{\theta}_k \boldsymbol{\beta}_p^{kn}.$$

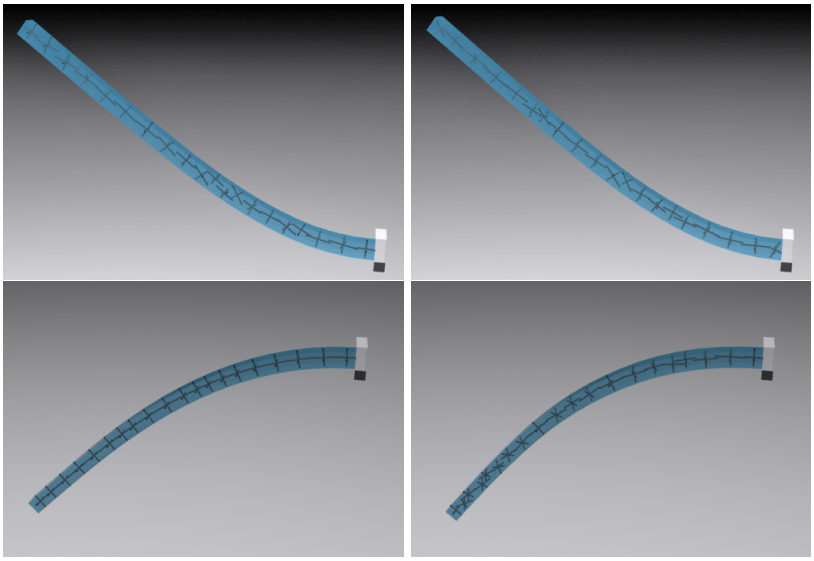


Figure 5.8: *Example of dynamic resampling of a swinging bar showing elaston configurations before (left) and after (right) resampling. Thanks to the smooth position and velocity interpolation, the simulation is free of any visible popping artifacts.*

Plasticity Transfer for Resampled Elastons. So far, we are able to transform the plastic strain from a position in the undeformed configuration to the corresponding position in the deformed configuration. However, we would like to sample the new rest state independently from the old sampling. We therefore first sample the new positions together with the new local frames. For each of these new elastons we then look up the closest transformed elaston and use its new coordinate frame for transferring the plastic strain to the transformed elaston.

As last step, we then just need to extrapolate these strains to the new elastons. We do this by defining a coordinate frame at the closest transformed elaston where the axes correspond to the local frame of the new elaston. In this frame, we can express the elaston's position as $\boldsymbol{\theta}$ and since we assume the strain to be a linear function in the vicinity of the transformed elaston, we can simply extrapolate the plastic membrane strain to the new position as

$$\boldsymbol{\alpha}_p^n + \sum_k \boldsymbol{\theta}_k \boldsymbol{\beta}_p^{kn}.$$

The bending strain stays the same. While different interpolation schemes could be chosen for interpolating the plastic strain to the new elaston locations, we did not observe any problem using this linear extrapolation approach. Fig. 5.8 shows an example of a fixed elastic bar which is resampled

during runtime using the proposed approach and which shows no visible popping artifacts.

5.6.2 Topological Changes

Cutting applications and effects of brittle materials that fracture under stress are further areas where thin structures can appear dynamically and which require a proper handling by the simulation method. For such simulations, we need to introduce a notion of *connectivity* between elastons.

We do this by first computing a connectivity graph between the densely sampled material spheres $\mathcal{M}$, where two spheres are connected if they overlap. The Lloyd relaxation performed in the elaston sampling step associates each candidate point with an elaston, which also implies a connectivity graph between elastons (see Fig. 5.9 for an example). Cutting can then be implemented by simply removing connections between two previously connected candidate points.

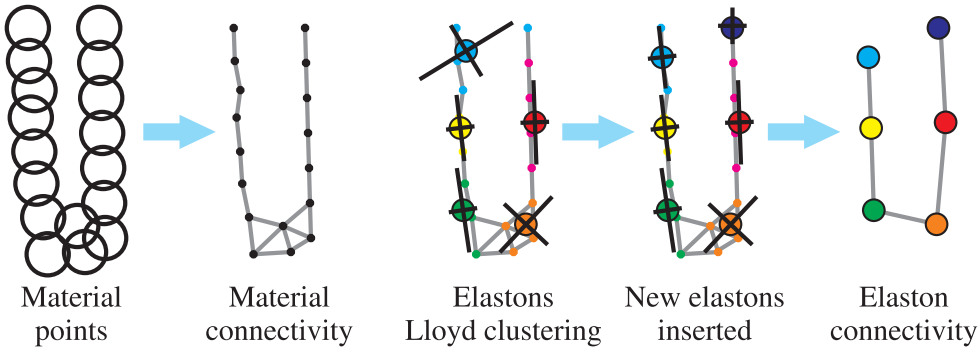


Figure 5.9: *Overlapping material spheres implicitly imply connectivity (left). Elastons containing disconnected material components are split up and elaston connectivity is computed (right).*

Before continuing the simulation, and after each change to the connectivity graphs, a meshless equivalent of the "virtual node algorithm" [Molino et al., 2004] is performed. This consists of two steps:

1. After the material point connectivity has been updated (according to a cut or fracture event), first new elastons are introduced to make sure the local connectivity graph of each candidate subset $\mathcal{M}_i$ is in itself connected.This can be performed by looking at the material subset $\mathcal{M}_i$ of an elaston and considering the connections between

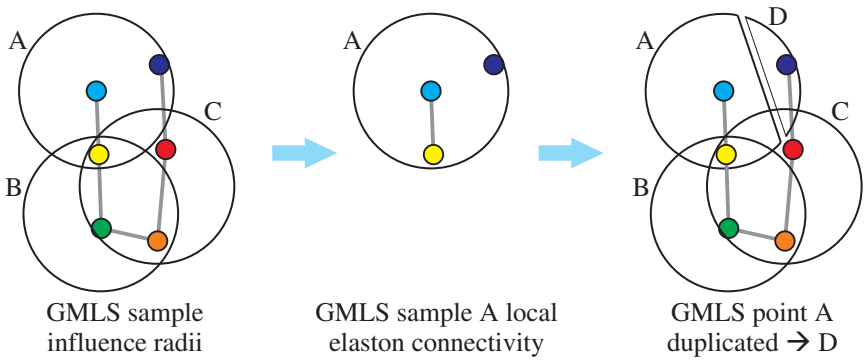


Figure 5.10: *DOFs containing disconnected elastons in their support are duplicated for each component.*

these points. If this local connectivity graph consists of multiple disconnected components, the old elaston is removed and new elastons are created, one for each component. The local frame of the new elastons is again found through a covariance analysis.

2. For each GMLS sample point, the local connectivity graph of all elastons influenced by that sample point is considered. If this connectivity graph consists of multiple disconnected components, the GMLS sample point is duplicated and each component gets its own independent copy (see Fig. 5.10).

5.7 Results

In order to evaluate the presented approach, we start with a quantitative convergence analysis and qualitative evaluations of our approach, followed by experiments that demonstrate the generality of the method. For visualization we embed a high-resolution triangle mesh into the deformation field, which could, however, be replaced by any sample-based geometry representation (triangle soups, point clouds).

Convergence. We performed a series of numerical evaluations to verify the accuracy of our method. Figure 5.11 shows representative plots for solids, shells, and rods, which are subjected to a gravitational force (and twist for the lower-right rod). The cube is constrained at its top, the shell and rod are clamped at their left-hand side as depicted in the insets of Fig. 5.11. The thickness of the shell and rod is equal to one percent of the object's side length.

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

Qualitative Verification. We also verified the qualitative behavior of our model on a couple of well-known test cases for shells and rods. Figure 5.12 shows a thin cylinder that develops the expected buckling patterns as it is compressed. Figure 5.13 demonstrates that we are able to reproduce the characteristic dynamic behavior of rods, building plectonemes and helical perversions as shown previously by Spillmann and Teschner [2007] and Bergou et al. [2008].

Non-Manifold Connections. Real-world objects often consist of complex assemblies of different forms of geometry. Our method is able to handle these

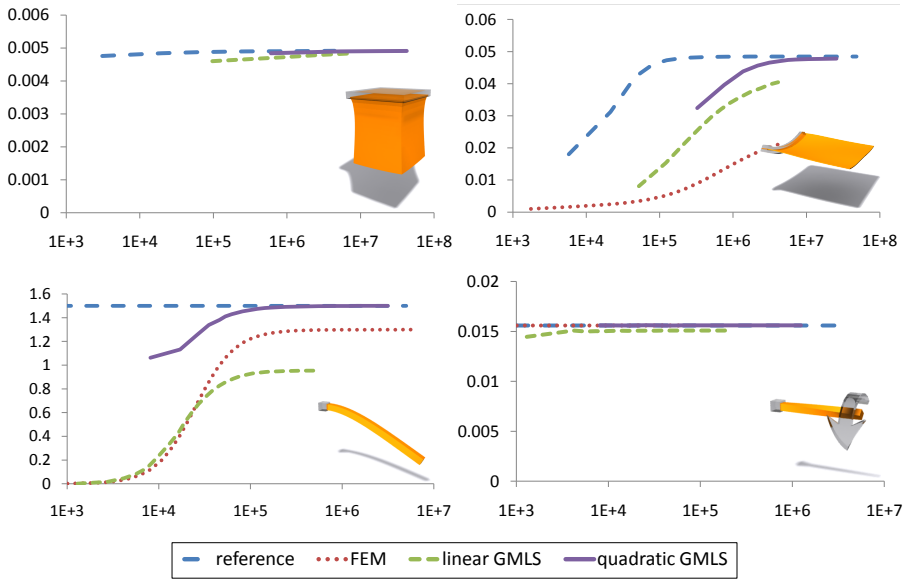


Figure 5.11: *Convergence of our method compared to standard approaches or analytic solutions for a clamped solid, shell, and rod, subject to gravity and twist. The plots show displacement values of the furthest point for increasing (computational) complexity, measured as the number of non-zeros of the stiffness matrix.*

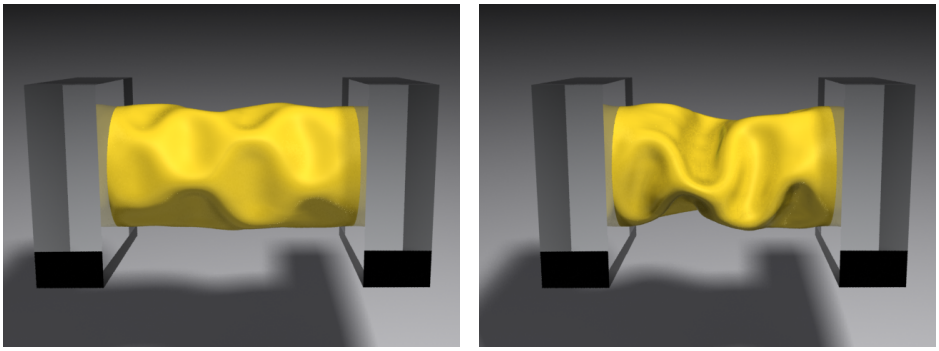


Figure 5.12: *A cylinder shows the typical buckling patterns as it is getting more and more compressed.*

mixed cases in a unified manner as demonstrated in Fig. 5.14. These examples would be difficult to realize by combining several specialized methods.

Plasticity. Objects can not only transition between solid, shell, and rod in space (Fig. 5.14 and Fig. 5.16), regime changes can also develop over time [Terzopoulos and Fleischer, 1988]. In plastic and viscous deformations, for instance, material can be stretched into thin sheets or strands (Fig. 5.15), whose elastic behavior can correctly be captured by our approach.

Cutting. Thin structures can also show up dynamically due to cutting or fracturing. We handle cutting by adapting the idea of a connectivity graph of Steinemann et al. [2006b] and the virtual node approach of Molino et al. [2004]. Virtual nodes also enable efficient simulation of spatially close features. By introducing new elastons and splitting GMLS samples, we keep

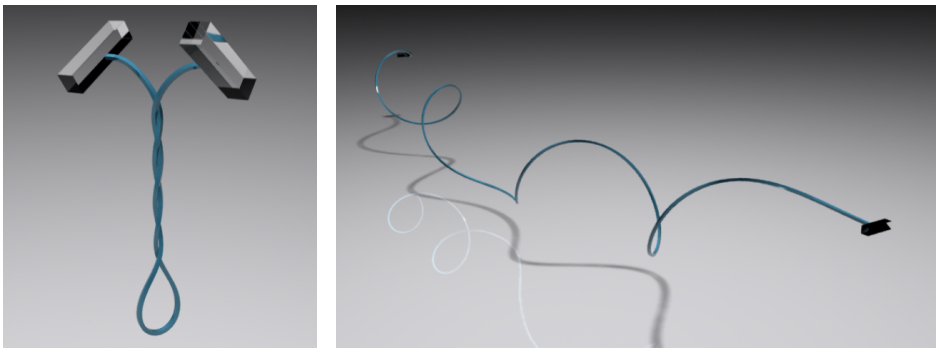


Figure 5.13: *Twisting a thin rod at both ends generates a* plectoneme *(left). Straightening a helical rod and moving its two ends back together results in a* helical perversion.

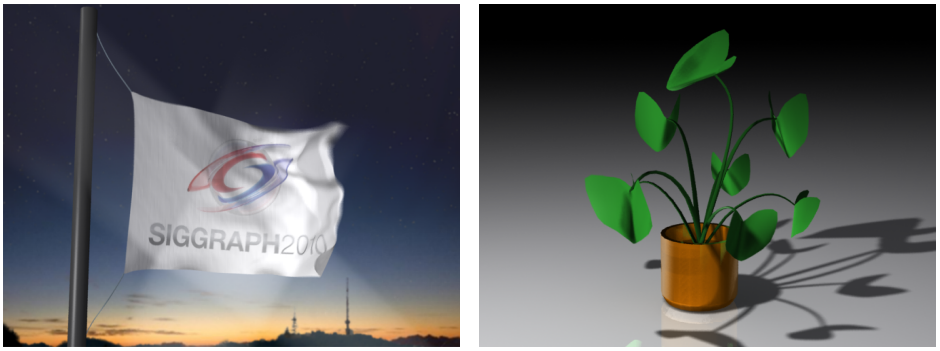


Figure 5.14: *These models show complex interaction between different types of geometry, handled in a unified way by our approach.*

distinct the motion of close features while avoiding high sampling density (e.g., the fish's spikes in Fig. 5.16).

Merging. To merge objects we simply resample as per Section 5.5.1. As soon as elastons of an object fall within influence of another object's GMLS basis, resampling merges the objects. When merging is not desired, we use the virtual node approach described above. Fig. 5.16 depicts four-bunny fusion.

Timings. For a representative selection of examples, Table 5.1 shows average timings for matrix precomputation, per-frame matrix assembly, and per-frame solution of the involved linear system. It can be observed that the time required for solving the linear system depends not only on the number of DOFs, but also on the density of the stiffness matrix, which decreases from solids over shells to rods. While the measured timings are comparably high, we should not expect a very general method to outperform specialized meth-

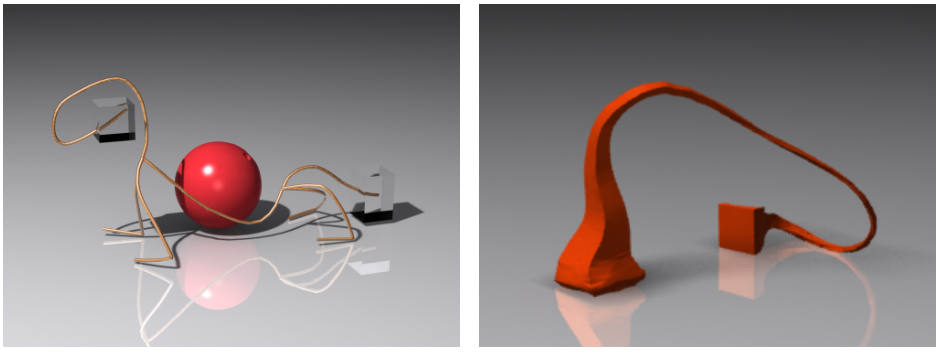


Figure 5.15: *A ball drops on a wire-dog and deforms it plastically (left). An elastoplastic cuboid is stretched under gravity (right).*

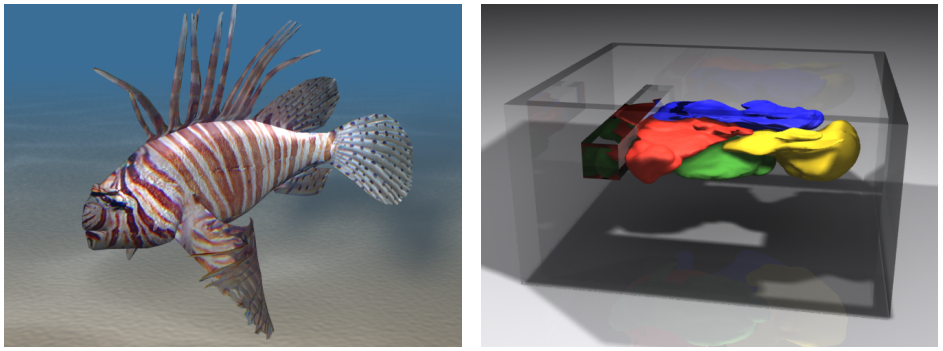


Figure 5.16: *The fish model consists of solid-, shell-, and rod-like regions (left). Four elastoplastic bunnies are compressed and merged into a solid block (right).*

ods in the sense of accuracy vs. computational costs. We believe that when a unified code is desirable, the simplicity and generality of our approach outweigh this limitation. Moreover, note that the matrix assembly could easily be sped up by parallelizing the strain corotation over all elastons.

