Diss. ETH No. 22225

Computational Design and Fabrication of Deformable Objects

A thesis submitted to attain the degree of
DOCTOR OF SCIENCES of ETH ZURICH
(Dr. sc. ETH Zurich)

presented by
MELINA SKOURAS
Diplôme d'Ingénieur, ENSIMAG, France
born on 27.11.1981
citizen of France

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

2014

For M.C., who always believed that one day I would get a PhD

Abstract

Deformable objects have a plethora of applications: they can be used for entertainment, advertisement, engineering or even medical purposes to cite but a few examples. However it is precisely because they deform that such objects are difficult to design. Their creator must *foresee* and *invert* effects of external forces on the behavior of the figure in order to take the proper design decisions. In this thesis, we investigate approaches based on physics-based simulation and inverse optimization techniques which alleviate these difficulties and we propose a complete framework to design custom deformable objects by automating some of the most tedious aspects of the design process. This framework is tailored to various applications in which optimization of diverse variables comprising rest shape, materials and actuation system is alternately considered. Validation of our method is performed by fabricating representative sets of physical prototypes designed with our method and compared to the results predicted by simulation.

We start by introducing the basic concepts of physics-based simulation and optimization on which our inverse modeling component, central to our approach, is built. We show how the optimization of the system's degrees of freedom can be cast as a constrained minimization problem combining strict enforcement of physical consistency with target shape approximation.

A method to automatically design and easily fabricate rubber balloons of desired inflated shapes is presented in the following part of the thesis. In that section we precisely describe the different stages of the pipeline including material modeling and fitting, rest shape optimization and physical fabrication.

We then extend our framework to address the problem of creating physical replicas of two- and three-dimensional animated digital characters which can be actuated using pins, strings or posed by hand. Our approach automatically computes a sparse set of actuators as well as their locations on the surface and optimizes the internal material distribution such that the resulting character exhibits the desired deformation behavior.

Fully automatic systems are not always desirable. Aesthetics considerations, for

example, are subjective aspects and are consequently difficult, not to say impossible, to handle fully automatically. In the last part of this thesis we show how optimization can be incorporated in an interactive design tool to help the designer to create inflatable structures without hindering his creativity. The artistic task of sketching the boundaries of the structure segments on the desired target inflated shape is let to the user while a fast underlying optimizer automatically computes the flat panels to assemble, thus allowing interactive exploration of the seam layout space.

Résumé

Les objets déformables ont un nombre innombrable d'applications, que ce soit dans l'industrie du divertissement, de la publicité, en ingénierie ou encore en médecine. Mais leur nature déformable est justement ce qui les rend si difficiles à concevoir. Leur créateur doit *prévoir* et *inverser* les effets des forces externes qui régissent leur comportement afin de prendre les bonnes décisions lors de leur conception. Cette thèse propose l'étude d'approches basées sur la simulation physique et la modélisation inverse qui permettent d'alléger ces difficultés et présente un cadre complet pour concevoir des objets déformables personnalisés grâce à l'automatisation de certains des aspects les plus fastidieux du processus de création. Nous adaptons ce cadre à de multiples applications, pour lesquelles l'optimisation de différentes variables telles que la forme initiale, les matériaux mais aussi le système actionneur sont considérées tour à tour. Enfin, nous validons notre méthode en fabriquant un nombre représentatif de prototypes physiques conçus à l'aide de notre système et en les comparant aux résultats prédits par simulation.

Nous commencerons par introduire les concepts fondamentaux de simulation physique et d'optimisation sur lesquelles reposent notre composant de modélisation inverse, au cœur de notre approche. Nous montrerons également comment l'optimisation des degrés de liberté du système peut se ramener à un problème de minimisation sous contraintes combinant stricte garantie de la cohérence physique de l'objet et bonne approximation de la forme souhaitée.

Nous présenterons ensuite une méthode pour concevoir et fabriquer aisément des ballons de baudruche aux formes personnalisées. Dans cette section, nous décrirons précisément les différentes étapes du processus comprenant la modélisation des matériaux employés, l'optimisation de la forme initiale des ballons ainsi que leur fabrication.

Nous étendrons ensuite le cadre développé au problème de la réplication physique de personnages virtuels animés bi- et tridimensionnels, manipulés à l'aide de fils, d'attaches rigides ou encore positionnés manuellement.

Un système entièrement automatisé n'est pas toujours souhaitable. Les considérations esthétiques, par exemple, sont des aspects subjectifs qu'il est par conséquent difficile, si ce n'est impossible, de traiter de manière entièrement automatique. Dans la dernière partie de cette thèse, nous montrerons comment combiner optimisation et conception interactive au sein d'un outil permettant à l'utilisateur de créer des structures gonflables tout en lui garantissant entière créativité. Pour ce faire, nous laissons à l'utilisateur la tâche artistique d'esquisser les contours des pièces du ballon directement sur la forme désirée, tandis que leur patron est automatiquement calculé, lui permettant ainsi une exploration interactive de l'ensemble des assemblages possibles.

Acknowledgments

Changes are exciting. Sometimes scaring. Always enriching. When I left Dassault Systèmes to come to Zurich, I left a great team and a job that I liked. I knew why I wanted to do a PhD, but one can only be certain of what one leaves behind, not of what one will find. After more than four years in the Computer Graphics Lab, I do not regret my choice by any means and I am deeply grateful to Prof. Dr. Markus Gross, my thesis supervisor, for having accepted to take me in his lab and having provided such fantastic working conditions at the Computer Graphics Lab as well as at Disney Research.

Collaborating with Disney was a great experience and, as a member of his team, I warmly thank Bernd Bickel for his support, enthusiasm and great leadership.

I have an infinite gratitude to Bernhard Thomaszewski for all what I learned from him, his constant guidance, all the hours of passionate discussions that we spent, and for being a great challenger in intellectual jousts that I did not always won.

I would also like to thank my other close collaborators Prof. Eitan Grinspun and his sparkling ideas, Stelian Coros, Peter Kaufmann and Akash Garg, as well as all the persons who contributed in one way or another to the success of my projects, in particular Christian Schumacher, Maurizio Nitti and Ronnie Gänsli.

I warmly thank my various office mates and all the members of the CGL, DRZ and IGL teams. They highly contributed to make the lab a fun and friendly place to work in.

I want to thank my family for their unconditional love and support, and for always being there for me.

Last but not least, I deeply thank my boyfriend Nicolas for his love, understanding, patience, especially when deadlines approach, and in sharing my life and making it so happy.

This research has been partially supported by the NCCR Co-Me of the Swiss National Science Foundation.

Contents

CHAPTER 1

Introduction

Creating and animating digital characters are ones of the core topics of computer graphics. Decades of research in related fields led to the development of myriads of tools to process, edit, animate and render computer-generated or digitally acquired models. However, while extremely convincing figures can be seen in animation movies and video games, many applications also require their physical embodiment in the real world. Shows and other attractions involving animatronics imitating famous characters are very popular in theme parks, cartoons figurines are always successful toys in the eyes of children, recognizable balloons are invariably cheered by the public during parades...