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

Table 5.1: *Timings (in seconds) for stiffness matrix precomputation (t_{pre}), assembly of the global stiffness matrix (t_{asm}), and linear system solve (t_{sol}), taken on an Intel Core2 Duo 2.4 GHz. The first three columns denote the number of elastons, number of GMLS samples and the number of DOFs, respectively.*

5.8 Discussion and Outlook

The synthesis of elaston-based integration with GMLS-based displacement discretization opens exciting avenues for further exploration. While interactive for basic examples, our code will benefit from optimization and concrete improvements: First, our rudimentary collision handling framework should incorporate hierarchical detection accelerations and more stable collision response. Second, our implementation uniformly samples the integration

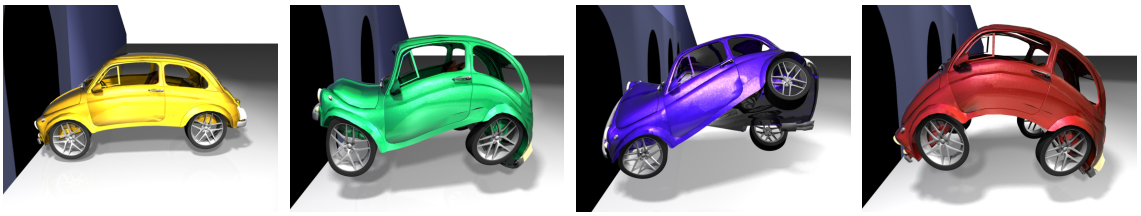
domain. Deformation modes vary widely in space and time, suggesting the profitable use of adaptive placement of integration points guided by geometric or data-driven criteria [An et al., 2008]. Online adaptive refinement of the simulation DOFs would help detailed features such as creases to form at lower cost.

The expressive range of our system could be further explored. We used a simple embedded triangle mesh to visualize a detailed surface, which complicates topological changes such as cutting. Recent advanced embedding strategies, surface tracking or point-based strategies could address this challenge. Given the promising results thus far, further refinement of our error analysis for the method would certainly be insightful.

Beside these improvements on the actual forward simulation, it would also be attractive to include artist-friendly control techniques into the framework to make the generation of desired animation sequences more directable. In the next chapter we will pursue this goal and present an example-based guiding technique on the basis of desired poses. While we introduce this method in the context of nonlinear FEM, the actual principles are general and should be transferable to an extended version of the presented elaston approach that builds up on geometrically nonlinear strain. As shown in the master thesis of Jeronimo Bayer [2011], such an extension is indeed realizable, furthermore also allowing for effectively eliminating numerical dissipation during time integration.

CHAPTER 6

Art-directable Elastic Potentials



The previous chapters primarily considered a frequently taken approach to mimic physics of real-world deformable objects: Starting with continuum mechanical models we developed new discretization procedures to simplify their practical and numerical handling. The overall goal, however, was still aiming at generating the dynamic behavior described by the *classic* governing PDEs of elasticity.

In this chapter we will go a step further and take a artist-centric view on simulations. When animating elastic objects, their deformation is mainly driven by the chosen material model and its parameter values. Often, an artist starts with a vision on how an object should deform. However developing the necessarily skills to get an intuitive understanding on how to set up according (inhomogeneous and possibly anisotropic) material parameters is a difficult task in general.

We therefore present an approach that flips the causal relation between material model and resulting deformation. It allows setting up a specialized elastic

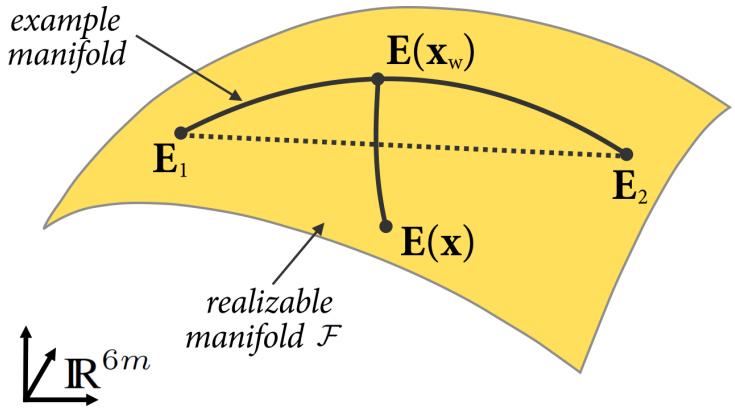


Figure 6.1: *Method Overview: The shape descriptor* $\mathbf{E}(\mathbf{x})$ *for the current configuration* $\mathbf{x}$ *is projected onto the example manifold spanned by* $\mathbf{E}_1$ *and* $\mathbf{E}_2$*. The projection* $\mathbf{E}(\mathbf{x}_\mathbf{w})$ *is then used to construct an elastic potential attracting* $\mathbf{x}$ *toward the examples.*

model in a very simple manner: Provided with a set of preferred example poses, our approach generates automatically a corresponding elastic potential that acts additionally to an existing conventional elastic potential and guides deformations toward these preferred poses. As such, it *implicitly* defines a highly complex inhomogeneous and anisotropic material model that guides the deformations without requiring non-physical forces to be introduced.

6.1 Overview

The presented method builds on the foundations of nonlinear continuum mechanics — but it will also extend other types of simulators to handle example-based materials. The essential idea is to define an additional elastic potential that attracts a solid to its subspace of characteristic deformations, to which we refer as the *example manifold*.

We first introduce the example manifold (Section 6.2), then explain how to project arbitrary configurations onto it (Section 6.3) and finally derive the elastic example potential along with a dedicated integrator (Section 6.4).

Before we describe each of these components in detail, we will briefly sketch their interplay during example-based simulation. Although we eventually use an implicit solver, a clearer picture can be drawn when considering the case of explicit example-based dynamics, which is summarized in Algorithm 3. Fig. 6.1 gives a visual illustration of the involved notions and concepts.

We start by converting the k example poses into shape space descriptors $\mathbf{E}_i$ spanning the example manifold (line 1). In each simulation step, we first

Require: initial state $\mathbf{x}$, $\mathbf{v}$
Require: example poses $\mathbf{x}^0, \ldots, \mathbf{x}^k$
1: compute shape vectors $\mathbf{E}_i = \mathbf{E}(\mathbf{x}^i)$ // §6.2
2: **while** simulating **do**
3: $\mathbf{x_w} = \text{project}(\mathbf{x})$ // §6.3
4: $\mathbf{f}_{\text{ex}} = -\nabla_{\mathbf{x}} W(\mathbf{x_w}, \mathbf{x})$ // §6.4
5: compute elastic forces $\mathbf{f}_{\text{el}}$ and external forces $\mathbf{f}_{\text{ext}}$
6: step dynamics using $\mathbf{f}_{\text{tot}} = \mathbf{f}_{\text{ex}} + \mathbf{f}_{\text{el}} + \mathbf{f}_{\text{ext}}$
7: **end while**

Algorithm 3: Explicitly integrated example-based simulation

project the current configuration $\mathbf{x}$ onto the example manifold by minimizing (6.5) to obtain $\mathbf{x_w}$ (line 3). In order to compute forces $\mathbf{f}_{\text{ex}}$ that attract the current configuration to the example manifold, we temporarily use $\mathbf{x_w}$ as a rest state and construct an elastic potential $W(\mathbf{x_w}, \mathbf{x})$ (line 4). Adding forces from the conventional simulator (line 5), we step positions and velocities forward in time (line 6).

6.2 Example Manifold

A deformation is a change in shape. We will develop a definition of the *example manifold*, which describes the set of typical, desirable deformations of a solid. Before we can define this set, we need a way to think about sets of deformations.

Strain as a basis for the space of all deformations. When we think about deformations, we want to "factor out" global rotation and translations, as these do not affect *shape.* The same request also applies locally: if parts of a solid (the arms of a character) transform rigidly, they have (locally) not changed in shape. The same reasoning can be found in the construction of nonlinear deformation measures — and, indeed, the metric tensor offers exactly these desired properties [Terzopoulos et al., 1987]: it measures only local stretching and shearing and is therefore a natural basis for constructing a "space of all deformations."

Let us now formalize this construction in the discrete setting: assume that we are given a discrete representation of a solid in the form of a tetrahedral mesh with n nodes and m elements. Further, let $\bar{\mathbf{x}} \in \mathbb{R}^{3n}$ and $\mathbf{x} \in \mathbb{R}^{3n}$ denote position vectors describing undeformed and deformed configurations, respectively. The deformation induced by a given configuration $\mathbf{x}$ can be quantified

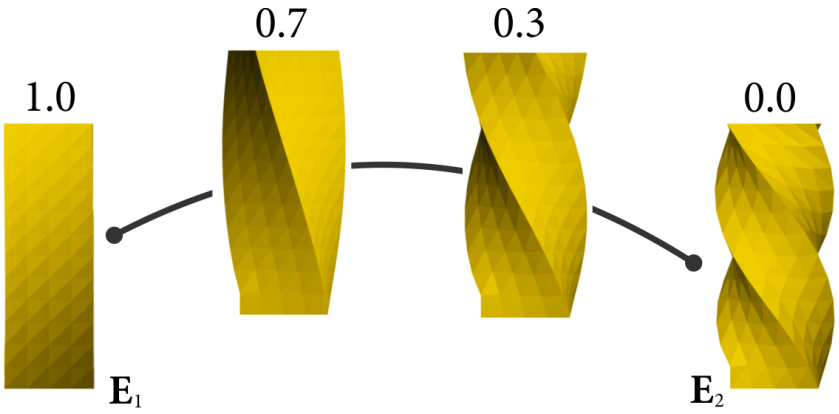


Figure 6.2: *Example interpolation: reconstructed geometry for different convex combinations of shape descriptors $\alpha\mathbf{E}_1 + (1-\alpha)\mathbf{E}_2$ (as indicated above each pose).*

pointwise (equivalently) by the deformation gradient $\mathbf{F}(\bar{\mathbf{x}},\mathbf{x}) = \partial\mathbf{x}/\partial\bar{\mathbf{x}}$, the rotation-invariant right Cauchy-Green ("metric") tensor $\mathbf{C}(\bar{\mathbf{x}},\mathbf{x}) = \mathbf{F}^T\mathbf{F}$, or the Green strain tensor $\mathbf{E}(\bar{\mathbf{x}},\mathbf{x}) = \frac{1}{2}(\mathbf{C}(\bar{\mathbf{x}},\mathbf{x}) - \mathbf{I})$ (see Section 3.1.2 or [Bonet and Wood, 1997]).

Restricting $\mathbf{x}(\bar{\mathbf{x}})$ to be piecewise linear over elements, the Green strain is constant per tetrahedron. Excluding degenerate configurations (with inverted elements), the $6m$-vector of elemental strains $\mathbf{E} = [\mathbf{E}_1,..,\mathbf{E}_m]^T \in \mathbb{R}^{6m}$ fully encodes any specific deformation, i.e., it serves as a unique *descriptor* of the deformation, and in particular one that is invariant under elemental rotations.

While every deformation maps to a descriptor, the converse is false: Not every descriptor is *reconstructible* in the sense that it corresponds to a deformation. The space of reconstructible descriptors, the image of the map $\mathbf{x} \to \mathbf{E}(\mathbf{x})$, is the *realizable manifold* $\mathcal{F} \subset \mathbb{R}^{6m}$ (see Fig. 6.1). This definition fulfills our first goal, for we can now refer to *sets of deformations*, in a rotation- and frame-invariant manner, by refering to subsets of $\mathcal{F}$.

Example manifold by example interpolation Suppose that we are given two specific example poses $\mathbf{x}_1$ and $\mathbf{x}_2$. We might interpolate between these examples by interpolating their descriptors $\mathbf{E}_1 = \mathbf{E}(\mathbf{x}_1)$ and $\mathbf{E}_2 = \mathbf{E}(\mathbf{x}_2)$ in $\mathbb{R}^{6m}$:

$$\mathbf{E}(w) = (1-w)\mathbf{E}_1 + w\mathbf{E}_2, \tag{6.1}$$

for some interpolation weight w. This approach linearly interpolates the stretch and shear of each element, and therefore results in smooth interpolation of all elements as illustrated in Fig. 6.2. As we show below, the length of any line segment inside an interpolated element is bounded by its corresponding length in the two examples.

Unfortunately, the interpolated descriptor is generally not realizable: Isolated elements can always satisfy the prescribed strains of the descriptor, however, assemblies of elements generally cannot. In a second step, we therefore find the closest realizable strain $\mathbf{E}(\mathbf{x}_w) \in \mathcal{F}$ and corresponding configuration $\mathbf{x}_w$ by solving the least squares minimization

$$\min_{\mathbf{x}_w} W_I(\mathbf{x}_w, w) = \min_{\mathbf{x}_w} \frac{1}{2} |\mathbf{E}(\mathbf{x}_w) - \mathbf{E}(w)|_F^2, \tag{6.2}$$

where the vector norm $|\cdot|_F$ is defined to match the sum of the Frobenius norms of the elemental strain tensors. We refer to the objective function $W_I(\mathbf{x}_w, w)$ as the *interpolation energy*, whose minimization defines the projection $\Pi : R^{6m} \to \mathcal{F}$ as $w \mapsto \mathbf{x}_w$. The image of the interpolating line segment $\mathbf{E}(w)$, $w \in [0,1]$ (dotted line in Fig. 6.1) under the projection Π is an *example curve* $\mathbf{E}(\mathbf{x}_w)$ (continuous line in Fig. 6.1) on the realizable manifold $\mathcal{F}$.

The procedure described above is readily generalized to an arbitrary number of poses n, where we introduce a weight w_i for each example pose $\mathbf{E}_i = \mathbf{E}(\mathbf{x}_i)$. The interpolated strain then simply becomes $\mathbf{E}(\mathbf{w}) = \sum_i^n w_i \mathbf{E}_i$ with $\mathbf{w} = (w_1, \ldots, w_n)^T$. By using these definitions in (6.2) we obtain the *example manifold* $\mathcal{E} \subset \mathcal{F}$ of realizable strains.

Bounded Lengths. Interpolating poses in shape space gives us the useful property of invariance under (local) translation and rotation. Having a closer look at the right Cauchy-Green tensor $\mathbf{C}$ reveals additionally the following nice property for the interpolants: Consider again two shapes $\mathbf{x}^1$ and $\mathbf{x}^2$ with strain space vectors $\mathbf{E}^1$ and $\mathbf{E}^2$. Using the definition of the Green strain, the linear strain interpolation $\mathbf{E}(w)$ can also be written as

$$\begin{aligned} \mathbf{E}(w) &= w\mathbf{E}^1 + (1-w)\mathbf{E}^2 \\ &= \frac{1}{2}(w\mathbf{C}^1 + (1-w)\mathbf{C}^2 - \mathbf{I}) = \frac{1}{2}(\mathbf{C}(w) - \mathbf{I}) \end{aligned}$$

where $\mathbf{C}^1$ and $\mathbf{C}^2$ are the concatenated right Cauchy-Green deformation tensors of the two shapes. Now assume we are given an arbitrary direction vector $\mathbf{D}$ in the undeformed configuration whose squared length is given by $\mathbf{D}^T\mathbf{D}$. Let $\mathbf{d} = \mathbf{F}\mathbf{D}$ denote the corresponding direction vector mapped to the deformed state. We can determine its squared length by

$$\mathbf{d}^T\mathbf{d} = \mathbf{D}^T\mathbf{F}^T\mathbf{F}\mathbf{D} = \mathbf{D}^T\mathbf{C}\mathbf{D},$$

i.e., $\mathbf{C}$ measures the new length after deformation [Bonet and Wood, 1997]. If we now use the linear interpolated tensor $\mathbf{C}(w)$ for measuring the length

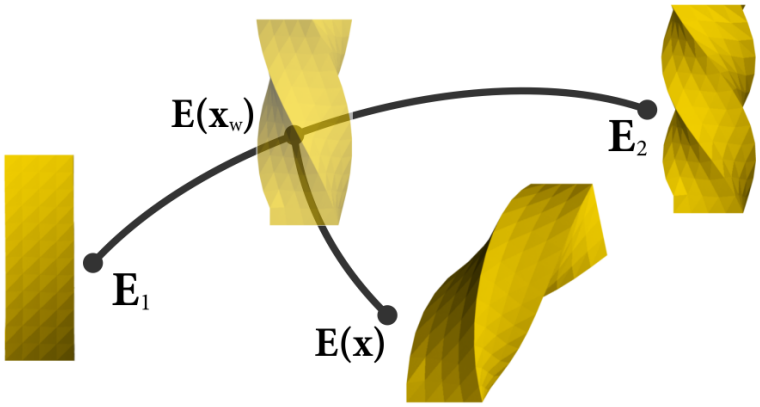


Figure 6.3: *Example projection: the current configuration (in strain space representation)* $\mathbf{E}(\mathbf{x})$ *is projected onto the example manifold, yielding the closest point* $\mathbf{E}(\mathbf{x_w})$.

$l^2(w)$ in between two shapes, we find

$$\begin{aligned} l^2(w) &= \mathbf{D}^T\mathbf{C}(w)\mathbf{D} \\ &= \mathbf{D}^T(w\mathbf{C}^1 + (1-w)\mathbf{C}^2)\mathbf{D} \\ &= w\mathbf{D}^T\mathbf{C}^1\mathbf{D} + (1-w)\mathbf{D}^T\mathbf{C}^2\mathbf{D} \\ &= wl_1^2 + (1-w)l_2^2. \end{aligned}$$

This means that the squared length at the two poses gets linearly interpolated during shape interpolation and therefore are upper and lower bounds for all lengths achieved in poses in between. It is important to note that while this property is true for individual elements, the interpolated strains $\mathbf{C}(w)$ can in general not be realized by elements organized in meshes and a least squares solution is sought. Nevertheless, due to the least squares optimality, we can still expect the interpolated meshes to nicely interpolate between the shapes. This is also supported by the various interpolation experiments we performed.

6.3 Manifold Projection

Our ultimate goal is to formulate a force that attracts the current configuration of the solid toward the example manifold, equivalently, attracting $\mathbf{x}$ toward its projection $\mathbf{x_w}$ on $\mathcal{E}$ (see Fig. 6.3).

Formulating the projection requires a suitable distance measure. Inspired by Chao et al. [2010] and Wirth et al. [2010], we approximate the geodesic distance on $\mathcal{F}$ between two shapes [Kilian et al., 2007] using the elastic potential $W(\bar{\mathbf{x}}, \mathbf{x})$. Many reasonable choices for W are compatible with our

approach (see Section 3.1.2); for concreteness we choose the potential arising from the energy density

$$W(\bar{\mathbf{x}},\mathbf{x}) = \mu|\mathbf{E}(\bar{\mathbf{x}},\mathbf{x})|_F^2 + \frac{\lambda}{2}\left(\frac{V(\mathbf{x})}{V(\bar{\mathbf{x}})} - 1\right)^2, \tag{6.3}$$

where λ and μ are material coefficients and $V(\cdot)$ measures the volume of a given configuration as the sum of elemental volumes. This simple extension of the St. Venant-Kirchhoff material [Bonet and Wood, 1997] replaces the usual second term with one that allows the simulation to recover from inversions [Picinbono et al., 2003].

The projection $\mathbf{x} \mapsto \mathbf{x_w}$ corresponds to the minimization

$$\min_{\mathbf{x_w}} W(\mathbf{x_w},\mathbf{x}) \quad s.t. \quad \mathbf{x_w} \in \mathcal{E}$$

or equivalently, invoking the extremizing conditions of (6.2),

$$\min_{\mathbf{x_w},\mathbf{w}} W(\mathbf{x_w},\mathbf{x}) \quad s.t. \quad \nabla_{\mathbf{x_w}} W_I(\mathbf{x_w},\mathbf{w}) = 0. \tag{6.4}$$

The resulting constrained optimization problem is nonlinear both in the objective function and the constraints. Applying the penalty method for constraint enforcement, we minimize

$$W_p(\mathbf{x_w},\mathbf{w},\mathbf{x}) = W(\mathbf{x_w},\mathbf{x}) + \gamma\,|\nabla_{\mathbf{x_w}} W_I(\mathbf{x_w},\mathbf{w})|^2 \tag{6.5}$$

with respect to $\mathbf{x_w}$ and $\mathbf{w}$, for a sufficiently large fixed penalty stiffness γ. The weights $\mathbf{w}$ are constrained such that $w_i \geq 0$ and $\sum_i w_i = 1$. These constraints restrict the example space to interpolations of the provided examples. Since small extrapolations are in general not harmful, we enforce these constraints also weakly by adding simple quadratic energies to (6.5).

6.4 Example-based Simulation

With the definition of the example manifold in place and the projection procedure defined, we are all set for proceeding to the actual example-based simulation. As mentioned before, our system integrates readily with existing solid simulators, but is particularly convenient to build on top of a finite element solver, allowing the reuse of code for deformation measure and elastic potentials.

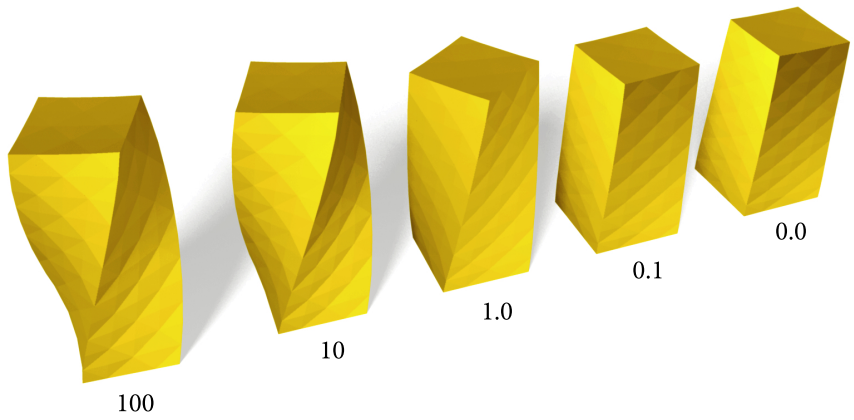


Figure 6.4: *Impact of scaling the material stiffness of the example potential relative to the conventional potential's stiffness. An elastic bar is subjected to gravity and simulated statically with different scaling coefficients (as indicated).*

6.4.1 Variational Statics

Assume that the conventional potential is given by $W_c(\mathbf{x}) = W(\bar{\mathbf{x}}, \mathbf{x})$. In order to solve for static equilibrium we require that the sum of all external forces $\mathbf{f}_{\text{ext}}$ equal the internal forces induced by the new, augmented material with potential $W_c(\mathbf{x}) + W_p(\mathbf{x}_\mathbf{w}, \mathbf{x})$ (see also Section 3.3.5). This yields the following system of equations

$$\nabla_\mathbf{x} W_c(\mathbf{x}) + \nabla_\mathbf{x} W_p(\mathbf{x}_\mathbf{w}, \mathbf{w}, \mathbf{x}) = \mathbf{f}_{\text{ext}}. \tag{6.6}$$

In this expression we implicitly assume that the projection $\mathbf{x}_\mathbf{w}$ and weight vector $\mathbf{w}$ are always the corresponding minimizer of (6.4). In order to handle these two minimizations in the same framework, we can recall that equation (6.6) is actually the necessary condition for a minimizer of the joint total energy

$$W_{\text{tot}}(\mathbf{x}_\mathbf{w}, \mathbf{w}, \mathbf{x}) = W_c(\mathbf{x}) + W_p(\mathbf{x}_\mathbf{w}, \mathbf{w}, \mathbf{x}) + W_{\text{ext}}(\mathbf{x}), \tag{6.7}$$

where we assumed that the external forces are conservative. By minimizing W_{tot} simultaneously in $\mathbf{x}_\mathbf{w}$, $\mathbf{w}$ and $\mathbf{x}$, we solve both problems at the same time: The static solution is found with the correct projection for the example potential.

The two elastic potentials in (6.7) can be chosen independently, but in our implementation we use (6.3) for both of them. In order to further reduce the number of variables, we set the material constants of the example potential to scalar multiples of those of the conventional potential, leaving a single parameter to set. Fig. 6.4 illustrates the impact of this parameter for a range of practical values.

6.4.2 Variational Implicit Euler

For dynamic simulation, we opt for the implicit Euler method and pursue a similar strategy as for statics, i.e., we formulate the time stepping scheme as an optimization problem. We start from the canonical equations of motion

$$\mathbf{M}\ddot{\mathbf{x}} + \nabla_{\mathbf{x}} W_c(\mathbf{x}) + \nabla_{\mathbf{x}} W_p(\mathbf{x_w}, \mathbf{w}, \mathbf{x}) = \mathbf{f}_{\text{ext}} \tag{6.8}$$

where $\mathbf{M}$ is the mass matrix. In using our augmented elastic potential, we again assume that $\mathbf{x_w}$ and $\mathbf{w}$ are minimizers of (6.4). In order to arrive at a single optimization problem we first apply the implicit Euler integration scheme (Section 3.3.5) to obtain a nonlinear system of equations

$$\mathbf{M}(\frac{\mathbf{x}_n - \mathbf{x}_o}{h^2} - \frac{\mathbf{v}_o}{h}) + \nabla_{\mathbf{x}} W_c(\mathbf{x}_n) + \nabla_{\mathbf{x}} W_p(\mathbf{x_w}, \mathbf{w}, \mathbf{x}_n) = \mathbf{f}_{\text{ext}},$$

where h is the step size, $\mathbf{x}_n$ are new positions and $\mathbf{x}_o$ and $\mathbf{v}_o$ the old positions and velocities, respectively. We can solve this system in an elegant way by minimizing the objective function

$$\begin{aligned} H(\mathbf{x}_n, \mathbf{x_w}, \mathbf{w}) \quad = \quad & \frac{h^2}{2}(\frac{\mathbf{x}_n - \mathbf{x}_o}{h^2} - \frac{\mathbf{v}_o}{h})^T \mathbf{M}(\frac{\mathbf{x}_n - \mathbf{x}_o}{h^2} - \frac{\mathbf{v}_o}{h}) \\ & + W_c(\mathbf{x}_n) + W_p(\mathbf{x_w}, \mathbf{w}, \mathbf{x}_n) + W_{ext}(\mathbf{x}_n)\ . \end{aligned} \tag{6.9}$$

By optimizing (6.9) for $\mathbf{x}_n$, $\mathbf{x_w}$, and $\mathbf{w}$, we simultaneously solve the coupled problems of projection and time stepping.

6.4.3 Numerical Optimization

The discussed optimization problems for the static and dynamic case are solved robustly by a classic Newton-Raphson procedure. In order to improve convergence, we employ a line search scheme and additionally apply diagonal Hessian correction in case of indefinite matrices [Nocedal and Wright, 2006]. The dimension of the resulting linear systems is roughly twice that of conventional simulations. We employ a sparse direct Cholesky solver for solving the resulting symmetric positive definite systems [Schenk and Gärtner, 2002].

Observe that the first term of (6.4) involves the optimization of the elastic energy with respect to the undeformed configuration. This formulation is reminiscent of problems found in, e.g., variational shape optimization and mesh adaptation [Thoutireddy and Ortiz, 2004]. Although the derivations do not pose any particular problems, these so-called *configurational forces* are to our knowledge new to graphics, which is why we next list the required gradients and Hessians.

Gradients and Hessians. In order to perform Newton optimization of the energies (6.7) and (6.9) we need first and second derivatives of (6.2) and (6.3). While derivatives of the second term (volume-change) in (6.3) are quite simple to derive [Picinbono et al., 2003], the derivatives of the first term (Green strain) are more involved and given here. We resort to index notation in order to write the element-wise derivatives in compact form. Let $\bar{\mathbf{x}}_{mn}$ and $\mathbf{x}_{mn}$ denote the n-th components of position vectors for node m in the undeformed and deformed configuration, respectively. Introducing the matrix

$$\mathbf{S} = \begin{pmatrix} -1 & -1 & -1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and defining $\mathbf{d}_{ij} = \mathbf{x}_{ki}\mathbf{S}_{kj}$ and $\mathbf{D}_{ij} = \bar{\mathbf{x}}_{ki}\mathbf{S}_{kj}$, the deformation gradient becomes

$$\mathbf{F}_{ij} = \mathbf{d}_{ik}\mathbf{D}^{-1}_{kj}.$$

The first derivative of $\mathrm{tr}(\mathbf{E}^T\mathbf{E}) = \mathbf{E}_{ij}\mathbf{E}_{ij}$ for the deformed configuration can then be stated compactly as

$$\frac{\partial(\mathbf{E}_{ij}\mathbf{E}_{ij})}{\partial \mathbf{x}_{mn}} = 2(\mathbf{S}\mathbf{D}^{-1}\mathbf{E}\mathbf{F}^T)_{mn},$$

and the second derivative is obtained as

$$\begin{aligned} \frac{\partial^2(\mathbf{E}_{ij}\mathbf{E}_{ij})}{\partial \mathbf{x}_{mn}\partial \mathbf{x}_{st}} &= (\mathbf{F}\mathbf{F}^T)_{nt}(\mathbf{S}\mathbf{D}^{-1}\mathbf{D}^{-T}\mathbf{S}^T)_{ms} \\ &+ (\mathbf{S}\mathbf{D}^{-1}\mathbf{F}^T)_{mt}(\mathbf{S}\mathbf{D}^{-1}\mathbf{F}^T)_{sn} \\ &+ 2\delta_{nt}(\mathbf{S}\mathbf{D}^{-1}\mathbf{E}\mathbf{D}^{-T}\mathbf{S}^T)_{ms}. \end{aligned}$$

The derivatives with respect to the undeformed configuration assume a similar form: the first derivatives are

$$\frac{\partial(\mathbf{E}_{ij}\mathbf{E}_{ij})}{\partial \bar{\mathbf{x}}_{mn}} = -2(\mathbf{S}\mathbf{D}^{-1}\mathbf{E}\mathbf{C})_{mn},$$

while second derivatives follow as

$$\begin{aligned} \frac{\partial^2(\mathbf{E}_{ij}\mathbf{E}_{ij})}{\partial \bar{\mathbf{x}}_{mn}\partial \bar{\mathbf{x}}_{st}} &= (\mathbf{C}\mathbf{C})_{nt}(\mathbf{S}\mathbf{D}^{-1}\mathbf{D}^{-T}\mathbf{S}^T)_{ms} \\ &+ (\mathbf{S}\mathbf{D}^{-1}\mathbf{C})_{mt}(\mathbf{S}\mathbf{D}^{-1}\mathbf{C})_{sn} \\ &+ 2\mathbf{C}_{nt}(\mathbf{S}\mathbf{D}^{-1}\mathbf{E}\mathbf{D}^{-T}\mathbf{S}^T)_{ms} \\ &+ 2(\mathbf{S}\mathbf{D}^{-1})_{sn}(\mathbf{S}\mathbf{D}^{-1}\mathbf{E}\mathbf{C})_{mt} \\ &+ 2(\mathbf{S}\mathbf{D}^{-1})_{mt}(\mathbf{S}\mathbf{D}^{-1}\mathbf{E}\mathbf{C})_{sn}. \end{aligned}$$

6.5 Example Design and Implementation

Our method relies on example poses to model the characteristic deformation behavior of elastic solids. This section describes how to design *good* examples, how to efficiently approximate geometrically and mechanically complex models, and how to extend the range of effects by controlling the influence region of examples.

6.5.1 Example Design

Our method accepts example poses in the form of deformed element meshes that have the same topology as the undeformed mesh. There are no specific requirements on the way in which these examples are created and, for instance, any geometric modeling tool [Gain and Bechmann, 2008] can be used for this purpose. An alternative way of designing natural examples is by using a static solver built on the same elastic potential W_c that is also used in simulation [Barbič et al., 2009; Mezger et al., 2008]. While the geometric modeling approach imposes virtually no restrictions on creativity, this physics-based metaphor has the advantage that deformations propagate naturally (unless enforced otherwise) and that the resulting meshes are unlikely to exhibit severely distorted or inverted elements.

Apart from the technique used for generating examples, another important question is what kind of examples should be used. In order to represent natural transitions to the rest pose, we use the undeformed configuration as part of the example set in all our experiments. Any additional example should, first and foremost, represent characteristic or extreme poses. However, examples should also be sufficiently different in order to span a diverse space of expressive deformations with as few poses as possible.

6.5.2 Embedding Triangle Meshes

Embedding denotes a class of techniques for deforming highly-detailed geometry in accordance to the deformations of a (potentially) much coarser approximation (Section 3.3.3). On the practical side, embedding is a very efficient way for increasing the level of detail to impressive amounts, as recently demonstrated by the work of Wojtan et al. [2009].

Embedded meshes can augment coarse volumetric simulations with high-quality surface details. But since the deformations of the embedded surface

follow the coarse physics of the embedding mesh, the physical detail can generally not be increased — or only to limited amounts. Our example-based approach does, however, not impose as strict restrictions as the usual fine-to-coarse coupling. Using a static solver, it is quite easy to deform an embedding mesh such that its enveloped surface assumes a desired shape. We can thus design examples that account for realistic deformations of sub-element geometry (see this chapter's teaser figure), but we can also create examples that emulate the deformation behavior of more complex mechanics, such as the buckling patterns of thin-walled cylinders (see Fig. 6.9).

Apart from manually designing example poses, one could also use an approach similar to Barbič et al. [2009] in order to automatically compute the embedding mesh that best approximates a given input surface. Using this technique, one would first run an offline simulation of a high-resolution volumetric model and then automatically deform the embedding mesh to match the surfaces of some characteristic frames. This approach could be also used to generate embedding meshes for thin shell or rod simulations.