But translating from digital to real is far from straightforward. The behaviour of real objects in the real world is governed by complex physical laws. The models are not simple animated shapes anymore but have a mass, are made of diverse complex materials, deform under the action of real forces caused by real physical phenomena. Countless parameters, potentially acting as just as many design variables, may affect the final shape of the envisioned object, so well that its creator needs to predict and revert their effects in order to take the proper design decisions. Bringing to life specific characters with particular behaviours clearly becomes a challenging task, and while existing softwares are often sufficient for traditional applications of computer graphics, many artifices employed when generating realistic virtual scenes in video games or animation movies can simply not be used in this context, strengthening the need for dedicated tools.

This demand is not reserved for an expert few. In a society where one can already personalize one's car, one's furniture, one's clothing, people do not always want to

buy items imagined by others but create their own custom objects, as demonstrated by the boom of arts and crafts stores and 3d printing services. Simple and intuitive tools to easily bridge the gap between what users have in mind and what they will hold in their hands are consequently more and more desirable.

In computer graphics, as in engineering and mechanics, physics-based simulation techniques proved to be very efficient means to accurately compute the behaviour of an object over time. Conversely, if the object states are known, inverse modeling approaches can be employed to infer the system's unknown variables. This is the avenue that we pursue in this thesis to develop novel design tools allowing users to conveniently create complex objects by focusing on the target shapes — possibly making use of of the abundant models already available — rather than on the tedious task of parameter tuning.

The use of inverse modeling methods for fabrication purposes is not new per se. Such approaches have been successfully utilized to optimize static rigid structures, with the aim of improving various criteria such as stability [Smith et al., 2002; Whiting et al., 2012; Stava et al., 2012], appearance [Weyrich et al., 2009; Hasan et al., 2010], arrangement of multiple pieces [Yu et al., 2011; Xin et al., 2011]. Some recent works looked into articulation [Bächer et al., 2012; Calì et al., 2012] or even actuation of mechanical structures [Zhu et al., 2012; Coros et al., 2013], but objects exhibiting *large deformations*, although ubiquitous in our environment in a variety of forms, received much less attention and their case has been little explored. In this thesis, we focus on the latter and propose novel methods to design deformable objects of different natures, shells but also solids, quasi-inextensible to largely stretchable ones, made of one or multiple materials, static and animated ones. We explore the challenges arising from this diversity via the study of concrete types of deformable objects for which we developed tailored design systems allowing us to fabricate real, physical prototypes.

We start by looking into the problem of large in-plane deformations, that we address in the context of the design of rubber balloons. These balloons are thin shells typically made of latex that largely expand under inflation. Cheap and easy to fabricate, they are very popular as decorative items or for entertainment purposes. However most of the balloons available in the stores have very simple shapes. This lack of originality can be explained by the difficulty to infer the initial rest shape of more complex figures. Rubber materials have indeed a particular behaviour when they are subject to pressure forces that is very unintuitive and incorrectly captured by most material models commonly used in physics-based simulation. We present a process to accurately model and fit balloon materials and automatically compute the uninflated shape of balloons of extremely diverse target forms.

We then focus on the challenges posed by animated heterogeneous material struc-

tures, widely used in the entertainment industry. From animatronics in theme parks to monsters in feature films, such objects are typically made of rigid structures covered by a soft skin, thus combining different materials and are actuated using pins, strings, rods or any other mean of applying forces at specific locations on the surface. The complexity of actuated deformable characters and the difficulty of designing them become obvious when one thinks of the immense size of the design space — materials to combine, shapes of hidden components, actuation system to use may all be degrees of freedom of the original design problem. The animated nature of such figures leads to extra challenges as the design of the actuation system itself is now part of the problem. We introduce a three-stage process to create physical replicas of animated digital characters which, given a deformable object and a set of target poses, automatically computes optimal number and placement of a sparse set of actuators, as well as the internal material distribution of the object in order to best approximate the desired deformation behavior.

Finally, we consider the question of aesthetic control through the study of panel-based structures. Indeed, planar pieces of fabric, paper, metal foils, bent and assembled to form complex structures that can in some cases be inflated or stuffed are omnipresent in our environment. While segmenting and flattening arbitrary models is already nontrivial from a geometric point of view, approximating freeform shapes by deformable quasi-inextensible shells is even more challenging as one must solve the paneling problem in the space of physically feasible structures. Furthermore, the concept of optimality itself is here a fuzzy notion since panels boundaries often appear as visible seams in the assembly, so much so that their ideal locations may depend on subjective criteria. This reveals a fundamental limitation of fully automatic approaches, namely the lack of control on the obtained solution. Even when functional requirements or other measurable criteria can be automatically fulfilled, the treatment of aesthetics or artistic aspects, if only to specify them, often requires user intervention. However, a computational design system does not and should not necessarily aim at replacing the user. It can also simply assist him in his design task by taking over tedious or difficult aspects while still leaving him command of what falls under subjectivity. Following that direction, we propose a method to design inflatable structures which allows the user to interactively explore various seams layouts by letting him directly draw the boundaries of the panels on the inflated target shape while offloading the computation of the corresponding flat patterns to the computer. Our approach is based on a fast physics-based model for inflatable structures using tension field theory and a dedicated constraint-optimization component for computing the 2D patterns.

1.1 Contributions

This thesis makes the following main contributions:

- A complete framework for designing and fabricating free-form rubber balloons, including parameter acquisition, accurate computational modeling, rest shape optimization and physical fabrication. The result of our design system is used to 3D-print custom positive molds for producing silicone balloons of a large diversity of forms.
- A comprehensive process to create actuated deformable characters. Given as input a set of target poses and a pair of base materials with different properties, our method is able to optimize the actuation system — number and locations of actuators — as well as the material distribution of the object in order to best match the input poses. The output of the pipeline consists in silicone-molded and 3D-printed physical prototypes that are animated using pins, strings or simply posed by hand.
- A tailored design tool to create inflatable structures with possible internal panels. By combining user-guided seam specifications on the target model with fast and automatic pattern generation, our system allows interactive exploration of various seam layouts. Our approach relies on an accurate coarse-scale simulator for inflatable membranes using tension field theory and on a specialized constrainted optimization method allowing adjustable balance between expected shape approximation and seam quality.

1.2 Thesis Outline

The organization of this thesis is as follows:

- **Chapter 2** reviews related work in the fields of simulation, material modeling and fitting, control, shape and structural design optimization, and fabrication. Existing work connected to more specific concepts will be presented in their respective chapters.
- **Chapter 3** introduces the basics of physics-based simulation and optimization necessary to the understanding of the subsequent chapters. It also exposes the generic scheme that we use for optimizing the various system variables that we alternatively consider in the different parts of the thesis.
- **Chapter 4** focuses on the design of custom rubber balloons. The different stages of the full pipeline, i.e. the acquisition of rubber properties, the model-

ing of latex and silicone, the optimization of the rest shape of the balloons and their fabrication are successively described.