6.5.3 Local and Global Examples

Until now, we have assumed that examples are defined on the entire domain of the solid. For such *global* examples, the deformation of one part of the object directly influences all other parts as shown in Fig. 6.8. However, it is also interesting to limit the influence of an example to individual parts of an object. Such *local* examples can be used to define deformation behavior locally and independent of other regions. Additionally, different local examples can be combined to yield even more complex global behavior as shown, e.g., in Fig. 6.7. Vice versa, by dividing a complex pose into local examples, we can already specify much of an object's characteristic behavior using only a single deformed configuration.

On a technical note, we constrain the example space to convex combinations of the individual poses in the case of global examples, which is necessary in order to obtain well-behaved strain interpolation. While it may appear tempting to allow extrapolation, doing so entails the risk of running into invalid strains: It is a simple matter to determine weights such that the extrapolation of two valid strain tensors yields a metric tensor with negative values on its diagonal — but this is not meaningful since $\mathbf{C}_{ii} = \sum_k \mathbf{F}_{ki}\mathbf{F}_{ki}$ is always positive.

For local examples, however, the convexity constraint can be relaxed: we can simply form groups of *interacting* examples such that any two poses

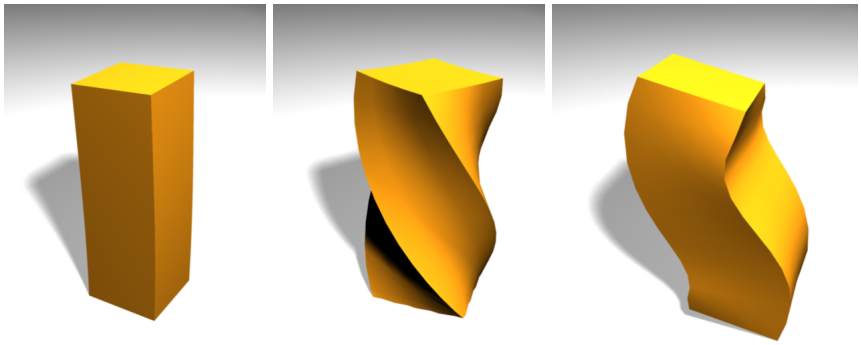


Figure 6.5: *An elastic cuboid deforms under gravity using no example, a twist example and an S-shaped example.*

from different groups have no deformed element in common – which are orthogonal to each other. In this way, we can safely enforce the convexity constraints on each group in isolation. As a practical implication, doing so allows individual parts of an object to deform independently of other parts and enlarges the space of preferred shapes in an efficient manner.

6.6 Results

This section presents a set of examples that illustrate different aspects of our approach and demonstrate typical applications.

Our method allows an animator to design and simulate complex elastic materials by merely providing a set of example poses that correspond to characteristic, desirable, or extreme deformations. Fig. 6.5 shows a simple example that illustrates this idea: by augmenting an elastic bar with an example potential constructed from twist or *S*-shaped poses, we can significantly change its deformation behavior and thus imprint different styles onto the animation. Though possible in theory, achieving the same results with a conventional simulator would require tedious tuning of an inhomogeneous, anisotropic, and probably nonlinear material. By contrast, our approach is intuitive and output-oriented, making it well-suited to design processes commonly used for creative applications.

Global examples are used to directly specify preferred deformations for an entire object, which can be understood as a *what-you-see-is-what-you-get* approach to material design. However, there are also many common objects for which the characteristic deformations are rather local than global. Moreover, different local deformations can typically occur simultaneously and

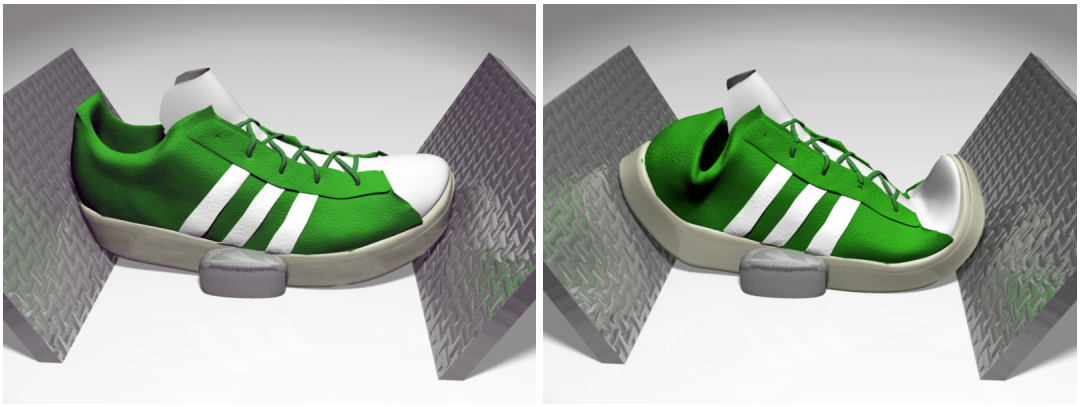


Figure 6.6: *Compressed sneaker simulated as a coarse solid. Without examples (left) and augmented with two local examples (right).*

independently of each other. This kind of behavior is illustrated in the animation shown in Fig. 6.6, for which two characteristic deformations of a shoe, namely the buckling of its toe and the bulging of its heel, are provided as local examples to the simulation.

This animation also showcases the application of embedding: the high-resolution geometry of the shoe deforms in accordance to the coarse embedding mesh — but it does so in a very plausible way. This, in turn, is due to the fact that the volumetric example meshes were generated such that the embedded mesh assumes the desired deformations, irrespective of the actual shape of the embedding mesh.

Local examples do not necessarily have to be defined over connected components, but can also couple remote regions while still affecting only a small part of the entire object. An example of this application can be seen in Fig. 6.7, which shows that, in an artistic setting, compressing the nose of a balloon dog can lead to an inflation of its ears.

Our method can be used to design deformation styles which are difficult to generate with conventional elastic materials, but are still within the range of

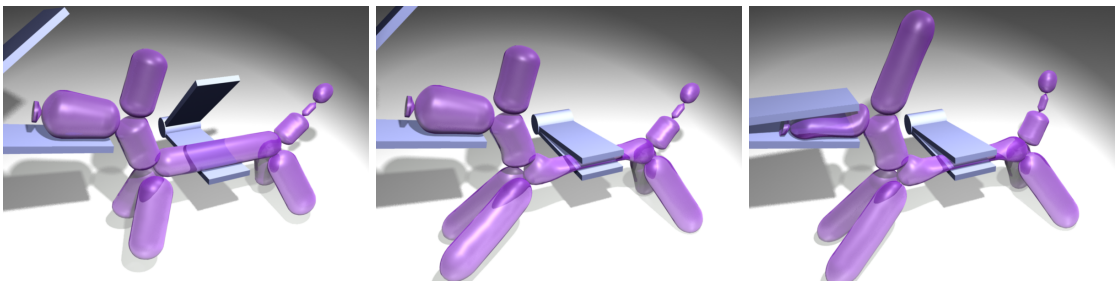


Figure 6.7: *Local examples defined over unconnected regions showcased on a balloon dog.*

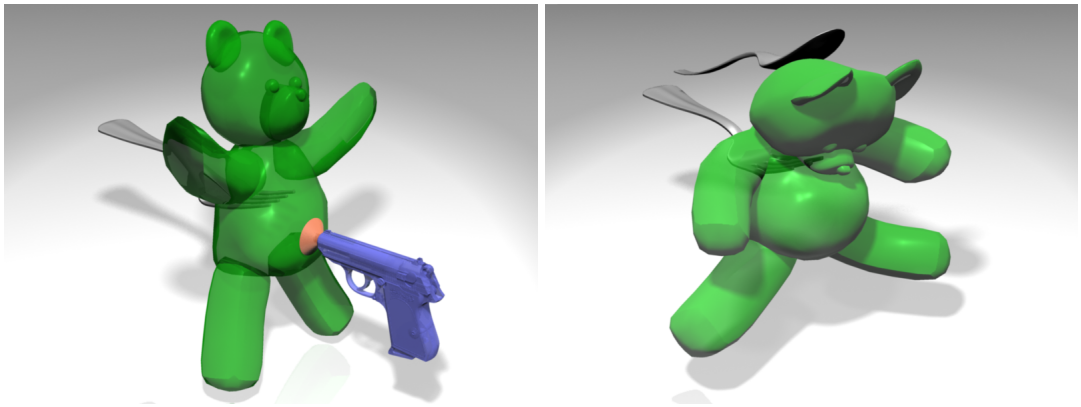


Figure 6.8: *A gummy bear is equipped with expressive examples to create an impression of personality.*

what we might expect from some exotic material. We can, however, also go a step beyond and use examples to induce deformations that clearly exceed the realm of conventional elastic materials. An example that goes along these lines can be seen in Fig. 6.8, which shows a gummy bear that, despite its jelly-like appearance, seems to have a personality of its own, driving it to deform in very peculiar ways in response to user interaction.

Similar in spirit is the example shown in the teaser figure at the beginning of the chapter, which depicts the unfortunate incident of a toy car hitting the wall beneath a fake tunnel. Using example-based simulation, we can make the car react to the impact in very diverse ways, following the exaggerated-physics style frequently found in cartoons.

Our method primarily aims at volumetric solids, but thanks to the embedding technique it can also be used to mimic the behavior of more complex mechanical models such as thin shells. The two images on top of Fig. 6.9 show our approach applied to a cylindrical shell which, when compressed, exhibits the typical diamond-shaped buckling patterns. Note that the example was not computed with a thin shell simulation code but designed with a static version of our solid simulator. The reason why the buckling patterns can still appear in a plausible way is simply that the example potential renders these deformations energetically favorable. As shown in the remaining images of Fig. 6.9, we can again specify various deformation examples to obtain diverse material effects, ranging from physically plausible deformations to art-directed physical animation.

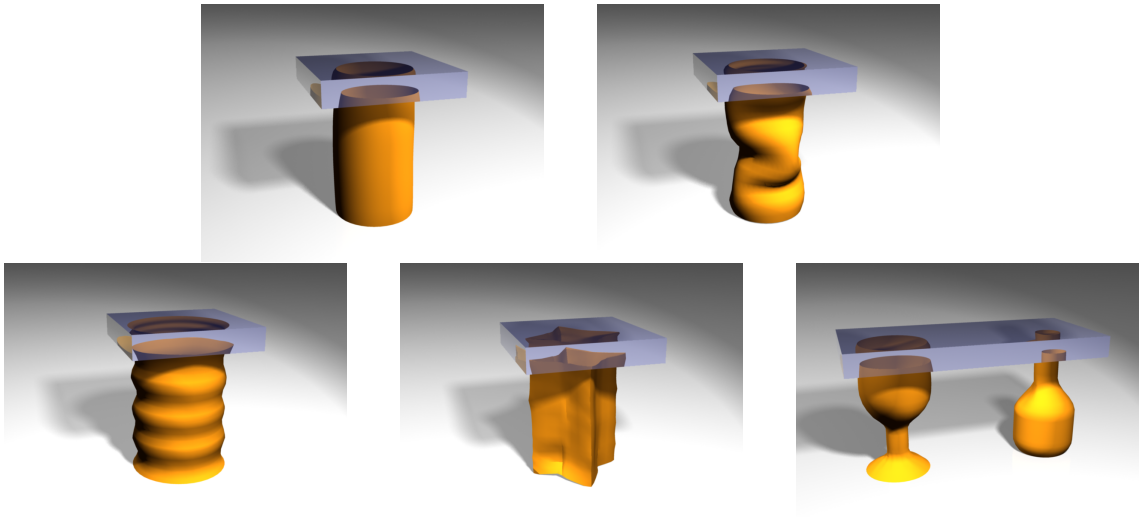


Figure 6.9: *A cylindrical surface mesh embedded in a volumetric simulation mesh. Using a buckled example pose allows us to emulate thin shell behavior. By varying the input examples we can effectively control the deformation behavior of the embedded geometry.*

Performance. We provide timings for all examples presented in this section in Tab. 6.1. All simulations were performed with a manifold constraint penalty of 10^4, a convex weight penalty of 100 and a timestep size of 0.02s.

It should be noted that the benefits of example-based elastic materials come at the price of additional computation costs. As can be seen from Tab. 6.1, the largest fraction of the time is spent on the assembly of the linear system, including the computation of gradients and Hessians, and its solution. Currently, the performance of our method does not allow its use in interactive applications such as video games. In the context of artistic material design, however, the additional costs of our approach seem acceptable as they are

Model	#DOFs	t_{asm}	t_{newton}	t_{tot}	α
Cuboid Twist	975	80	141 / 308	528 / 3064	40
Sneaker	942	107	159 / 173	680 / 1288	20
Teddy	828	59	61 / 65	1333 / 1410	40
Cylinder	227	43	57 / 328	502 / 1214	20
Car	1410	110	192 / 204	2990 / 3292	50
Balloon	1320	106	99 / 195	125 / 2196	1000

Table 6.1: *Timings (in ms) for single gradient/Hessian assembly (t_{asm}) and Newton step with line search (t_{newton}, min/max time), as well as min/max total time per timestep (t_{tot}), taken on a single core of a Intel Core i7 960, 3.2 GHz. α denotes the stiffness ratio between manifold and conventional potential.*

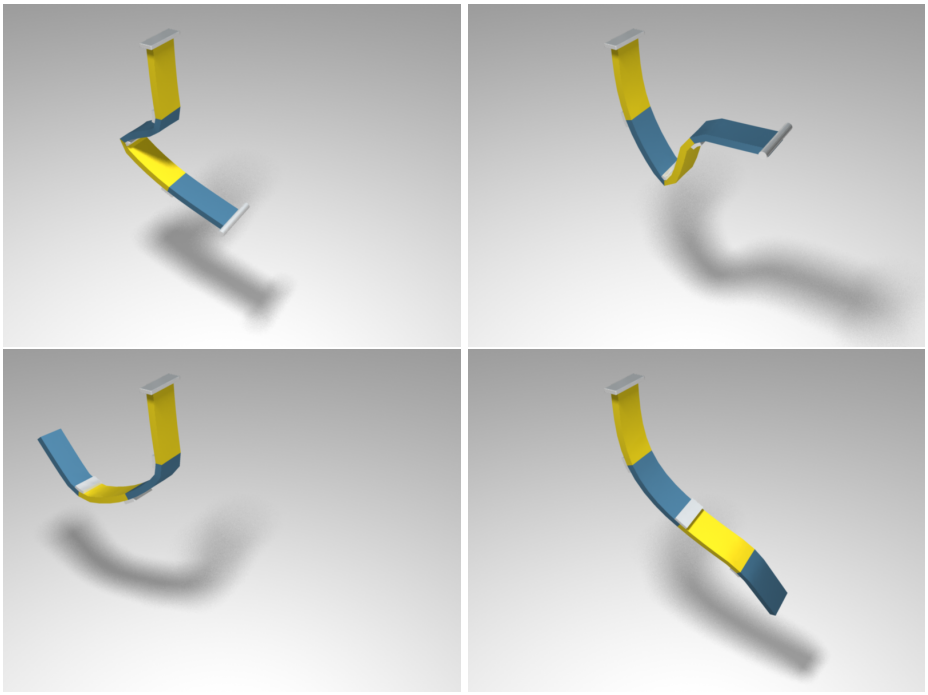


Figure 6.10: *The pendulum is fixed on the top, first lifted with a handle and then released to swing. Three hinges that only allow one-sided bending are modeled using local example poses.*

very likely to pay off in terms of time saved on tuning material parameters of conventional material models. Additionally, we foresee that future improvements on the algorithmic level can lead to substantial speed-up.

As another performance indicator, we also investigated the scaling of our method with respect to the number of example poses. Our method can faithfully handle the case of multiple examples as exemplified, e.g., in the animations of the car (Fig. 6.11), the balloon-dog (Fig. 6.7), or the pendulum (Fig. 6.10). For a quantitative analysis, we measured computation times for an increasing number of example poses on the cuboid animation. The results shown in Tab. 6.2 indicate that the number of examples is not a limiting factor

	1 ex.	2 ex.	3 ex.	4 ex.	8 ex.	12 ex.
t_{asm}	120	124	128	132	148	165
t_{slv}	244	216	213	218	220	232

Table 6.2: *Performance scaling for multiple examples (top row) illustrated on the cuboid animation (Fig. 6.5). Average timings (in ms) for assembling (t_{asm}) and solving (t_{slv}) the linear system in a single Newton step.*

Figure 6.11: *A car with four example poses, each of which is activated during the animation in response to different impact events.*

of our method: Using twelve poses instead of one, the time spent on solving the nonlinear system increases by only 15%.

6.7 Discussion and Outlook

We have only scratched the surface of what is becoming possible with example-based simulation. We see many promising directions which we would like to explore in the future. While we framed our approach in the context of deformable solids, we believe that the underlying theory can be generalized to different elastic models, such as shells and rods, being interesting in the contexts of cloth and hair simulation, for example. But also generalizations to unified approaches as presented in the last chapter are thinkable, due to the strain-based foundations of all these theories. Furthermore, we have not used the information obtained through the projection on the example manifold — an exciting application could be to couple the example weights to secondary effects, e.g., for modulating texture or surface geometry. There is also much room for further exploring the definition of the shape space, which currently only considers position information via the Green strain. It might be interesting to also account for velocities (rate of strain) or even forces. Another idea would be to directly encode different

aspects of deformation (such as incompressibility) into the definition of the shape space and, e.g., define an example manifold that *sees* only deviatoric (i.e., volume-preserving) deformations.

Our prototype implementation already indicates that our method has great potential in designing materials and art-directing simulations. However, we also see various possibilities for extensions and improvements. In particular the performance of our optimization scheme should be increased and we anticipate that Lagrangian methods will lead to better convergence than our current penalty approach [Nocedal and Wright, 2006]. Furthermore, we currently rely on the user to create examples that are meaningful in that they do neither contradict each other nor strongly counteract the underlying elastic potential. It would be desirable to develop methods that assist the user in this process by providing appropriate feedback on the quality of examples. Another promising direction would be to automatically select a set of example poses from a given input animation.

CHAPTER 7

Conclusion

In this chapter we summarize and discuss the principal contributions and suggest directions for future investigations.

7.1 Discussion

In this thesis, we visited three important areas for the simulation of deformable objects and adapted each of them to create novel methods that better match the challenging settings of graphics applications. In particular, we extended the basic elasticity models for more art-directability by introducing example-based materials, proposed a unification for the different specialized models for thin geometries by means of elastons, and presented new discretization approaches based on flexible polyhedral elements, meshless samplings, or feature-preserving cages. The thesis presented these three parts in reversed order to clarify the motivation for each of them and better demonstrate their interrelationship.

The combination of modern coordinates [Joshi et al., 2007; Lipman et al., 2008] with the Galerkin principle for elasticity followed in Chapter 4 allows us to come up with exciting and powerful simulation methods for various applications. The frequent topological changes occurring in FEM simulations during adaptive refinement, cutting and fracturing can be handled consistently with

a single code by introducing support for arbitrary polyhedral elements. This became possible by suitably combining harmonic coordinates [Joshi et al., 2007] with the method of fundamental solutions for their efficient numerical computation. Using the manifold spanned by Green coordinates [Lipman et al., 2008] as discretization space and introducing suitable linearizations allowed us to generate simulations that preserve features, independently of their actual scale. Following the idea of choosing solution subspaces with additional benefits, using MLS as meshless interpolation scheme allowed us to set up a corotated EFG approach that allows for the simplest discretization possible, while providing scalable performance and accuracy.

The ease in discretization of the developed meshless approach allows potentially handling complex graphics scenarios (e.g. scenarios requiring frequent resampling due to plastic flows or large deformations) in a straightforward manner. As shown in Chapter 4, this approach is well suited for solid geometries but becomes unstable and erroneous when handling thin geometric forms. To generalize the method to arbitrary geometries and omitting the requirement of specialized codes, Chapters 3 and 5 first follow the line of specialized codes to eventually come up with point-based elastic model, the elaston, for describing material of small extent. Applied as a quadrature rule for larger geometries of any form, this allows reproducing the results of specialized codes for each of these types in an unified manner. Extending standard MLS to its "Hermite" version, GMLS allows to get rid of colinearity issues and to generate arbitrary DOF samplings on the reduced geometries of thin-walled objects. By further enhancing the approach with an additive plasticity model, with a resampling and simulation variable transfer method and with a meshless virtual node algorithm for cutting and fracturing, the resulting code becomes broadly applicable: it can handle any dynamically changing geometry and allows reproducing a large spectrum of material effects.

While this elaston-based approach allows to flexibly reproduce classic forward dynamics described by the models of continuum physics, controllability of the simulation outcome is restricted to parameter tuning and depends largely on the intuition of the user. Chapter 6 therefore introduces intuitive example-based materials allowing the user to provide the system with a number of key poses which are then "translated" into a corresponding elastic potential that favors similar deformations. To realize this technically, we introduce "strain space" as a new space of shapes in which we naturally interpolate different deformations. Defining a projection procedure onto such interpolations allows us to formulate an additional directing potential besides an existing underlying material, both driving the final simulation. Compared to other directing techniques, this control approach *is part of* the physics and

does not employ artificial external forcing: it provides conservative forces that do not interfere with the system by undermining energy conservation, but is rather channeling the existing energy in intuitive artist-specified ways.

7.2 Future Work

We see many exciting paths for future investigations on the topics dealt in this thesis.

Designing Subspaces. In order to realize polyhedral elements, we choose a radial basis approach for numerically approximating the harmonic coordinates within each element which in turn are then used to span the actual solution subspace. Conceptually, we select just a subset of possible deformations that could be modeled by the entire radial basis function representation. In this particular case, we make the choice of approximating harmonic coordinates by a certain linear combination of radial bases to gain their interpolation properties and to be able to improve the compatibility between different element types in a mesh-based simulation framework.

This concept could, however, also be applied in a broader sense by generally using larger function spaces and, possibly dynamically, constraining them to get efficient simulations and special interpolation properties. For example, feature preservation as pursued with Green coordinates in Chapter 4 could be formulated by introducing a specific relationship between the base functions to generically create a lower-dimensional solution subspace with additional properties. Such an approach could improve the presented feature-preserving simulation approach by allowing for locally supported basis functions leading to efficiently computable sparse systems. Also since feature-preservation is not desired at any scale of the geometry (e.g. because it could prevent volume preservation) it might be interesting to investigate into suitable solution space designs that enable volume preservation in the large while being feature-preserving in the small.

Alternatively, using cage-based Green coordinates, one could also think of a discontinuous Galerkin-type of approach [Kaufmann et al., 2008], where these coordinates are employed per element (each being a cage) and C^0-continuity between elements is enforced weakly. This is similar in mind to the recent approaches of Barbić et al. [2011] or Wicke et al. [2009] building the solution spaces from local reduced bases, or the approach of Nesme et al. [2009] and Faure et al. [2011] who are designing coarse basis functions adapted to material inhomogeneities.

Inspired by the construction of MLS and GMLS interpolation, a different approach to creating special purpose basis functions lies in defining them as minimizers of specific functionals that penalize deviation from the desired properties, in order to "bake" these directly into the set of spanning functions. The approach of Ben-Chen et al. [2009] is similar to this idea in that they also enforce different properties such as rigidity and smoothness of mappings variationally.

Unified Simulation. The main drawback of the presented unified method consists in its rather poor efficiency. It stems from the fact that GMLS concentrates up to 30 DOFs at single locations such that when interacting with neighboring samples, dense coupling between the DOFs emerge, leading to poor performance when solving the resulting linear systems. One approach to tackle this problem lies in parallelization. Basically all stages of the simulation pipeline can be distributed well onto multiple cores; the linear solve being the only bottleneck. Other investigations could go into replacing the actual meshless interpolation scheme with a cheaper one, while retaining the current convergence properties, or optimizing for the number of elastons, similar to An et al. [2008]. Also from the application point of view there are open problems: While the presented approach conceptually allows for adaptive simulations, further investigations for the optimal sampling criteria need to be performed. The proposed meshless virtual node algorithm would also support fracturing material; however more research in the handling of cracking embedded visualization surfaces is necessary.

While our approach offers simple setup and handling due to its meshless nature and the reduced elaston model which makes it ideally suited for demanding graphics task, investigating further into comparing and reconciling our method with other unification efforts would be insightful. In what key properties does our method conceptually differ from higher-order FEM methods or the Cosserat point theory of Rubin [1985]? What are the fundamental requirements a unified method needs to fulfill?

Currently, we represent the solution using a volumetric solution field while the energy is measured at elastons on the reduced geometry. Further investigations could also go in the direction of directly formulating the discrete models [Grinspun et al., 2003; Bergou et al., 2008] on point sets and eliminating separate DOF and elaston samplings. Such an approach could be similar to the one of Müller et al. [2011], however with focus on physical correctness. First steps into this direction have been pursued with the master thesis on meshless shells of Liana Manukyan [2011], where a similar discrete model is constructed for shell simulations.

Example-based Simulation. Clearly, the primary deficit of the presented example-based simulation approach is its performance which stems from the inherent complexity of the nonlinear optimization problem. While fully coupling the projection and time integration step into a single iterative procedure is conceptually elegant, the resulting system still contains roughly twice the number of DOFs than the according conventional simulation. Furthermore, it ties the convergence of the projection and time integration together, therefore not allowing for the commonly used simplifications for implicit integrators like the semi-implicit variant [Baraff and Witkin, 1998]. Enforcing constrains using the penalty method is also not an optimal solution and we expect Lagrangian methods like sequential quadratic programming [Nocedal and Wright, 2006] to perform much better. It seems also an attractive direction to find an alternative problem formulation working solely with example weights $\mathbf{w}$ instead of maintaining and optimizing additionally for the projection $\mathbf{x}_\mathbf{w}$.

From an application point of view, extension to shells and rods seem also very attractive in order to further extend this example-based control paradigm to larger number of deformable objects such as cloth or hair. The strainspace definition for solid FEM can be generalized straightforwardly and first experiments already showed satisfactory results. In this respect, also its extension to the presented elaston-based simulation framework is attractive, being not too difficult based on the nonlinear extension of Jeronimo Bayer's master thesis [2011]. But also for other effects such as plastic deformation or fracturing it could be interesting to adapt the example-based simulation idea. For example, forcing the rest state deformation during plastic flow to stay within an example-manifold would allow restricting possible simulation outcomes in a simple manner, being valuable e.g. in interactive settings like car racing games, where the plastic deformation on impacts and crashes could be controlled. Also when fracturing ductile material, an example-based approach could be used to steer the deformation, but could also be extended to determine the point of rupture along each such deformation.

APPENDIX A

Notation and Glossary

A.1 Notation

This section reviews the notation employed throughout the thesis.

A.1.1 Operators

$f_{,i} = \frac{\partial f}{\partial x_i}$	First derivative with respect to the i-th argument
$f_{,ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$	Second derivative with respect to the i-th and j-th argument
∇f	gradient of f
Δf	Laplacian of f
H_f	Hessian of f
$\dot{f}$	First time derivative
$\ddot{f}$	Second time derivative
$\cdot$	Vector dot product
$\times$	Vector cross product
$:$	Tensor double contraction
det	Determinant

A.1.2 Spaces

$\mathbb{N}$	The set of natural numbers
$\mathbb{R}$	The set of real numbers
$\mathbb{R}^n$	The n-dimensional real vector space
$\mathbb{R}^{n\times m}$	Space of real-valued $m \times n$-matrices
$\Omega \subset \mathbb{R}^3$	Object domain
$\Gamma \subset \Omega$	Object domain boundary
$\Gamma_{BC} \subset \Gamma$	Boundary condition domain
V	Infinite dimensional function space
$V_N \subset V$	Finite dimensional subspace of V
$L^2(\Omega)$	Space of square-integrable functions over Ω
$C^n(\Omega)$	Space of n-times continuously differentiable functions over Ω
$H^n(\Omega)$	Sobolev space of order n over Ω

A.1.3 General Notation

δ_{ij}	Kronecker delta
t	Time
$\bar{\mathbf{x}} \in \mathbb{R}^3$	Undeformed configuration
$\bar{\mathbf{x}}(\boldsymbol{\theta}) : \Omega \to \mathbb{R}^3$	Undeformed configuration in curvilinear coordinates
$\mathbf{x}(\bar{\mathbf{x}}) : \Omega \to \mathbb{R}^3$	Deformed configuration in Cartesian coordinates
$\mathbf{x}(\boldsymbol{\theta}) : \Omega \to \mathbb{R}^3$	Deformed configuration in curvilinear coordinates
$\mathbf{u}(\bar{\mathbf{x}}) : \Omega \to \mathbb{R}^3$	Displacement in Cartesian coordinates
$\mathbf{u}(\boldsymbol{\theta}) : \Omega \to \mathbb{R}^3$	Displacement in curvilinear coordinates
$\mathbf{u}_{BC}(\bar{\mathbf{x}}) : \Omega \to \mathbb{R}^3$	Prescribed displacement as boundary condition
$\mathbf{v} : \Omega \to \mathbb{R}^3$	Velocity in Cartesian coordinates
$\boldsymbol{\epsilon} \in \mathbb{R}^{3\times3}$	Cauchy strain
$\boldsymbol{\alpha} \in \mathbb{R}^{3\times3}$	Linearized membrane strain
$\boldsymbol{\beta} \in \mathbb{R}^{3\times3}$	Linearized bending strain
$\boldsymbol{\sigma} \in \mathbb{R}^{3\times3}$	Cauchy stress
$\mathbf{n} \in \mathbb{R}^3$	Normal vector

$\mathbf{f} \in \mathbb{R}^3$ ..... Force vector
$W : H^1(\Omega) \to \mathbb{R}$ ..... Energy potential
$\mathbf{f}_{int} : \Omega \to \mathbb{R}^3$ ..... Internal force vector
$W_{int} : H^1(\Omega) \to \mathbb{R}$ ..... Internal potential energy
$\mathbf{f}_{ext} : \Omega \to \mathbb{R}^3$ ..... External force vector
$W_{BC} : L^2(\Omega) \to \mathbb{R}$ ..... Boundary condition penalty potential
$W_{plane} : L^2(\Omega) \to \mathbb{R}$ ..... Plane collisions penalty potential
$W_{krod} : H^2(\Omega) \to \mathbb{R}$ ..... Kirchhoff rod potential
$W_e : H^1(\Omega) \to \mathbb{R}$ ..... Elemental energy potential
$W_{tot} : H^1(\Omega) \to \mathbb{R}$ ..... Total potential energy
$\rho : \Omega \to \mathbb{R}$ ..... Density
$\mathbf{C} \in \mathbb{R}^{3\times3\times3\times3}$ ..... Fourth-order Hookean material tensor
$E \in \mathbb{R}$ ..... Young modulus
$\nu \in \mathbb{R}$ ..... Poisson's ratio
$\mathbf{R} \in \mathbb{R}^{3\times3}$ ..... Rotation matrix
$\mathbf{F} \in \mathbb{R}^{3\times3}$ ..... Deformation gradient
$\mathbf{C} \in \mathbb{R}^{3\times3}$ ..... Right Cauchy-Green deformation tensor
$\mathbf{E} \in \mathbb{R}^{3\times3}$ ..... Green strain tensor
$\Psi : H^1(\Omega) \to \mathbb{R}$ ..... Energy density
$h_i \in \mathbb{R}$ ..... Material thickness in direction i
$a : H^1 \times H^1 \to \mathbb{R}$ ..... Spd bilinear form
$f : H^1 \to \mathbb{R}$ ..... Linear form
$N_i : \Omega \to \mathbb{R}$ ..... Basis function associated to DOF i
$\mathbf{M} \in \mathbb{R}^{3N\times3N}$ ..... Mass matrix
$\mathbf{K} \in \mathbb{R}^{3N\times3N}$ ..... Stiffness matrix
e ..... Element domain
h ..... Timestep size

A.1.4 Tayloring Solution Subspaces

$\psi(\bar{\mathbf{x}}) : \mathbb{R}^3 \to \mathbb{R}$ ..... Radial basis function kernel
$\mathbf{k}_i \in \mathbb{R}^3$ ..... Kernel centers
$b_i(\bar{\mathbf{x}}) : \Omega \to \mathbb{R}$ ..... Boundary value function for DOF i
$l(\bar{\mathbf{x}}) : \mathbb{R}^3 \to \mathbb{R}$ ..... Arbitrary linear scalar function
$a_i \in \mathbb{R}$ ..... Polynom coefficients
$\mathbf{c}_i \in \mathbb{R}^3$ ..... Collocation points
$\xi \in \mathbb{R}$ ..... Kernel offset distance
$\mathbf{H}_e(\bar{\mathbf{x}}) \in \mathbb{R}^{3\times 3k}$ ..... Per element basis function matrix
$\mathbf{B}_e(\bar{\mathbf{x}}) \in \mathbb{R}^{6\times 3k}$ ..... Per element gradient matrix
$w(\bar{\mathbf{x}}) : \mathbb{R}^3 \to \mathbb{R}$ ..... Weight function
$\mathbf{p}(\bar{\mathbf{x}}) : \mathbb{R}^3 \to \mathbb{R}^d$ ..... Monomial vector
$J : C^\infty(\Omega) \to \mathbb{R}$ ..... MLS cost function
$\mathbf{G}(\bar{\mathbf{x}}) : \mathbb{R}^3 \to \mathbb{R}^{d\times d}$ ..... Moment matrix
$M_j : \mathbb{R}^3 \to \mathbb{R}$ ..... Basis function associated to face j
$s_j \in \mathbb{R}$ ..... Green coordinates scaling factor
$G(\bar{\mathbf{x}}, \bar{\mathbf{x}}') : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$ .Green's function/fundamental solution
$\Gamma_i(\bar{\mathbf{x}}') : \mathbb{R}^3 \to \mathbb{R}$ ..... Hat function on triangular mesh
$\mathbf{H} \in \mathbb{R}^{3M\times 3N}$ ..... Stiffness matrix relating normals to nodal forces

A.1.5 Unifying Resultant-based Models

$W_e : H^2(\Omega) \to \mathbb{R}$ ..... Potential energy per elaston
$\mathcal{M}$ ..... Material point set
$\mathbf{m}_i$ ..... Material point centers
$\mathcal{M}_i$ ..... Partitioned material point set
r_i ..... Material point radii
$\mathbf{e}_i$ ..... Elaston centers
λ_i ..... Eigenvalues
$\mathbf{A}_i \in \mathbb{R}^{6\times 3}$ ..... Membrane strain matrix
$\mathbf{B}_i \in \mathbb{R}^{6\times 3}$ ..... Bending strain matrix