- **Chapter 5** extends the proposed framework to the creation of actuated deformable characters. Additional variables such as actuator locations and dual material distributions, as well as multiple target poses, are now considered.
- **Chapter 6** presents an interactive design system that allows non-expert users to quickly create inflatable structures. The optimization procedure previously introduced is tailored to the problem of computing 2D flat patterns and is combined to a sketching interface for drawing desired panel seams on a target model.
- **Chapter 7** concludes this thesis with a discussion on its main contributions and provides directions for potential future work.

1.3 Publications

This thesis is based on the following peer-reviewed publications:

M. SKOURAS, B. THOMASZEWSKI, B. BICKEL, and M. GROSS. Computational Design of Rubber Balloons, *Proceedings of Eurographics (Cagliari, Italy, May 13-18, 2012), Computer Graphics Forum*, vol. 31, no. 2, pp. 835–844, 2012.

M. SKOURAS, B. THOMASZEWSKI, S. COROS, B. BICKEL and M. GROSS. Computational Design of Actuated Deformable Characters, *Proceedings of ACM SIGGRAPH (Anaheim, USA, July 21-25, 2013), ACM Transactions on Graphics*, vol. 32, no. 4, pp. 82:1–82:10, 2013.

M. SKOURAS, B. THOMASZEWSKI, B. BICKEL, P. KAUFMANN, A. GARG, E. GRINSPUN and M. GROSS. Designing Inflatable Structures, *Proceedings of ACM SIGGRAPH (Vancouver, Canada, August 10-14, 2014), ACM Transactions on Graphics*, vol. 34, no. 4, 2014.

During the course of this thesis, the following peer-reviewed paper was also published:

B. BICKEL, P. KAUFMANN, M. SKOURAS, B. THOMASZEWSKI, D. BRADLEY, T. BEELER, P. JACKSON, S. MARSCHNER, W. MATUSIK and M. GROSS. Physical Face Cloning, *Proceedings of ACM SIGGRAPH (Los Angeles, USA, August 5-9, 2012), ACM Transactions on Graphics*, vol. 31, no. 4, pp. 118:1–118:10, 2012.

CHAPTER 2

Related Work

Computational design is a relatively new field in computer graphics, which is rapidly gaining interest in our community. Nevertheless, being at a crossroads between different disciplines — computer graphics, mechanics, engineering, mathematics — the related literature is particularly abundant. A typical fabrication-oriented design system comprises various elements ranging from an acquisition system for capturing material properties to modeling components, including simulation and optimization units. Each of these pieces has its own history. This chapter reviews existing work related to these topics and most connected to our research. It also provides a general background for the methods and ideas developed throughout this thesis.

2.1 Simulation

In 1987, Terzopoulos et al. introduced the concept of physics-based simulation to the computer graphics community [Terzopoulos et al., 1987]. The model that they presented relied on Newtonian mechanics and elasticity theory [Landau et al., 1986] to realistically animate deformable objects. After more than two decades of active research in this field, myriads of methods for physically plausible simulation abound. Approaches based on mass-springs systems, finite element methods [Hughes, 2000] or particle models [Müller et al., 2004] have been proposed to simulate a wide range of behaviours, ranging from elastic deformation to collision and fracture [O'Brien and Hodgins, 1999], and were successfully applied to the simulation of multiple kinds of media, e.g. solids

but also fluids [Bridson, 2008] and melting objects [Terzopoulos et al., 1989; Carlson et al., 2002]. Reviewing all these models is out of the scope of this thesis and we refer to Gibson and Mirtich [1997], and Nealen et al. [2006] for excellent surveys on this topic.

The fantastic popularity of physics-based simulation techniques stem from their ability to produce extremely convincing results at a relatively limited cost, if only in terms of required user-interaction. However, different applications will impose specific requirements. Animation movies and feature films will typically favor high realism and easy control over the simulations, while video games will focus on speed or low memory footprint. In the context of fabrication-oriented design, accuracy is generally the major concern and continuum-based formulations [Bonet and Wood, 1997] associated with finite element methods [Hughes, 2000] are largely privileged.

This thesis focuses primarily on inflatable membranes (Chapter 4 and Chapter 6) and solids (Chapter 5), that we simulate using a standard finite element approach recapped in Chapter 3. While simulating the solids does not pose any particular problem, correctly modeling the membranes appears to be particularly challenging. The inflatable structures that we consider are of two kinds: balloons made of rubber, usually latex, which stretch significantly during inflation and have curved rest shapes, and inflatables consisting of assemblies of flat panels that typically resist stretching but are pliant to bending. While both types of balloons are widely used for entertainment and advertisement purposes, their different natures yield different kinds of experiences. Rubber balloons are fascinating because they largely expand. Inflatable panel-based structures are exciting because of their potentially larger dimensions – think of parade balloons –, wider variety of shapes – creases can be obtained –, richer visual aspect – panels of different colors or decorated with ink can be used–, and more diverse functions – they can also be used as furniture. Unsurprisingly, simulating such fundamentally different materials requires specific techniques and raises different issues. Accurate modeling of the nonlinear behaviour of silicone and latex materials is of primary importance when simulating rubber balloons; efficient treatment of compressive stresses is crucial when dealing with piecewise developable structures.

Membrane models have been largely used in computer graphics in the past, in particular in the context of cloth simulation. Wu et al. [Wu et al., 2001] simulate nonlinear membranes using finite elements. Volino et al. [Volino et al., 2009] use a membrane model derived from continuum mechanics for simulating cloth material. Grinspun et al. [Grinspun et al., 2003] describe a discrete model for simulating thin shells. The mechanics of inflatable membranes have been studied extensively in engineering [Bonet et al., 2000]. In computer vision, simplified balloon models

based on active contour approaches [Kass et al., 1988] have been used for shape recovery and tracking [Cohen, 1991; McInerney and Terzopoulos, 1993].

The wrinkling behaviour of inflatable structures made of developable patterns resemble the one of cloth, that also buckles at the onset of compression. This behavior leads to the characteristic folding patterns that define the typical appearance of real textiles, but it is inherently difficult to treat numerically: compression gives rise to negative eigenvalues in the force Jacobian and thus slows or even breaks most linear solvers. Choi and Ko [2002] proposed a mass-spring model that turns off force and Jacobian contribution from compressed springs and replaces them with custom-tailored *buckling springs*. Another approach to combat indefiniteness was presented by Teran et al. [2005], who clamp negative eigenvalues of elemental stiffness matrices. Instead of trying to avoid indefiniteness, the method of Rohmer et al. [2010] exploits the compression field extracted from elemental deformation tensors in order to add detailed wrinkles to a coarse simulation. As for Wang and Tang [2007; 2010], they analyse directional tensile to optimize patterns for compression garment. Inspired by tension field theory [Pipkin, 1986; Steigmann, 1990], we propose in Chapter 6 a fast physics-based model that addresses the difficulty of wrinkling analysis based on a relaxed energy which fades to zero before compressive stresses can occur. Such an approach was previously employed by Baginski et al.[2008] to simulate high altitude scientific balloons and to analyse formation of wrinkles on simple patterns.