$(\cdot)^n$ ..... Measures in the new domain
$\boldsymbol{\alpha}_e : \Omega \to \mathbb{R}^{3\times3}$ ..... Effective elastic membrane strain
$\boldsymbol{\beta}_e^k : \Omega \to \mathbb{R}^{3\times3}$ ..... Effective elastic bending strain
$\boldsymbol{\alpha}_g : \Omega \to \mathbb{R}^{3\times3}$ ..... Geometric membrane strain
$\boldsymbol{\beta}_g^k : \Omega \to \mathbb{R}^{3\times3}$ ..... Geometric bending strain
$\boldsymbol{\alpha}_p : \Omega \to \mathbb{R}^{3\times3}$ ..... Plastic membrane strain
$\boldsymbol{\beta}_p^k : \Omega \to \mathbb{R}^{3\times3}$ ..... Plastic bending strain
$\mathbf{G} \in \mathbb{R}^{3n\times3n}$ ..... Gram matrix
$\mathbf{T} \in \mathbb{R}^{3\times3}$ ..... Elaston local frame in undeformed configuration
$\mathbf{M} \in \mathbb{R}^{3\times3}$ ..... Transformation matrix between old and new rest state

A.1.6 Art-directable Elastic Potentials

$\mathbf{E}(\mathbf{x}) : \mathbb{R}^{3n} \to \mathbb{R}^{6m}$ ..... Configuration to strain space mapping
$\mathbf{x}_i \in \mathbb{R}^{3n}$ ..... Example poses configuration vectors
$\mathbf{E}_i \in \mathbb{R}^{6m}$ ..... Example poses strain space vectors
$\mathbf{w} \in \mathbb{R}^{d}$ ..... Example weight vector
$\mathbf{x}_\mathbf{w} \in \mathbb{R}^{3n}$ ..... Projected example space rest state configuration
$\mathcal{F}$ ..... Realizable manifold
$\mathcal{E}$ ..... Example manifold
$H(\mathbf{x}_n, \mathbf{x}_\mathbf{w}, \mathbf{w})$ ..... Implicit Euler objective function

A.2 Glossary

BC ..... Boundary condition
BEM ..... Boundary element method
CM ..... Continuum mechanics
CSE ..... Constant strain elements
DOF ..... Degree of freedom
EFG ..... Element-free Galerkin
FDM ..... Finite difference method
FE ..... Finite element
FEM ..... Finite element method
GC ..... Green coordinates
GMLS ..... Generalized moving least squares
HC ..... Harmonic Coordinates
MLS ..... Moving least squares
MVC ..... Mean value coordinates
MFS ..... Method of fundamental solutions
PDE ..... Partial differential equation
PU ..... Partition of unity
spd ..... Symmetric positive definite
SPH ..... Smoothed-particle hydrodynamics

Bibliography

[Adams et al., 2008] Bart Adams, Maks Ovsjanikov, Michael Wand, Hans-Peter Seidel, and Leonidas J. Guibas. Meshless modeling of deformable shapes and their motion. In *Proc. of Symp. on Computer Animation*, pages 77–86, 2008.

[Adams et al., 2009] Bart Adams, Martin Wicke, Maks Ovsjanikov, Michael Wand, Hans-Peter Seidel, and Leonidas Guibas. Meshless shape and motion design for multiple deformable objects. *Comput. Graphics Forum*, 2009.

[Alexa et al., 2000] Marc Alexa, Daniel Cohen-Or, and David Levin. As-Rigid-As-Possible Shape Interpolation. In *Proc. ACM SIGGRAPH*, pages 157–164, 2000.

[Alliez et al., 2005] Pierre Alliez, David Cohen-Steiner, Mariette Yvinec, and Mathieu Desbrun. Variational tetrahedral meshing. *ACM Trans. on Graphics*, 24(3):617–625, 2005.

[An et al., 2008] Steven S. An, Theodore Kim, and Doug L. James. Optimizing cubature for efficient integration of subspace deformations. *ACM Trans. on Graphics*, 27(5):164:1–164:11, 2008.

[Areias and Belytschko, 2005] Pedro M. A. Areias and Ted Belytschko. Analysis of three-dimensional crack initiation and propagation using the extended finite element method. *International Journal for Numerical Methods in Engineering*, 63(5):760–788, 2005.

[Atluri et al., 1999] S.Ñ. Atluri, J.Ỹ. Cho, and H.-G. Kim. Analysis of thin beams, using the meshless local Petrov-Galerkin method, with generalized moving least squares interpolations. *Computational Mechanics*, (24):334–347, 1999.

[Bao et al., 2007] Zhaosheng Bao, Jeong-Mo Hong, Joseph Teran, and Ronald Fedkiw. Fracturing rigid materials. *IEEE Transactions on Visualization and Computer Graphics*, 13:370–378, March 2007.

[Baraff and Witkin, 1998] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proc. of ACM SIGGRAPH*, pages 43–54, 1998.

[Baran et al., 2009] Ilya Baran, Daniel Vlasic, Eitan Grinspun, and Jovan Popović. Semantic deformation transfer. In *ACM Trans. on Graphics*, pages 1–6, New York, NY, USA, 2009. ACM.

[Barbič and James, 2005] Jernej Barbič and Doug L. James. Real-time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Trans. on Graphics*, 24(3):982–990, August 2005.

[Barbič and Popović, 2008] Jernej Barbič and Jovan Popović. Real-time control of physically based simulations using gentle forces. *ACM Trans. on Graphics*, 27:163:1–163:10, December 2008.

[Barbič and Zhao, 2011] Jernej Barbič and Yili Zhao. Real-time large-deformation substructuring. *ACM Trans. on Graphics*, 30(4):91:1–91:7, 2011.

[Barbič et al., 2009] Jernej Barbič, Marco da Silva, and Jovan Popović. Deformable object animation using reduced optimal control. *ACM Trans. on Graphics*, 28(3), 2009.

[Bargteil et al., 2007] Adam W. Bargteil, Chris Wojtan, Jessica K. Hodgins, and Greg Turk. A finite element method for animating large viscoplastic flow. *ACM Trans. on Graphics*, 26(3):16.1–16.8, 2007.

[Bathe, 1995] K.-J. Bathe. *Finite Element Procedures*. Prentice Hall, 1995.

[Bayer, 2011] Jeronimo Bayer. *Nonlinear Elastons, Master Thesis*. Institute of Visual Computing, ETH Zurich, 2011.

[Belytschko et al., 1994] T. Belytschko, Y. Y. Lu, and L. Gu. Element-free galerkin methods. *International Journal for Numerical Methods in Engineering*, (37):229–256, 1994.

[Ben-Chen et al., 2009] Mirela Ben-Chen, Ofir Weber, and Craig Gotsman. Variational harmonic maps for space deformation. *ACM Trans. on Graphics*, 28:34:1–34:11, July 2009.

[Bergou et al., 2006] Miklós Bergou, Max Wardetzky, David Harmon, Denis Zorin, and Eitan Grinspun. A Quadratic Bending Model for Inextensible Surfaces. In *Proc. of Symp. on Geometry Processing*, pages 227–230, Jun 2006.

[Bergou et al., 2007] Miklós Bergou, Saurabh Mathur, Max Wardetzky, and Eitan Grinspun. Tracks: toward directable thin shells. *ACM Trans. on Graphics*, 26, July 2007.

[Bergou et al., 2008] Miklós Bergou, Max Wardetzky, Stephen Robinson, Basile Audoly, and Eitan Grinspun. Discrete elastic rods. *ACM Trans. on Graphics*, 27(3):63:1–63:12, 2008.

[Bergou et al., 2010] Miklós Bergou, Basile Audoly, Etienne Vouga, Max Wardetzky, and Eitan Grinspun. Discrete Viscous Threads. *ACM Trans. on Graphics*, 2010.

[Bertails et al., 2006] Florence Bertails, Basile Audoly, Marie-Paule Cani, Bernard Querleux, Frédéric Leroy, and Jean-Luc Lévêque. Super-helices for predicting the dynamics of natural hair. In *ACM Trans. on Graphics*, August 2006.

[Bickel et al., 2009] Bernd Bickel, Moritz Bächer, Miguel A. Otaduy, Wojciech Matusik, Hanspeter Pfister, and Markus Gross. Capture and modeling of non-linear heterogeneous soft tissue. In *ACM Trans. on Graphics*, pages 1–9, 2009.

[Bielser and Gross, 2000] Daniel Bielser and Markus Gross. Interactive simulation of surgical cuts. In *Proc. of Pacific Graphics*, pages 116–125, 2000.

[Bielser et al., 1999] D. Bielser, V.A. Maiwald, and M.H. Gross. Interactive cuts through 3-dimensional soft tissue. *Comput. Graphics Forum (Proc. Eurographics)*, 18(3):31–38, 1999.

[Bielser et al., 2003] Daniel Bielser, P. Glardon, Matthias Teschner, and Markus Gross. A state machine for real-time cutting of tetrahedral meshes. In *Proc. of Pacific Graphics*, pages 377–386, 2003.

[Bonet and Wood, 1997] J. Bonet and R. D. Wood. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press, 1997.

[Botsch and Sorkine, 2008] Mario Botsch and Olga Sorkine. On linear variational surface deformation methods. *IEEE Trans. on Visualization and Computer Graphics*, 14(1):213–230, 2008.

[Botsch et al., 2006] Mario Botsch, Mark Pauly, Markus Gross, and Leif Kobbelt. PriMo: coupled prisms for intuitive surface modeling. In *Proc. of Symp. on Geometry Processing '06*, pages 11–20, 2006.

[Botsch et al., 2007] M. Botsch, M. Pauly, M. Wicke, and M. Gross. Adaptive space deformations based on rigid cells. *Comput. Graphics Forum (Proc. Eurographics)*, 26(3):339–347, 2007.

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

[Bridson et al., 2003] R. Bridson, S. Marino, and R. Fedkiw. Simulation of clothing with folds and wrinkles. In *Proc. of Symp. on Computer Animation*, pages 28–36, 2003.

[Bridson, 2008] Robert Bridson. *Fluid Simulation for Computer Graphics*. A K Peters, 2008.

[Capell et al., 2002] Steve Capell, Seth Green, Brian Curless, Tom Duchamp, and Zoran Popović. A multiresolution framework for dynamic deformations. In *Proc. of Symp. on Computer Animation'02*, pages 41–47, 2002.

[Carlson et al., 2002] Mark Carlson, Peter J. Mucha, R. Brooks Van Horn, III, and Greg Turk. Melting and flowing. In *Proc. of Symp. on Computer Animation*, pages 167–174, New York, NY, USA, 2002. ACM.

[Chao et al., 2010] Isaac Chao, Ulrich Pinkall, Patrick Sanan, and Peter Schröder. A simple geometric model for elastic deformations. In *ACM Trans. on Graphics*, pages 38:1–38:6, New York, NY, USA, 2010. ACM.

[Chapelle and Bathe, 2003] Dominique Chapelle and Klaus-Jrgen Bathe. *The Finite Element Analysis of Shells: Fundamentals*. Springer, 2003.

[Chen et al., 2008] 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):1–14, 2008.

[Choi and Ko, 2002] Kwang-Jin Choi and Hyeong-Seok Ko. Stable but responsive cloth. *Proc. of ACM SIGGRAPH*, 21(3):604–611, 2002.

[Choi and Ko, 2005] Min Gyu Choi and Hyeong-Seok Ko. Modal warping: Real-time simulation of large rotational deformation and manipulation. *IEEE Transactions on Visualization and Computer Graphics*, pages 11:91–11:101, 2005.

[Chung, 1996] T. J. Chung. *Applied Continuum Mechanics*. Cambridge University Press, New York, 1996.

[Cirak et al., 2000] Fehmi Cirak, Michael Ortiz, and Peter Schröder. Subdivision surfaces: A new paradigm for thin-shell finite-element analysis. *Int. J. Numer. Methods Eng.*, 47(12):2039–2072, 2000.

[Clavet et al., 2005] Simon Clavet, Philippe Beaudoin, and Pierre Poulin. Particle-based viscoelastic fluid simulation. In *Proc. of Symp. on Computer Animation*, pages 219–228, 2005.

[Daux et al., 2000] Christophe Daux, Nicolas Mos, John Dolbow, Natarajan Sukumar, and Ted Belytschko. Arbitrary branched and intersecting cracks with the extended finite element method. *International Journal for Numerical Methods in Engineering*, 48(12):1741–1760, 2000.

[Debunne et al., 2001] Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H. Barr. Dynamic real-time deformations using space and time adaptive sampling. In *Proc. of ACM SIGGRAPH*, pages 31–36, 2001.

[Delingette, 2008] Herve Delingette. Triangular springs for modeling nonlinear membranes. *IEEE Trans. on Visualization and Computer Graphics*, 14(2):329–341, 2008.

[Desbrun and paule Gascuel, 1996] Mathieu Desbrun and Marie paule Gascuel. Smoothed particles: A new paradigm for animating highly deformable bodies. In *In Computer Animation and Simulation 96 (Proceedings of EG Workshop on Animation and Simulation*, pages 61–76. Springer-Verlag, 1996.

[Desbrun et al., 2008] Mathieu Desbrun, Eva Kanso, and Yiying Tong. Discrete differential forms for computational modeling. In *ACM SIGGRAPH ASIA 2008 courses*, SIGGRAPH Asia '08, pages 15:1–15:17, New York, NY, USA, 2008. ACM.

[Duchon, 1977] J. Duchon. Spline minimizing rotation-invariant semi-norms in Sobolev spaces. In W. Schempp and K. Zeller, editors, *Constructive Theory of Functions of Several Variables*, number 571 in Lecture Notes in Mathematics, pages 85–100. Springer Verlag, 1977.

[English and Bridson, 2008] Elliot English and Robert Bridson. Animating developable surfaces using nonconforming elements. *Proc. of ACM SIGGRAPH*, 27(3):1–5, 2008.

[Etzmuss et al., 2003] Olaf Etzmuss, Joachim Gross, and Wolfgang Strasser. Deriving a particle system from continuum mechanics for the animation of deformable objects. *IEEE Trans. on Visualization and Computer Graphics*, 9:538–550, 2003.

[Fairweather and Karageorghis, 1998] G. Fairweather and A. Karageorghis. The method of fundamental solutions for elliptic boundary value problems. *Advances in Computational Mathematics*, 9(1–2):69–95, 1998.

[Faloutsos et al., 1997] P. Faloutsos, M. van de Panne, and D. Terzopoulos. Dynamic free-form deformations for animation synthesis. *IEEE Transactions on Visualization and Computer Graphics*, 3(3):201–214, 1997.

[Faure et al., 2011] François Faure, Benjamin Gilles, Guillaume Bousquet, and Dinesh Pai. Sparse meshless models of complex deformable solids. *ACM Trans. on Graphics*, 2011.

[Floater et al., 2005] Michael S. Floater, G. Kos, and M. Reimers. Mean value coordinates in 3D. *Computer Aided Geometric Design*, 22(7):623–631, 2005.

[Floater, 2003] M. S. Floater. Mean value coordinates. *Computer Aided Geometric Design*, 20(1):19–27, 2003.

[Foster and Metaxas, 1997] Nick Foster and Dimitris Metaxas. Modeling the motion of a hot, turbulent gas. In *Proc. of ACM SIGGRAPH*, pages 181–188, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[Fries and Matthies, 2004] T. P. Fries and H. G. Matthies. Classification and overview of meshfree methods. Informatikbericht 2003-03, revised 2004, Institute of Scientific Computing, Technical University Braunschweig, 2004.

[Gain and Bechmann, 2008] James Gain and Dominique Bechmann. A survey of spatial deformation from a user-centered perspective. *ACM Trans. on Graphics*, 27:107:1–107:21, November 2008.

[Garg et al., 2007] Akash Garg, Eitan Grinspun, Max Wardetzky, and Denis Zorin. Cubic shells. In *Proc. of Symp. on Computer Animation*, pages 91–98, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[Gelfand and Fomin, 2000] I. M. Gelfand and S. V. Fomin. *Calculus of Variations*. Dover Publ., 2000.

[Georgii and Westermann, 2008] Joachim Georgii and Rüdiger Westermann. Corotated finite elements made fast and stable. In *Proceedings of the 5th Workshop On Virtual Reality Interaction and Physical Simulation*, pages 11–19, 2008.

[Gerszewski et al., 2009] Dan Gerszewski, Haimasree Bhattacharya, and Adam W. Bargteil. A point-based method for animating elastoplastic solids. In *Proc. of Symp. on Computer Animation*, pages 133–138, 2009.

[Gibson and Mirtich, 1997] S. F. Gibson and B. Mirtich. A survey of deformable models in computer graphics. Technical report, Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA, 1997.

[Gilles et al., 2011] Benjamin Gilles, Guillaume Bousquet, François Faure, and Dinesh K. Pai. Frame-based elastic models. *ACM Trans. on Graphics*, 2011.