2.2 Computational Materials

Fabrication-oriented design requires an accurate modeling of the materials to be used for fabrication. The behaviour of these materials is generally described in terms of stress-strain curves which relate applied loads to material deformations and define their so-called constitutive model [Bonet and Wood, 1997]. Many constitutive equations have been proposed to model deformable materials, starting with the simple Hooke's law for linear elastic materials to more sophisticated Ogden [Ogden, 1972] and polynomial models, capable of capturing nonlinearities in the material behaviour, in particular when dealing with large deformations. In the context of this work, a Neo-Hookean model [Bonet and Wood, 1997] appeared to be sufficient to adequately simulate solids (Chapter 5) and quasi-inextensible shells (Chapter 6), whereas the exponential model of Hart-Smith [1966] proved to be a good option in the case of inflatable rubbery structures (Chapter 4).

While choosing the proper material model is fundamental, selecting the right model parameters is at least as crucial for precise simulation. To circumvent the tedious task of tuning parameters by hand, a method of choice is to turn toward

fitting methods and to automatically estimate the model's unknowns from a set of example force-deformation measurements. Such data-driven techniques have been successfully applied to model and fit linear isotropic elastic materials [Becker and Teschner, 2007], nonlinear viscoelastic materials [Kauer et al., 2001] , nonlinear isotropic heterogeneous soft tissues [Bickel et al., 2009], nonlinear anisotropic cloth materials [Wang et al., 2011; Miguel et al., 2012] and plastic materials [Kajberg and Lindkvist, 2004]. A system to estimate material parameters in conjunction with contact texture and sound response was also proposed by Pai et al. [2001]. Martin et al. [2011] deduce a potential energy from a set of example shapes that allows to incorporate complex example-based material behavior for dynamic physical simulation. However, these materials might not be available to our fabrication process or, even worse, might not have a real-world counterpart. In this thesis, we follow the example-based modeling paradigm but, like Bickel et al. [2010], we stick to provided base materials – that we still fit to real data– and we play on different variables, i.e. the rest shape of the object and the arrangement of the base materials, to obtain a desired deformation behaviour.

2.3 Control

Deformation is the response of an object to the action of external forces. In other words, without external forces, a solid will not move from its rest configuration. This naturally raises the question of where forces need to be applied, and with which intensity, to trigger and steer desired deformations. The answer clearly depends on the target application and distinguishes between means, whatever they are, to create plausible animations, and physically-feasible control.

The first category is dominated by methods based on control forces which guide the simulation towards user-defined target poses or high level goals. The pioneering work on space-time constraints presented by Witkin and Kass [1988] more than two decades ago was followed by a number of techniques to art-direct simulation of fluids [McNamara et al., 2004; Thürey et al., 2006], shells [Bergou et al., 2007] and solids [Popović et al., 2000; Kondo et al., 2005; Barbič et al., 2009]. To counteract potential issues due to non-conservative compensation forces, Martin et al. [2011] build an example manifold that attracts the simulation, and Coros et al. [2012] use forces stemming from the changes in the rest shape of the object. However, while such approaches proved to be powerful to generate *realistic* animation, *fictive* forces cannot be applied to control *real* deformations.

Realistic control, on the other hand, mainly focuses on the animation of articulated characters. In particular, physically-based controllers for locomotion of humanoids or animals have been proposed by numerous researchers. Their ap-

proaches address walking and running simulation [Girard and Maciejewski, 1985; Raibert and Hodgins, 1991; Yin et al., 2007; Coros et al., 2010] but also flying [Wu and Popović, 2003] and swimming control [Tan et al., 2011]. For such skeleton-based characters, the question often lies in optimizing joint torques to replicate some desired behavior. In the context of biomechanics, activation of internal forces mimicking the action of muscles was also considered [Sifakis et al., 2005; Lee and Terzopoulos, 2006; Sueda et al., 2008]. All these techniques rely on an existing actuation system – based on the articulation of bones, the contraction of muscles at known locations, etc. – inspired by anatomical mechanisms and cannot be easily adapted to the control of actual deformable characters with no obvious skeletal structure such as fantasy creatures, cartoony figures or animatronics. Yet, several applications require a very different kind of actuation system. Deformable puppets used for special effects are typically actuated by external strings. The synthetic skin of robotic characters is generally animated thanks to movable rigid links. Bickel et al. [2012] optimize for the trajectories of actuators attached to the silicon face of an animatronic figure but they had no freedom for modifying the actuation device itself and their results were limited by the motion range of the input actuators. In Chapter 5, we integrate the design of the actuation system to the optimization problem and automatically compute the number and locations of such actuators.

2.4 Shape Optimization

Shape optimization is a highly relevant problem in various areas. The literature related to this topic and connected to our work can be divided into two main categories that we will review alternatively. First, shape optimization techniques can be used to revert the effects of forces acting on deformed objects and thus infer their rest geometries. Second, shape optimization is involved in the task of approximating given shapes by surfaces satisfying specific criteria such as developability, which has been largely investigated in the context of mesh segmentation and parametrization.

2.4.1 Taking physics into account

Physics-based simulation is a forward process in which forces are virtually applied to an object whose initial geometry, or rest pose, is known, in order to estimate its deformation. However, in many applications, we only have a deformed version of the object at hand, whether because the shape of the figure is captured from real data or because it is simply easier to model in such a state. Running the

forward process requires solving the backward problem of computing the rest shape first. Such problems have been studied in various contexts and for different kinds of forces. In biomechanics, the compensation of the effect of gravity and the deformations induced by the scanning process has been investigated by Pathmanathan et al. [2009]. In computer animation, artists create geometry already accounting for the effect of gravity. Derouet-Jourdan et al. [2010] proposed a new technique to fit the parameters of a 2D dynamic rod model in order to match the shape of a given sketched curve. In the same spirit, Twigg and Kačić-Alesić [2011] presented an approach for optimizing the spring lengths of mass spring systems to obtain sag-free simulations. In Chapter 4 and Chapter 6, we optimize for the rest shape of inflatable membranes subject to pressure forces. While our approach also aim at matching a provided target shape, our application exhibits significantly larger deformations and involves forces which depend on the deformed geometry, making the optimization problem particularly challenging.

Resistance to loads is addressed in structural design optimization. In engineering, design of elastic structures with minimal compliance gathered particular attention [Rozvany, 2001]. In a shape optimization problem [Haslinger and Mäkinen, 2003], the goal is to find an optimal shape defined by a prescribed domain. While sizing optimization adapts the thickness of components of a model to meet, for instance, structural stability thresholds without modifying the topology of the structure, topology optimization [Bendsoe and Sigmund, 2004] involves additional features such as the number and location of holes and allow changes in the connectivity of the domain. In computer graphics, realistic structural models with which it is possible to interact are important for convincing physical simulations. In this context, Smith et al. [2002] optimize the geometry and the mass of truss structures for designing bridges, towers and roof supports, while Whiting et al. [2009; 2012] focuses on soundness of masonry constructions. Structural optimization has also been applied to the modeling of trees by Hart et al. [2003], who proposed a tool using static analysis to balance the weight of branches and create realistic plant structures. In Chapter 5, we draw inspiration from topology optimization techniques to infer the number and the shapes of stiff insets to embed in a model. However, we do not optimize for structural stability but instead automatically design a structure that matches given example poses under large deformations.