[Gingold and Monaghan, 1977] R. A. Gingold and J. J. Monaghan. Smoothed particle hydrodynamics - theory and application to non-spherical stars. *Royal Astronomical Society, Monthly Notices*, 181:375–389, nov 1977.

[Goktekin et al., 2004] Tolga G. Goktekin, Adam W. Bargteil, and James F. O'Brien. A method for animating viscoelastic fluids. *ACM Trans. on Graphics*, 23(3):463–468, 2004.

[Goldenthal et al., 2007] Rony Goldenthal, David Harmon, Raanan Fattal, Michel Bercovier, and Eitan Grinspun. Efficient simulation of inextensible cloth. *Proc. of ACM SIGGRAPH*, 26(3):49:1–49:7, 2007.

[Golub and Loan, 1989] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1989.

[Grinspun et al., 2002] Eitan Grinspun, Petr Krysl, and Peter Schröder. CHARMS: A simple framework for adaptive simulation. *Proc. of ACM SIGGRAPH*, 21(3):281–290, 2002.

[Grinspun et al., 2003] Eitan Grinspun, Anil N. Hirani, Mathieu Desbrun, and Peter Schröder. Discrete shells. In *Proc. of Symp. on Computer Animation*, pages 62–67, 2003.

[Grinspun, 2003] Eitan Grinspun. *The Basis Refinement Method, PhD Thesis*. Caltech, 2003.

[Gross and Pfister, 2007] Markus Gross and Hanspeter Pfister. *Point-Based Graphics*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007.

[Guo et al., 2006] Xiaohu Guo, Xin Li, Yunfan Bao, Xianfeng Gu, and Hong Qin. Meshless thin-shell simulation based on global conformal parameterization. *IEEE Trans. on Visualization and Computer Graphics*, 12(3):375–385, 2006.

[Hadap et al., 2007] Sunil Hadap, Marie-Paule Cani, Ming Lin, Tae-Yong Kim, Florence Bertails, Steve Marschner, Kelly Ward, and Zoran Kačić-Alesić. *Strands and hair: modeling, animation, and rendering*, pages 1–150. ACM SIGGRAPH 2007 Courses, 2007.

[Hauth and Strasser, 2004] M. Hauth and W. Strasser. Corotational simulation of deformable solids. In *Proc. of WSCG*, pages 137–145, 2004.

[Hrennikoff, 1941] A. Hrennikoff. Solution of problems of elasticity by the framework method. *ASME J. Appl. Mech.*, (8):A619–A715, 1941.

[Huber, 2009] Christoph Huber. *FEM-based Elasticity Using Quasi-conformal Deformations, Master Thesis*. Institute of Visual Computing, ETH Zurich, 2009.

[Hughes, 2000] Thomas J. R. Hughes. *The Finite Element Method. Linear Static and Dynamic Finite Element Analysis*. Dover Publications, 2000.

[Irving et al., 2004] G. Irving, J. Teran, and R. Fedkiw. Invertible finite elements for robust simulation of large deformation. In *Proc. of Symp. on Computer Animation*, pages 131–140, 2004.

[Irving et al., 2006] G. Irving, J. Teran, and R. Fedkiw. Tetrahedral and hexahedral invertible finite elements. *Graphical Models*, 68(2):66–89, 2006.

[Irving et al., 2007] Geoffrey Irving, Craig Schroeder, and Ronald Fedkiw. Volume conserving finite element simulations of deformable models. *ACM Trans. on Graphics*, 26(3):13.1–13.6, 2007.

[James and Fatahalian, 2003] Doug L. James and Kayvon Fatahalian. Precomputing interactive dynamic deformable scenes. *ACM Trans. on Graphics*, 22(3):879–887, July 2003.

[James and Pai, 1999] Doug L. James and Dinesh K. Pai. Artdefo: accurate real time deformable objects. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 65–72, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[James et al., 2004] D. James, J. Barbič, and C. Twigg. Squashing cubes: Automating deformable model construction for graphics. In *Proc. of SIGGRAPH '04 Sketches and Applications*, 2004.

[Jeřábková and Kuhlen, 2009] Lenka Jeřábková and Torsten Kuhlen. Stable cutting of deformable objects in virtual environments using xfem. *IEEE Comput. Graph. Appl.*, 29:61–71, March 2009.

[Joshi et al., 2007] Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. Harmonic coordinates for character articulation. *ACM Trans. on Graphics*, 26(3), 2007.

[Ju et al., 2005a] T. Ju, S. Schaefer, J. Warren, and M.Desbrun. Geometric construction of coordinates for convex polyhedra using polar duals. In *Proc. of Symp. on Geometry Processing*, pages 181–186, 2005.

[Ju et al., 2005b] Tao Ju, Scott Schaefer, and Joe Warren. Mean value coordinates for closed triangular meshes. *Proc. of ACM SIGGRAPH*, 24(3):561–566, 2005.

[Kaldor et al., 2008] Jonathan M. Kaldor, Doug L. James, and Steve Marschner. Simulating knitted cloth at the yarn level. *ACM Trans. on Graphics*, 27:65:1–65:9, August 2008.

[Kaldor et al., 2010] Jonathan M. Kaldor, Doug L. James, and Steve Marschner. Efficient yarn-based cloth with adaptive contact linearization. *ACM Trans. on Graphics*, 29:105:1–105:10, July 2010.

[Kaufmann et al., 2008] P. Kaufmann, S. Martin, M. Botsch, and M. Gross. Flexible simulation of deformable models using discontinuous Galerkin FEM. In *Proc. of Symp. on Computer Animation*, pages 105–115, 2008.

[Kaufmann et al., 2009a] P. Kaufmann, S. Martin, M. Botsch, and M. Gross. Implementation of discontinuous Galerkin Kirchhoff-Love shells. Technical Report no. 622, Department of Computer Science, ETH Zurich, 2009.

[Kaufmann et al., 2009b] Peter Kaufmann, Sebastian Martin, Mario Botsch, Eitan Grinspun, and Markus Gross. Enrichment textures for detailed cutting of shells. *ACM Trans. on Graphics*, 28(3):50:1–50:10, August 2009.

[Keiser et al., 2005] R. Keiser, B. Adams, D. Gasser, P. Bazzi, P. Dutre, and M. Gross. A unified lagrangian approach to solid-fluid animation. *Proceedings Eurographics/IEEE VGTC Symposium Point-Based Graphics*, 0:125–148, 2005.

[Kharevych et al., 2009] Lily Kharevych, Patrick Mullen, Houman Owhadi, and Mathieu Desbrun. Numerical coarsening of inhomogeneous elastic materials. *ACM Trans. Graph.*, 28:51:1–51:8, July 2009.

[Kikuuwe et al., 2009] Ryo Kikuuwe, Hiroaki Tabuchi, and Motoji Yamamoto. An edge-based computationally efficient formulation of Saint Venant-Kirchhoff tetrahedral finite elements. *ACM Trans. on Graphics*, 28(1):1–13, 2009.

[Kilian et al., 2007] Martin Kilian, Niloy J. Mitra, and Helmut Pottmann. Geometric modeling in shape space. In *ACM Trans. on Graphics*, page 64, New York, NY, USA, 2007. ACM.

[Kim and James, 2009] Theodore Kim and Doug L. James. Skipping steps in deformable simulation with online model reduction. *ACM Trans. on Graphics*, 28:123:1–123:9, December 2009.

[Kim et al., 2008] Theodore Kim, Nils Thürey, Doug James, and Markus Gross. Wavelet turbulence for fluid simulation. *ACM Trans. on Graphics*, 27:50:1–50:6, August 2008.

[Kircher and Garland, 2006] Scott Kircher and Michael Garland. Editing arbitrarily deforming surface animations. *ACM Trans. on Graphics*, 25:1098–1107, July 2006.

[Kondo et al., 2005] Ryo Kondo, Takashi Kanai, and Ken-ichi Anjyo. Directable animation of elastic objects. In *Proc. of Symp. on Computer Animation*, pages 127–134. ACM, 2005.

[Kreyszig, 2005] E. Kreyszig. *Advanced Engineering Mathematics, 9th edition*. Wiley, 2005.

[Krysl and Belytschko, 1996] Petr Krysl and Ted Belytschko. Analysis of thin shells by the element-free galerkin method. *International Journal of Solids and Structures*, 33(20-22):3057 – 3080, 1996.

[Krysl, 2005] Petr Krysl. *A Pragmatic Introduction to the Finite Element Method for Thermal and Stress Analysis*. Pressure Cooker Press, San Diego, 2005.

[Lai et al., 1978] W. Michael Lai, David Rubin, and Erhard Krempl. *Introduction to continuum mechanics*. Pergamon Press, Oxford ; New York :, rev. ed. in si/metric units. edition, 1978.

[Lancaster and Salkauskas, 1981] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, 1981.

[Landau and Lifshitz, 2003] L. D. Landau and E. M. Lifshitz. *Mechanics*. Pergamon Press, 2003.

[Langer and Singer, 1996] Joel Langer and David Singer. Lagrangian aspects of the kirchhoff elastic rod. *SIAM Review*, 38(4):pp. 605–618, 1996.

[Lasseter, 1987] John Lasseter. Principles of traditional animation applied to 3d computer animation. *SIGGRAPH Comput. Graph.*, 21:35–44, August 1987.

[Levin et al., 2011] David I. W. Levin, Joshua Litven, Garrett L. Jones, Shinjiro Sueda, and Dinesh K. Pai. Eulerian solid simulation with contact. *ACM Trans. on Graphics*, 2011.

[Li et al., 2004] Shaofan Li, Hongsheng Lu, Weimin Han, Wing Kam Liu, and Daniel C. Simkins. Reproducing kernel element method, part ii: Globally conforming i^m/c^n hierarchies. *Comput. Methods Appl. Mech. Engrg.*, 29(193):963–987, 2004.

[Li et al., 2007] Xin Li, Xiaohu Guo, Hongyu Wang, Ying He, Xianfeng Gu, and Hong Qin. Harmonic volumetric mapping for solid modeling applications. In *Proceedings of symposium on Solid and physical modeling*, pages 109–120, 2007.

[Li et al., 2010] Hao Li, Thibaut Weise, and Mark Pauly. Example-based facial rigging. In *ACM Trans. on Graphics*, pages 32:1–32:6. ACM, 2010.

[Lipman et al., 2005] Yaron Lipman, Olga Sorkine, David Levin, and Daniel Cohen-Or. Linear rotation-invariant coordinates for meshes. *ACM Trans. on Graphics*, 24(3):479–487, 2005.

[Lipman et al., 2008] Yaron Lipman, David Levin, and Daniel Cohen-Or. Green coordinates. In *ACM Trans. on Graphics*, volume 27, pages 1–10, 2008.

[Lloyd et al., 2007] Bryn Lloyd, Gábor Székely, and Matthias Harders. Identification of spring parameters for deformable object simulation. *IEEE Trans. on Visualization and Computer Graphics*, 13:1081–1094, 2007.

[Lloyd, 1957] S. Lloyd. Least squares quantization in PCM. Technical report, Tech. rep., Bell Telephone Laboratories, Murray Hill, NJ, 1957.

[Losasso et al., 2006] Frank Losasso, Tamar Shinar, Andrew Selle, and Ronald Fedkiw. Multiple interacting liquids. *ACM Trans. on Graphics*, 25:812–819, July 2006.

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

[Manukyan, 2011] Liana Manukyan. *Meshless Simulation of Thin Shells, Master Thesis*. Institute of Visual Computing, ETH Zurich, 2011.

[Martin et al., 2008] S. Martin, P. Kaufmann, M. Botsch, M. Wicke, and M. Gross. Polyhedral finite elements using harmonic basis functions. *Comput. Graphics Forum*, 27(5):1521–1529, 2008.

[Martin et al., 2009] Sebastian Martin, Christoph Huber, Peter Kaufmann, and Markus Gross. Shape-preserving animation of deformable objects. *Proceedings of Vision, Modeling, and Visualization (VMV) (Braunschweig, Germany, November 16-18, 2009)*, pages 65–72, 2009.

[Martin et al., 2010] Sebastian Martin, Peter Kaufmann, Mario Botsch, Eitan Grinspun, and Markus Gross. Unified simulation of elastic rods, shells, and solids. *ACM Trans. on Graphics*, 29(3):39:1–39:10, 2010.

[Martin et al., 2011] Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus Gross. Example-based elastic materials. *ACM Trans. on Graphics*, 30(4):72:1–72:8, 2011.

[Mazza et al., 2005] E. Mazza, O. Papes, M. B. Rubin, S. R. Bodner, and N. S. Binur. Nonlinear elastic-viscoplastic constitutive equations for aging facial tissues. *Biomech Model Mechanobiol.*, 4(2):178 – 189, 2005.

[McAdams et al., 2009] Aleka McAdams, Andrew Selle, Kelly Ward, Eftychios Sifakis, and Joseph Teran. Detail preserving continuum simulation of straight hair. *ACM Trans. Graph.*, 28:62:1–62:6, July 2009.

[McAdams et al., 2011] A. McAdams, Y. Zhu, A. Selle, R. Tamstorf, M. Embrey, J. Teran, and E. Sifakis. Efficient elasticity for character skinning with contact and collisions. *ACM Trans. on Graphics*, 2011.

[McNamara et al., 2004] Antoine McNamara, Adrien Treuille, Zoran Popović, and Jos Stam. Fluid control using the adjoint method. *ACM Trans. on Graphics*, 23:449–456, August 2004.

[Meyer et al., 2002] Mark Meyer, Haeyoung Lee, Alan Barr, and Mathieu Desbrun. Generalized barycentric coordinates on irregular polygons. *Journal of Graphics Tools*, 7:13–22, 2002.

[Mezger et al., 2008] Johannes Mezger, Bernhard Thomaszewski, Simon Pabst, and Wolfgang Straßer. Interactive physically-based shape editing. In *Proc. of ACM Symp. on Solid and Physical Modeling*, pages 79–89, 2008.

[Milliron et al., 2002] Tim Milliron, Robert J. Jensen, Ronen Barzel, and Adam Finkelstein. A framework for geometric warps and deformations. *ACM Trans. on Graphics*, 21(1):20–51, 2002.

[Molino et al., 2004] Neil Molino, Zhaosheng Bao, and Ron Fedkiw. A virtual node algorithm for changing mesh topology during simulation. *ACM Trans. on Graphics*, 23(3):385–392, 2004.

[Müller and Chentanez, 2011] Matthias Müller and Natapong Chentanez. Solid simulation with oriented particles. *ACM Trans. on Graphics*, 2011.

[Müller and Gross, 2004] Matthias Müller and Markus Gross. Interactive virtual materials. In *Proc. of Graphics Interface*, pages 239–246, 2004.

[Müller et al., 2001] Matthias Müller, Leonard McMillan, Julie Dorsey, and Robert Jagnow. Real-time simulation of deformation and fracture of stiff materials. In *Proceedings of the Eurographic workshop on Computer animation and simulation*, pages 113–124, New York, NY, USA, 2001. Springer-Verlag New York, Inc.

[Müller et al., 2002] Matthias Müller, Julie Dorsey, Leonard McMillan, Robert Jagnow, and Barbara Cutler. Stable real-time deformations. In *Proc. of Symp. on Computer Animation*, pages 163–170, 2002.

[Müller et al., 2004a] Matthias Müller, Richard Keiser, Andrew Nealen, Mark Pauly, Markus Gross, and Marc Alexa. Point-based animation of elastic, plastic and melting objects. In *Proc. of Symp. on Computer Animation*, pages 141–151, 2004.

[Müller et al., 2004b] Matthias Müller, Matthias Teschner, and Markus Gross. Physically based simulation of objects represented by surface meshes. In *Proc. of Computer Graphics International*, pages 26–33, 2004.

[Müller et al., 2005] Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. Meshless deformations based on shape matching. *ACM Trans. on Graphics*, 24(3):471–478, 2005.

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

[Naghdi, 1972] P.M. Naghdi. *Handbuch der Physik, Mechanics of Solids II*, volume VI a/2. Springer, Berlin, 1972.

[Nealen et al., 2006] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically based deformable models in computer graphics. *Comput. Graphics Forum*, 25(4):809–836, 2006.

[Nesme et al., 2006] Matthieu Nesme, Yohan Payan, and François Faure. Animating shapes at arbitrary resolution with non-uniform stiffness. In *3rd Workshop in Virtual Reality Interaction and Physical Simulation, VRIPHYS '06, November, 2006*, Madrid, Espagne, November 2006.

[Nesme et al., 2009] Matthieu Nesme, Paul G. Kry, Lenka Jeřábková, and François Faure. Preserving topology and elasticity for embedded deformable models. *ACM Trans. on Graphics*, 28(3):52:1–52:9, 2009.

[Nocedal and Wright, 2006] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, August 2006.

[Norton et al., 1991] Alan Norton, Greg Turk, Bob Bacon, John Gerth, and Paula Sweeney. Animation of fracture by physical modeling. *Vis. Comput.*, 7:210–219, July 1991.

[O'Brien and Hodgins, 1999] James F. O'Brien and Jessica K. Hodgins. Graphical modeling and animation of brittle fracture. In *Proc. of ACM SIGGRAPH*, pages 137–146, 1999.

[O'Brien et al., 2002] James F. O'Brien, Adam W. Bargteil, and Jessica K. Hodgins. Graphical modeling and animation of ductile fracture. *ACM Trans. on Graphics*, 21(3):291–294, 2002.

[Otaduy et al., 2007] Miguel A. Otaduy, Daniel Germann, Stephane Redon, and Markus Gross. Adaptive deformations with fast tight bounds. In *Proc. of Symp. on Computer Animation*, pages 181–190, 2007.

[Pai, 2002] Dinesh K. Pai. STRANDS: interactive simulation of thin solids using Cosserat models. *Comput. Graphics Forum*, 21(3):347–352, 2002.

[Pauly et al., 2005] Mark Pauly, Richard Keiser, Bart Adams, Philip Dutre, Markus Gross, and Leonidas J. Guibas. Meshless animation of fracturing solids. *ACM Trans. on Graphics*, 24(3):957–964, 2005.

[Picinbono et al., 2000] Guillaume Picinbono, Herve Delingette, and Nicholas Ayache. Real-time large displacement elasticity for surgery simulation: Non-linear

tensor-mass model. In *Proceedings of the Third International Conference on Medical Image Computing and Computer-Assisted Intervention*, MICCAI '00, pages 643–652, London, UK, 2000. Springer-Verlag.

[Picinbono et al., 2003] Guillaume Picinbono, Hervé Delingette, and Nicholas Ayache. Non-linear anisotropic elasticity for real-time surgery simulation. *Graph. Models*, 65:305–321, September 2003.

[Popović et al., 2000] Jovan Popović, Steven M. Seitz, Michael Erdmann, Zoran Popović, and Andrew Witkin. Interactive manipulation of rigid body simulations. In *Proc. of ACM SIGGRAPH*, pages 209–217, 2000.

[Press et al., 2007] Willian H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes, The Art of Scientific Computing, 3rd Edition*. Cambridge University Press, 2007.

[Provot, 1995] Xavier Provot. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Graphics Interface '95*, pages 147–154, May 1995.

[Rabczuk and Belytschko, 2004] T. Rabczuk and T. Belytschko. Cracking particles: a simplified meshfree method for arbitrary evolving cracks. *International Journal for Numerical Methods in Engineering*, (61):2316–2343, 2004.

[Rivers and James, 2007] Alec R. Rivers and Doug L. James. FastLSM: fast lattice shape matching for robust real-time deformation. *ACM Trans. on Graphics*, 26(3):82, 2007.

[Rubin and Bodner, 2002] M. B. Rubin and S. R. Bodner. A three-dimensional nonlinear model for dissipative response of soft tissue. *International Journal of Solids and Structures*, 39(19):5081 – 5099, 2002.

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

[Schenk and Gärtner, 2002] Olaf Schenk and Klaus Gärtner. Solving unsymmetric sparse systems of linear equations with pardiso. In *Proceedings of the International Conference on Computational Science-Part II*, ICCS '02, pages 355–363, London, UK, UK, 2002. Springer-Verlag.

[Selle et al., 2008] Andrew Selle, Michael Lentine, and Ronald Fedkiw. A mass spring model for hair simulation. *ACM Trans. on Graphics*, 27(3):64.1–64.11, 2008.

[Sheffer and Kraevoy, 2004] Alla Sheffer and Vladislav Kraevoy. Pyramid Coordinates for Morphing and Deformation. In *Proc. 3D Data Processing, Visualization, and Transmission*, pages 68–75, 2004.

[Sifakis et al., 2007a] Eftychios Sifakis, Kevin G. Der, and Ron Fedkiw. Arbitrary cutting of deformable tetrahedralized objects. In *Proc. of Symp. on Computer Animation*, pages 73–80, 2007.

[Sifakis et al., 2007b] Eftychios Sifakis, Tamar Shinar, Geoffrey Irving, and Ronald Fedkiw. Hybrid simulation of deformable solids. In *Proc. of Symp. on Computer Animation*, pages 81–90, 2007.

[Smith et al., 2001] Jeffrey Smith, Andrew P. Witkin, and David Baraff. Fast and controllable simulation of the shattering of brittle objects. *Comput. Graph. Forum*, 20(2):81–90, 2001.

[Solenthaler et al., 2007] B. Solenthaler, J. Schlfli, and R. Pajarola. Pajarola r.: A unified particle model for fluid solid interactions: Research articles. *Comput. Animat. Virtual Worlds*, 18:69–82, 2007.

[Sorkine et al., 2004] O. Sorkine, Y. Lipman, D. Cohen-Or, M. Alexa, C. Rossl, and Hans-Peter Seidel. Laplacian surface editing. In *Proc. of Symp. on Geometry Processing*, pages 179–188, 2004.

[Spillmann and Teschner, 2007] J. Spillmann and M. Teschner. CoRdE: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *Proc. of Symp. on Computer Animation*, pages 63–72, 2007.

[Stam, 1999] Jos Stam. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, Proc. of ACM SIGGRAPH, pages 121–128, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[Stam, 2009] Jos Stam. Nucleus: Towards a unified dynamics solver for computer graphics. In *IEEE International Conference on Computer-Aided Design and Computer Graphics*, pages 1–11, Aug. 2009.

[Steinemann et al., 2006a] Denis Steinemann, Matthias Harders, Markus Gross, and Gabor Szekely. Hybrid cutting of deformable solids. In *Proc. of IEEE VR*, pages 35–42, 2006.

[Steinemann et al., 2006b] Denis Steinemann, Miguel A. Otaduy, and Markus Gross. Fast arbitrary splitting of deforming objects. In *Proc. of Symp. on Computer Animation*, pages 63–72, 2006.

[Steinemann et al., 2008] Denis Steinemann, Miguel A. Otaduy, and Markus Gross. Fast adaptive shape matching deformations. In *Proc. of Symp. on Computer Animation*, pages 87–94, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association.

[Sueda et al., 2011] Shinjiro Sueda, Garrett L. Jones, David I. W. Levin, and Dinesh K. Pai. Large-scale dynamic simulation of highly constrained strands. *ACM Trans. on Graphics*, 2011.

[Sumner and Popović, 2004] Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. In *ACM Trans. on Graphics*, pages 399–405. ACM, 2004.

[Sumner et al., 2005] Robert W. Sumner, Matthias Zwicker, Craig Gotsman, and Jovan Popović. Mesh-based inverse kinematics. In *ACM Trans. on Graphics*, pages 488–495, New York, NY, USA, 2005. ACM.

[Terzopoulos and Fleischer, 1988] D. Terzopoulos and K. Fleischer. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. In *Proc. of ACM SIGGRAPH*, pages 269–278, 1988.

[Terzopoulos et al., 1987] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *Proc. of ACM SIGGRAPH*, pages 205–214, 1987.

[Terzopoulos et al., 1989] D. Terzopoulos, J. Platt, and K. Fleischer. Heating and melting deformable models (from goop to glop). In *Graphics Interface*, 1989.

[Teschner et al., 2004] Matthias Teschner, Bruno Heidelberger, Matthias Müller, and Markus Gross. A versatile and robust model for geometrically complex deformable solids. In *Proc. of Computer Graphics International'04*, pages 312–319, 2004.

[Thomaszewski et al., 2006] Bernhard Thomaszewski, Markus Wacker, and Wolfgang Straßer. A consistent bending model for cloth simulation with corotational subdivision finite elements. In *Proc. of Symp. on Computer Animation*, pages 107–116, 2006.

[Thomaszewski et al., 2009] Bernhard Thomaszewski, Simon Pabst, and Wolfgang Straßer. Continuum-based Strain Limiting. *Comput. Graphics Forum*, 4 2009.

[Thoutireddy and Ortiz, 2004] P. Thoutireddy and M. Ortiz. A variational r-adaption and shape-optimization method for finite-deformation elasticity. *Int. J. Numer. Meth. Engng.*, 61:1–21, 2004.

[Thürey et al., 2006] N. Thürey, R. Keiser, M. Pauly, and U. Rüde. Detail-preserving fluid control. In *Proc. of Symp. on Computer Animation*, pages 7–12, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[Thürey et al., 2010] Nils Thürey, Chris Wojtan, Markus Gross, and Greg Turk. A multiscale approach to mesh-based surface tension flows. *ACM Trans. on Graphics*, 29(3), 2010.

[Toledo et al., 2003] S. Toledo, D. Chen, and V. Rotkin. Taucs: A library of sparse linear solvers. http://www.tau.ac.il/∼stoledo/taucs, 2003.

[Treuille et al., 2003] Adrien Treuille, Antoine McNamara, Zoran Popović, and Jos Stam. Keyframe control of smoke simulations. *ACM Trans. on Graphics*, 22:716–723, July 2003.

[Treuille et al., 2006] Adrien Treuille, Andrew Lewis, and Zoran Popović. Model reduction for real-time fluids. *ACM Trans. on Graphics*, 25:826–834, July 2006.

[Twigg and James, 2007] Christopher D. Twigg and Doug L. James. Many-worlds browsing for control of multibody dynamics. *ACM Trans. Graph.*, 26, July 2007.

[Twigg and Kačić-Alesić, 2010] Christopher D. Twigg and Zoran Kačić-Alesić. Point cloud glue: constraining simulations using the procrustes transform. In *Proc. of Symp. on Computer Animation*, pages 45–54, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

[Van Gelder, 1998] Allen Van Gelder. Approximate simulation of elastic membranes by triangulated spring meshes. *Journal of Graphics Tools*, 3(2):21–42, 1998.

[Veubeke, 1976] B. Fraeijs De Veubeke. The dynamics of flexible bodies. *International Journal of Engineering Science*, 14(10):895 – 913, 1976.

[Volino et al., 2009] Pascal Volino, Nadia Magnenat-Thalmann, and Francois Faure. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Trans. on Graphics*, 28(4):1–16, 2009.

[Wachspress, 1975] Eugene L. Wachspress. *A Rational Finite Element Basis*. Academic Press, 1975.

[Wang and Devarajan, 2005] Xiuzhong Wang and Venkat Devarajan. 1D and 2D structured mass-spring models with preload. *Visual Computer*, 21(7):429–448, 2005.

[Wang et al., 2011] Huamin Wang, Ravi Ramamoorthi, and James F. O'Brien. Data-driven elastic models for cloth: Modeling and measurement. *ACM Trans. on Graphics*, 30(4):71:1–11, July 2011. Vancouver, BC Canada.

[Wardetzky et al., 2007] Max Wardetzky, Mikls Bergou, David Harmon, Denis Zorin, and Eitan Grinspun. Discrete Quadratic Curvature Energies [CAGD Most Cited Paper Award for 2010]. *Computer Aided Geometric Design*, 24(8-9):499–518, Nov 2007.

[Weber et al., 2009] Ofir Weber, Mirela Ben-Chen, and Craig Gotsman. Complex barycentric coordinates with applications to planar shape deformation. *Computer Graphics Forum (Proc. of Eurographics)*, 28(2), 2009.

[Wei et al., 2009] Li-Yi Wei, Sylvain Lefebvre, Vivek Kwatra, and Greg Turk. State of the art in example-based texture synthesis. In *Proc. of Eurographics '09, State of the Art Report*, 2009.

[Wicke et al., 2005] Martin Wicke, Denis Steinemann, and Markus Gross. Efficient animation of point-sampled thin shells. *Comput. Graphics Forum*, 24:667–676, 2005.

[Wicke et al., 2007] Martin Wicke, Mario Botsch, and Markus Gross. A finite element method on convex polyhedra. *Comput. Graphics Forum (Proc. Eurographics)*, 26(3):355–364, 2007.

[Wicke et al., 2009] Martin Wicke, Matt Stanton, and Adrien Treuille. Modular bases for fluid dynamics. *ACM Trans. on Graphics*, 28(3), 2009.

[Winkler et al., 2010] T. Winkler, J. Drieseberg, M. Alexa, and K. Hormann. Multi-scale geometry interpolation. *Computer Graphics Forum*, 29(2):309–318, May 2010. Proceedings of Eurographics.

[Wirth et al., 2010] Benedikt Wirth, Leah Bar, Martin Rumpf, and Guillermo Sapiro. A continuum mechanical approach to geodesics in shape space. *Int. J. of Computer Vision*, pages 1–26, 2010.

[Witkin and Kass, 1988] Andrew Witkin and Michael Kass. Spacetime constraints. *SIGGRAPH Comput. Graph.*, 22:159–168, June 1988.

[Wojtan and Turk, 2008] Chris Wojtan and Greg Turk. Fast viscoelastic behavior with thin features. *ACM Trans. on Graphics*, 27(3):47:1–47:8, 2008.

[Wojtan et al., 2006] Chris Wojtan, Peter J. Mucha, and Greg Turk. Keyframe control of complex particle systems using the adjoint method. In *Proc. of Symp. on Computer Animation*, pages 15–23, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[Wojtan et al., 2009] Chris Wojtan, Nils Thürey, Markus Gross, and Greg Turk. Deforming meshes that split and merge. *ACM Trans. on Graphics*, 28(3):76:1–76:10, 2009.

[Wojtan et al., 2010] Chris Wojtan, Nils Thürey, Markus Gross, and Greg Turk. Physics-inspired topology changes for thin fluid features. *ACM Trans. Graph.*, 29(4):1–8, 2010.

[Wu et al., 2001] Xunlei Wu, Michael S. Downes, Tolga Goktekin, and Frank Tendick. Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes. *Comput. Graphics Forum (Proc. Eurographics)*, 20(3):349–358, 2001.

[Yang et al., 2000] H.T.Y. Yang, S. Saigal, A. Masud, and R.K. Kapania. A survey of recent shell finite elements. *Int. J. Numer. Methods Eng.*, (47):101–127, 2000.

[Zerbato et al., 2007] D. Zerbato, S. Galvan, and P. Fiorini. Calibration of mass spring models for organ simulations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 370–375, 29 2007-Nov. 2 2007.

Curriculum Vitae

Sebastian Martin

Personal Data

Oct. 30, 1980	Born in Sursee, Switzerland
Nationality	Swiss

Education

Oct. 7, 2011	Ph.D. defense.
Jun. 2007 – Sep. 2011	Research assistant and Ph. D. student at the Computer Graphics Laboratory of the Swiss Federal Institute of Technology (ETH) Zurich, Prof. Markus Gross.
Feb. 2007	Diploma degree in Computer Science.
Oct. 2001 – Feb. 2007	Diploma Studies of Computer Science, ETH Zurich, Switzerland. Specialization: Computational Sciences; Complementary studies: Computer Graphics.

Awards

February 2007	ETH Medal, Master Thesis "Spherical Parameterization"
July 2008	Best Student Paper Award "Polyhedral Finite Elements Using Harmonic Basis Functions" at Symposium on Geometry Processing 2008.

Scientific Publications

S. MARTIN, B. THOMASZEWSKI, E. GRINSPUN, and M. GROSS. Example-based Elastic Materials. In *Proceedings of ACM SIGGRAPH (Vancouver, Canada, August 7-11, 2011), ACM Transaction on Graphics, vol. 30, no. 4, pp. 72:1-72:8.*

S. MARTIN, P. KAUFMANN, M. BOTSCH, E. GRINSPUN, and M. GROSS. Unified Simulation of Elastic Rods, Shells, and Solids. In *Proceedings of ACM SIGGRAPH (Los Angeles, USA, July 25-29, 2010), ACM Transaction on Graphics, vol. 29, no. 3, pp. 39:1-39:10.*

S. MARTIN, C. HUBER, P. KAUFMANN, and M. GROSS. Shape-Preserving Animation of Deformable Objects. In *Proceedings of Vision, Modeling, and Visualization (VMV) (Braunschweig, Germany, November 16-18, 2009).*

P. KAUFMANN, S. MARTIN, M. BOTSCH, and M. GROSS. Implementation of Discontinuous Galerkin Kirchhoff-Love Shells. In *Technical Report No. 622, Institute of Visual Computing, ETH Zurich, 2009.*

P. KAUFMANN, S. MARTIN, M. BOTSCH, E. GRINSPUN, and M. GROSS. Enrichment Textures for Detailed Cutting of Shells In *Proceedings of ACM SIGGRAPH (New Orleans, USA, August 3-7, 2009), ACM Transactions on Graphics, vol. 28, no.3, pp. 50:1-50:10.*

P. KAUFMANN, S. MARTIN, M. BOTSCH, and M. GROSS. Flexible Simulation of Deformable Models Using Discontinuous Galerkin FEM In *Journal of Graphical Models, 2009.*

S. MARTIN,P. KAUFMANN, M. BOTSCH, and M. GROSS. Polyhedral Finite Elements Using Harmonic Basis Functions In *Proceedings of Eurographics Symposium on Geometry Processing 2008 (Copenhagen, Denmark, July 2-4, 2008), Computer Graphics Forum, (Best Student Paper Award)*

P. KAUFMANN, S. MARTIN, M. BOTSCH, and M. GROSS. Flexible Simulation of Deformable Models Using Discontinuous Galerkin FEM In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (Dublin, Ireland, July 7-9, 2008).*

Employment

Jun. 2007 – Sep. 2011	Research assistant at ETH Zurich, Zurich, Switzerland.
Aug. 2004 – Jan. 2005	Internship at ABB AG, Baden, Switzerland.
May 2001 – Aug. 2001	Internship at Datacolor AG, Dietlikon, Switzerland.