2.4.2 Developable Approximation

In Chapter 6 we extend our framework to the design of inflatables. Such structures are typically made of flat panels connected to each other. Designing an inflatable involves two steps: (i) a segmentation stage in which the panel layout is defined; (ii) a panel optimization stage in which the shape of the panels is computed. Such a

workflow is similar to the one commonly used in mesh parametrization, for which many techniques have been developed over the years. We refer to the review by Hormann et al. [2007] for an exhaustive description of the existing approaches and will focus on the ones directly related to our work.

Most popular parametrization approaches minimize a distortion energy between the three-dimensional input shapes and the corresponding developable patterns. Typical distortion metrics capture differences in edge lengths, face internal angles and other intrinsic geometric properties.The quality of the final mapping depends on the segmentation of the initial mesh, the more segments are used, the lower the energy typically is. One common requirement in most applications is to keep the number of segments, and cuts in general, as small as possible since seams between adjacent pieces create discontinuities in the parametrization and often translate into visual artefacts. Because of this close relationship between mesh distortion and seam length, some flattening methods deal with cutting and parametrization simultaneously. Sorkine et al. [2002], for example, propose and iterative procedure based on patch growing which automatically partitions the mesh to meet a parametrization quality threshold. Julius et al. [2005] segments models into almost developable patches, which can then be flattened using *ABF++* [Sheffer et al., 2005] to create approximately conformal parametrizations. Their approach extends the Lloyd algorithm employed by Sander et al. [2003] and Cohen-Steiner et al. [2004] for approximating a mesh using planar pieces. Shatz et al. [2006] also use a greedy approach but they start with a seed per face and progressively merge segments into larger clusters approximating planes and cones. All these techniques employ local information to flatten the mesh and do not provide explicit control on the location of the seams nor the shapes of the generated patterns. Yet, higher level considerations like symmetry preservation or semantics-aware segmentation, particularly difficult to handle automatically, are important for the visual appeal of the final result in many applications. This is why we lay aside fully automated approaches in favor of an interactive, optimization-in-the-loop methodology.

A related issue when flattening multiple panels independently is the generation of patterns with incompatible seam lengths, requiring alterations such as pleats or cuts that complicate fabrication and impose a very particular visual appearance. To address this point, Wang [2008] flattens the boundary of the segments first using length-preserving optimization and maps interior regions using intrinsic parametrization [Desbrun et al., 2002]. This approach is well adapted to flatten nearly developable segments but can produce patterns with high distortion in the general case. The method that we introduce in Chapter 6 integrates the fabrication constraint related to the compatible seam lengths into the distortion minimization problem, treating both aspects simultaneously.

Deformable surfaces are largely used in the sheet metal industry and for the

design of free form structures in architecture [Pottmann et al., 2007]. For such applications which require the modeling of perfectly developable surfaces, strip- and quad-based representations are commonly consider. As a subset of ruled-surfaces, developable surfaces can be defined by sweeping out a line along a given directrix curve [do Carmo, 1976]. This representation has been used by Massarwi et al. [2007] and Rose et al. [2007] to construct developable patterns from their boundaries. Mitani and Suzuki [2004] defines the ruled surfaces by approximating a given mesh by a set of triangle strips, which can then be unfolded to produce papercraft objects. The modeling of creases and other singularities in the interior of the developable surface has been addressed by Frey et al. [2004] and Kilian et al.[2008], while Solomon et al. [2012] describe an interactive design system to model discrete developable surfaces consisting of a finite number of rulings. Although these methods work well for designing physically-realizable surfaces made out of paper, developability does not guarantee that the object's surface is in equilibrium under pressure. In our case, we are optimizing for developable (flat) patches, that, when stitched together and then inflated, resemble a 3D shape at a desired equilibrium. However, our fast physics-based simulation approach does not necessarily compute an *inflated surface* with zero Gaussian curvature and therefore does not guarantee a piecewise developable surface. Conversely, developability does not mean stability under pressure – perfectly developable meshes such as cylinders, cubes and double cones, which are trivially flattened (after a cut) with no distortion, are not stable shapes. The metric to use to approximate inflatables must therefore account for the physics of inflation and include effects of pressure forces.

2.5 Fabrication-Oriented Design

The development of tools to facilitate the design of physical objects with desired behaviour recently gained attention in computer graphics. Many concepts, traditionally investigated by the computer graphics community, can indeed be applied to the fabrication of actual artifacts. Research in this field focuses on various aspects ranging from appearance to motion and deformation, including stability and durability.

Manufacture of objects with custom reflectance properties was studied in a number of recent works. A good overview of the different existing approaches can be found in the survey by Hullin et al. [2013]. While Weyrich et al. [2009] optimized surface microgeometry to mill objects exhibiting user-specified appearance, Levin et al. [2013] lately relied on wave optics and a photolithography process to increase the spacial resolution of the constructed BRDFs. The printing of documents with spatially varying reflectance properties using suitable inks and foils was studied

by Matusik et al. [2009]. Frameworks for physical reproduction of printed layered figures [Hasan et al., 2010; Dong et al., 2010] and soft homogeneous materials [Papas et al., 2013] with desired subsurface scattering were also proposed.

Structural soundness is a fundamental requirement of fabricated models for being functional. Stava et al. [2012] proposed an algorithm to improve the strength of a 3D-printed object by automatically detecting weak areas and hollowing, thickening, and inserting struts in the structure, while Umetani and Schmidt [2013] exploited cross-sectional stress analysis to optimize the printing direction. Stability issues were addressed by Prévost et al. [2013] who proposed an interactive system for turning existing models into balanced designs. Physical validity was also investigated in the context of furniture design by Lau et al. [2011] and Umetani et al. [2012].

Motion of 3D-printed characters is another facet which was recently considered. Bächer et al. [2012] and Cali et al. [2012] presented systems for creating functional 3D-printable articulated characters. They used mechanical friction joints to allow their models to hold various poses. Kinematics of more complex mechanisms was later investigated by Zhu et al. [2012], Coros et al. [2013] and Ceylan et al. [2013] who proposed methods to synthesize mechanical automata given the motion of their features as input. Common to all these approaches is that they generate models consisting of static geometry or piecewise *rigid* parts. In contract, we address in Chapter 5 the problem of computing actuated *deformable* characters. Fabrication of deformable objects has been investigated by Bickel et al. in the context of replicating material with desired deformation behavior [2010] and creating synthetic skin for animatronic figures [2012]. The framework presented in this thesis shares some of these goals but is based on a significantly different approach. Like in the first of these works, we also optimize for multiple materials. However, our method relies on a relaxed formulation which allows us to use efficient continuous optimization techniques and is not limited to stacked materials. Moreover, contrary to the previous methods, optimization of the number of actuators, their locations and the forces that they apply is an integral part of our system, providing a much wider design space.

The frameworks for 3D-printing mentioned above address different instances of the more general problem of translating functional requirements to object material descriptions. Chen et al. [2013] proposed an abstraction mechanism to ease the development of new goal-based methods and the reuse of existing algorithms. Their solution combines a reducer tree which parametrizes the space of material assignments with a tuner network which describes and controls the optimization process. While the output of such a system typically consists in per-voxel material composition, input of most 3D-printing softwares is currently limited to per-material surface meshes. In order to fully exploit multi-material printers' capabilities at

high resolution, Vidimče et al. [2013] introduced a fabrication-specific language and a programming pipeline for procedural synthesis of the final material voxels.

Fabrication-oriented design is not limited to the creation of 3D-printed objects. As mentioned above, several pieces of literature on this topic are dedicated to the design of objects made of developable patterns such as paper craft objects [Mitani and Suzuki, 2004; Li et al., 2011], clothing [Okabe et al., 1992; Umetani et al., 2011] and stuffed animals. Mori and Igarashi [2007], in particular, proposed a system, *Plushie*, to interactively model plush toys while satisfying physics constraints. Their framework relied on a simple mass-spring simulation model which was later extended to discrete Kirchhoff triangular elements by Furuta et al. [2010]. The tool we present in Chapter 6 deals with similar inflatable structures but we address the problem from a very different perspective. Whereas *Plushie* projects begin with a blank canvas and helps the user to focus on the modeling task, our projects begin with a given target shape and let the user concentrates on seam placements and panel shapes, essential for obtaining compelling designs. In this sense, the workflow of our system resembles *Pillow*'s [Igarashi and Igarashi, 2008]. However, *Pillow* employs flattening that does not include an inflation-based metric and produces suboptimal results. In contrast, our algorithm optimizes the flat panels using optimization-friendly physics and warrants satisfaction of fabrication requirements. Our inverse modeling approach also shares some similarities with the work of Wang and Tang [2007; 2010] related to the design of medical braces and compression garment. However, their system aims at producing clothing with prescribed strains and normal pressures when our tool focuses on matching an inflated shape while adhering to user-provided seam constraints.

CHAPTER 3

Physics-based Optimization

This chapter introduces the fundamental concepts of physics-based simulation and optimization necessary to the understanding of this thesis. We start by reviewing the basics of the mechanics of deformable objects (Section 3.1), from its continuum formulation (Section 3.1.1) to its discretization (Section 3.1.2). We also derive in Section 3.1.4 the related equations for the case of membranes, that we will use to represent some of the objects that we intend to design. In the second part of this chapter, we present our approach to optimize the system's degrees of freedom while warranting strict satisfaction of physical constraints (Section 3.2). The method that we propose will be tailored to several applications in the next chapters of the thesis. After exposing the generic formulation of our problem (Section 3.2.1), we discuss several techniques that can be used to solve it numerically (Section 3.2.2). Finally, we conclude this chapter with a summary of the presented material (Section 3.3).

3.1 Modeling Deformable Objects

Designing deformable objects requires a solid understanding of the objects' mechanical behaviour. Since our final goal is their fabrication, accurate modeling and simulation of the items that we plan to manufacture is of primary importance. In this context, elasticity theory provides a sound framework to faithfully describe the relations which affect a deformable object's state. We will focus in what follows on its aspects directly related to our work. For a comprehensive exposition of the theory and its numerical treatment we refer to the excellent textbooks by

Bathe [1995], Hughes [2000], and Bonet and Wood [1997]. After introducing the relevant concepts in the idealized view of continuum mechanics in Section 3.1.1, we will address the practical aspects of the computation of the deformable object's shape in Sections 3.1.2 and 3.1.3. The mechanics of membranes will be eventually treated in Section 3.1.4.

3.1.1 Continuum Mechanics

Deformation

The geometry of a deformable object in its undeformed configuration, or rest pose, is typically described by the mapping $\bar{\mathbf{x}} : \Omega \to \Omega_{\bar{\mathbf{x}}} \subset \mathbb{R}^3$ with Ω denoting the object's material domain. When it is subject to external forces, the object *deforms*. Its deformed shape, in turn, is defined by the mapping $\mathbf{x} : \Omega \to \Omega_{\mathbf{x}} \subset \mathbb{R}^3$. In continuum mechanics, it is assumed that Ω is continuous and that the mappings $\bar{\mathbf{x}}$ and $\mathbf{x}$ are invertible and differentiable as many times as desired on Ω.

Let $\hat{\mathbf{x}} : \bar{\mathbf{x}} \mapsto \mathbf{x}$ denote the mapping between $\bar{\mathbf{x}}$ and $\mathbf{x}$. The mapping between the rest configuration $d\bar{\mathbf{x}}$ and the deformed configuration $d\mathbf{x}$ of an infinitesimally small line element of Ω is described by the so-called *deformation gradient* $\mathbf{F} = \frac{\partial \hat{\mathbf{x}}}{\partial \bar{\mathbf{x}}}$, such that

$$d\mathbf{x} = \mathbf{F} d\bar{\mathbf{x}} \tag{3.1}$$

The deformation gradient $\mathbf{F}$ is used to build the *right Cauchy Green tensor* $\mathbf{C}$ defined as

$$\mathbf{C} = \mathbf{F}^t \mathbf{F} \,. \tag{3.2}$$

Using $\mathbf{C}$, the squared length of the linear element in the deformed configuration is given by

$$||d\mathbf{x}||^2 = d\mathbf{x}^t \cdot d\mathbf{x} = d\bar{\mathbf{x}}^t \cdot \mathbf{F}^t \mathbf{F} d\bar{\mathbf{x}} = d\bar{\mathbf{x}}^t \cdot \mathbf{C} d\bar{\mathbf{x}} \tag{3.3}$$

Therefore, the tensor $\mathbf{C}$ can be used to measure how much the material *stretches* during the deformation, i.e how much the length of the linear element changes.

The difference between the squared lengths of the element in its deformed and undeformed configurations,

$$||d\mathbf{x}||^2 - ||d\bar{\mathbf{x}}||^2 = d\mathbf{x}^t \cdot d\mathbf{x} - d\bar{\mathbf{x}}^t \cdot d\bar{\mathbf{x}} = d\bar{\mathbf{x}}^t \cdot (\mathbf{C} - \mathbf{I}) d\bar{\mathbf{x}} \,, \tag{3.4}$$

allows us to introduce the *Green strain tensor*

$$\mathbf{E} = \frac{1}{2}(\mathbf{C} - \mathbf{I}) \,. \tag{3.5}$$

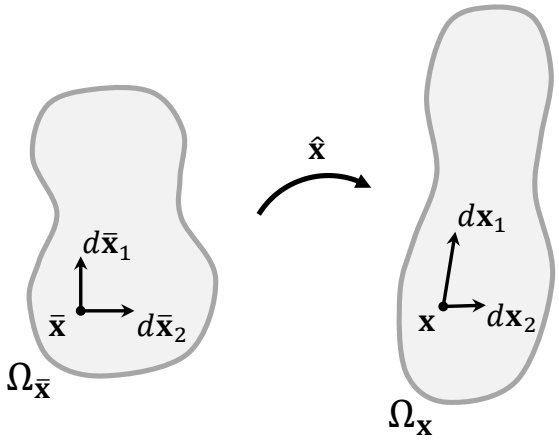


Figure 3.1: *Mapping between the undeformed and deformed configurations of two infinitesimally small line elements.*

The tensors $\mathbf{C}$ and $\mathbf{E}$ also allows to infer how much the material *shears*, i.e. how much the angle between two line elements located at the same material position and pointing towards arbitrary directions changes. This can be deduce from the dot product between the deformed states, $d\mathbf{x}_1$ and $d\mathbf{x}_2$, of the two line elements,

$$d\mathbf{x}_1^t \cdot d\mathbf{x}_2 = d\bar{\mathbf{x}}_1^t \cdot \mathbf{C} d\bar{\mathbf{x}}_2 \,, \tag{3.6}$$

and its difference with the dot product of the undeformed states of the two elements,

$$d\mathbf{x}_1^t \cdot d\mathbf{x}_2 - d\bar{\mathbf{x}}_1^t \cdot d\bar{\mathbf{x}}_2 = d\bar{\mathbf{x}}_1^t \cdot (\mathbf{C} - \mathbf{I}) d\bar{\mathbf{x}}_2 = d\bar{\mathbf{x}}_1^t \cdot (2\mathbf{E}) d\bar{\mathbf{x}}_2 \,. \tag{3.7}$$

It is worthwhile to note that $\mathbf{C}$, and therefore also $\mathbf{E}$, are symmetric and independent to global rotations. This becomes evident when decomposing $\mathbf{F}$ in its polar form as

$$\mathbf{F} = \mathbf{R}\mathbf{U} \tag{3.8}$$

where $\mathbf{R}$ is a rotation matrix and $\mathbf{U}$ a positive semi-definite matrix, which allows us to write $\mathbf{C}$ as

$$\mathbf{C} = \mathbf{F}^t\mathbf{F} = \mathbf{U}^t\mathbf{R}^t\mathbf{R}\mathbf{U} = \mathbf{U}^t\mathbf{U} \,. \tag{3.9}$$

Constitutive Model

The deformation behaviour of an elastic material is typically described by a constitutive equation, i.e. the relation between the material deformation and the internal forces acting to restore the material to its undeformed state. In the case of *hyperelastic* materials, such as the ones that we consider in this thesis, these forces do not depend on the deformation path and can therefore be derived from an energy

potential W^{int} which depends only on the deformation gradient $\mathbf{F}$. This energy potential is generally defined using an energy density Ψ which is integrated over the reference domain $\Omega_{\bar{\mathbf{x}}}$ as

$$W^{\text{int}} = \int_{\Omega_{\bar{\mathbf{x}}}} \Psi(\mathbf{F}) d\Omega_{\bar{\mathbf{x}}} \,. \tag{3.10}$$

When the material is additionally *isotropic*, the deformation energy density Ψ can be defined using the right Cauchy Green tensor $\mathbf{C}$ as a function of its invariants,

$$\begin{aligned} I_1 &= \text{tr}(\mathbf{C}) = \lambda_1^2 + \lambda_2^2 + \lambda_3^2 \,, \\ I_2 &= \tfrac{1}{2}(\text{tr}(\mathbf{C})^2 - \text{tr}(\mathbf{C}^t\mathbf{C})) = \lambda_1^2\lambda_2^2 + \lambda_2^2\lambda_3^2 + \lambda_3^2\lambda_1^2 \,, \\ I_3 &= \det(\mathbf{C}) = \lambda_1^2\lambda_2^2\lambda_3^2 \,, \end{aligned} \tag{3.11}$$

where λ_1, λ_2 and λ_3 are the eigenvalues of $\mathbf{F}$.

Furthermore, many materials such as rubbers are also characterized by their strong resistance to volume changes. Such materials are often approximated by *perfectly incompressible* materials for which $I_3 = 1$. This requirement is generally integrated into the formulation of the deformation energy density which then depends only on I_1 and I_2.

Equilibrium

When a deformable body is subject to external loads, its resulting deformation generates internal forces which counteract these external forces. For a hyperelastic body, these forces are conservative and are obtained from the derivatives of the energy potential W^{int} as

$$\mathbf{f}^{\text{int}}(\mathbf{x}) = -\frac{\partial W^{\text{int}}(\mathbf{x})}{\partial \mathbf{x}} \,. \tag{3.12}$$

The object reaches a *static equilibrium* state when point-wise balance of internal and external forces is achieved for all points of the body. By letting $\mathbf{f}^{\text{ext}}(\mathbf{x})$ denote the sum of all the external forces exerted at position $\mathbf{x}$, this condition can be written as

$$\mathbf{f}^{\text{ext}}(\mathbf{x}) = -\mathbf{f}^{\text{int}}(\mathbf{x}) \,, \mathbf{x} \in \Omega_{\mathbf{x}} \,. \tag{3.13}$$

Note that this state corresponds to a configuration which locally minimizes the energy of the system. When the external forces are also conservative, as it is the case, e.g., with gravitational forces or pressure forces, the external forces derive from a energy potential W^{ext} which can be added to W^{int}. Solving for the balance of forces (3.13) is then equivalent to minimizing the total energy W defined as

$$W = W^{\text{int}} + W^{\text{ext}}. \tag{3.14}$$

Solving this problem numerically requires a spatial discretization scheme, which is described next.

3.1.2 Discretization

We have considered so far the deformation of an object in the continuous setting, in which the object's deformed state was characterized by the equilibrium condition (3.13). This relation defines a system of partial differential equations (PDEs) whose unknowns form a continuous vector field. Solving this type of equations exactly is only possible for very simple problems, for which an analytical formulation of the solution exists. In the much more general case, one generally seeks to obtain an *approximation* of the true solution instead by discretizing the spatial domain and assuming a certain form of the solution that can be parametrized by a finite number of variables. This constitutes the essence of the Finite Element Method (FEM). We will now derive the discrete version of (3.13) arising from the application of the method to our specific problem.

As a starting point, we approximate the geometry of the object with a tetrahedron mesh with n vertices whose deformed and undeformed positions we denote by $\mathbf{x}_i$, respectively $\bar{\mathbf{x}}_i$, for $1 \leq i \leq n$. Let $\bar{\mathbf{x}}_j^e$ and $\mathbf{x}_j^e$, $0 \leq j \leq 3$, denote the vertex positions pertaining to a given element e and define corresponding edge vectors $\bar{\mathbf{e}}_k = \bar{\mathbf{x}}_k^e - \bar{\mathbf{x}}_0^e$ and $\mathbf{e}_k = \mathbf{x}_k^e - \mathbf{x}_0^e$, $1 \leq k \leq 3$. We will assume that the deformation is constant across each tetrahedron so that the geometry of a given deformed element is described by a single linear mapping $\mathbf{F}^e \in \mathbb{R}^{3\times 3}$, the deformation gradient. In this setting, the element's deformed geometry can be expressed in terms of its undeformed state as

$$[\mathbf{e}_1\ \mathbf{e}_2\ \mathbf{e}_3] = \mathbf{F}^e\,[\bar{\mathbf{e}}_1\ \bar{\mathbf{e}}_2\ \bar{\mathbf{e}}_3]\ , \tag{3.15}$$

from which we obtain the deformation gradient as

$$\mathbf{F}^e = [\mathbf{e}_1\ \mathbf{e}_2\ \mathbf{e}_3]\,[\bar{\mathbf{e}}_1\ \bar{\mathbf{e}}_2\ \bar{\mathbf{e}}_3]^{-1}\ . \tag{3.16}$$

From the deformation gradient $\mathbf{F}^e$ we write the discrete Cauchy Green tensor as

$$\mathbf{C}^e = (\mathbf{F}^e)^t\mathbf{F}^e\ . \tag{3.17}$$

Following the Total-Lagrangian finite element formulation [Bathe, 1995], we compute the elastic energy of a deformed element by integrating Ψ over the undeformed domain. Since $\mathbf{F}^e$ is constant, we simply have

$$W^e(\mathbf{F}^e) = \int_{\bar{V}^e} \Psi(\mathbf{F}^e)\,dV = \Psi(\mathbf{F}^e)\cdot\bar{V}_e\ , \tag{3.18}$$

where $\bar{V}^e$ is the volume of the undeformed element. The total elastic energy of the object is obtained by summing up elemental contributions as

$$W^{\text{int}} = \sum_e W^e\ . \tag{3.19}$$

Nodal internal forces are then obtained by deriving the internal energy such that

$$\mathbf{f}_i^{\text{int}} = -\frac{\partial W^{\text{int}}}{\partial \mathbf{x}_i} \in \mathbb{R}^3 \ , \ 1 \leq i \leq n \ . \tag{3.20}$$

In the discrete setting, the applied loads are lumped to the vertices of the mesh. The static equilibrium of the object is then characterized by nodal force balance as

$$\mathbf{f}_i^{\text{ext}} = -\mathbf{f}_i^{\text{int}} \ , \ 1 \leq i \leq n \ , \tag{3.21}$$

where $\mathbf{f}_i^{\text{ext}}$ denote the sum of all external forces applied to the vertex located at position $\mathbf{x}_i$.

3.1.3 Numerics

We solve for the deformed vertex positions satisfying Equation (3.21) using a Newton-Raphson procedure [Nocedal and Wright, 2000] that we describe below. In this section, we assume that $\mathbf{x} \in \mathbb{R}^{3n}$ denote the vector of concatenated vertex positions $\mathbf{x}_i$, $1 \leq i \leq n$ and $\mathbf{f}^{\text{ext}} \in \mathbb{R}^{3n}$, respectively $\mathbf{f}^{\text{int}} \in \mathbb{R}^{3n}$, the concatenated vector of external, respectively internal, nodal forces. We define as $\mathbf{f} = \mathbf{f}^{\text{ext}} + \mathbf{f}^{\text{int}}$ the total nodal forces.

Starting from an initial guess $\mathbf{x}^0$ for the solution of Equation (3.21), e.g. the rest pose of the object, the Newton-Raphson scheme iteratively correct the current root estimation $\mathbf{x}^k$ by computing a new vector $\mathbf{x}^{k+1}$ that approximately solve the system

$$\mathbf{f}(\mathbf{x}^{k+1}) = 0. \tag{3.22}$$

Using a first order Taylor series, $\mathbf{f}$ can be expanded around $\mathbf{x}^k$ as

$$\mathbf{f}(\mathbf{x}^{k+1}) \approx \mathbf{f}(\mathbf{x}^k) + \frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}}(\mathbf{x}^{k+1} - \mathbf{x}^k) \ . \tag{3.23}$$

Approximating Equation (3.22) by

$$\mathbf{f}(\mathbf{x}^k) + \frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}}(\mathbf{x}^{k+1} - \mathbf{x}^k) = 0 \ , \tag{3.24}$$

allows us to obtain a new estimate

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \left[\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}}\right]^{-1} \mathbf{f}(\mathbf{x}^k) \ . \tag{3.25}$$

This procedure is repeated until the norm of $\mathbf{f}$ corresponding to the current estimate falls beyond a given threshold.

In practice, we do not compute the inverse of the matrix $\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}}$. Instead, we directly solve the linear system

$$\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}} d\mathbf{x} = -\mathbf{f}(\mathbf{x}^k) \,, \tag{3.26}$$

and update the new iterate as

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha d\mathbf{x} \,. \tag{3.27}$$

The stepsize $0 \leq \alpha \leq 1$ is determined using a backtracking approach [Nocedal and Wright, 2000] so as to guarantee a decrease in the total potential energy of the system. Note that when the matrix $\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}}$ is not positive definite, $d\mathbf{x}$ might be an ascent direction. In that case, the matrix is regularized by adding to it a multiple of the identify matrix $\beta\mathbf{I}$, where $\beta > 0$ is progressively increased until a valid direction is found.

3.1.4 Membranes

Continuum Formulation

When the thickness of the object is negligible compared to its other dimensions, the quantities introduced above can be obtained from a reduced formulation in a two-dimensional space using a surface-centered representation. To this end, we focus on the deformation of the thin shell's middle surface and let the mappings $\bar{\mathbf{x}} : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^3$ and $\mathbf{x} : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^3$ describe the surface in its undeformed and deformed configurations.

Typical thin shell materials undergo both stretching and bending deformation. However, the thin shell structures that we consider in this thesis either exhibit extremely large deformation (Chapter 4) or are made of quasi-inextensible materials that largely resist stretching but are compliant to bending (Chapter 6). In both cases, we assume that the resulting in-plane forces overrule all bending contributions. Hence, the curvature of the surface is not important and only the stretching, i.e., the membrane deformation needs to be quantified.

Let $(u, v) \in \Omega$ denote the material coordinates of a given point of the middle surface. We start by introducing tangent vectors on the undeformed surface

$$\bar{\mathbf{a}}_u = \frac{\partial \bar{\mathbf{x}}}{\partial u} \,, \quad \text{and} \quad \bar{\mathbf{a}}_v = \frac{\partial \bar{\mathbf{x}}}{\partial v} \,, \tag{3.28}$$

and analogously define tangents $\mathbf{a}_u$ and $\mathbf{a}_v$ on the deformed surface

$$\mathbf{a}_u = \frac{\partial \mathbf{x}}{\partial u} \,, \quad \text{and} \quad \mathbf{a}_v = \frac{\partial \mathbf{x}}{\partial v} \,. \tag{3.29}$$

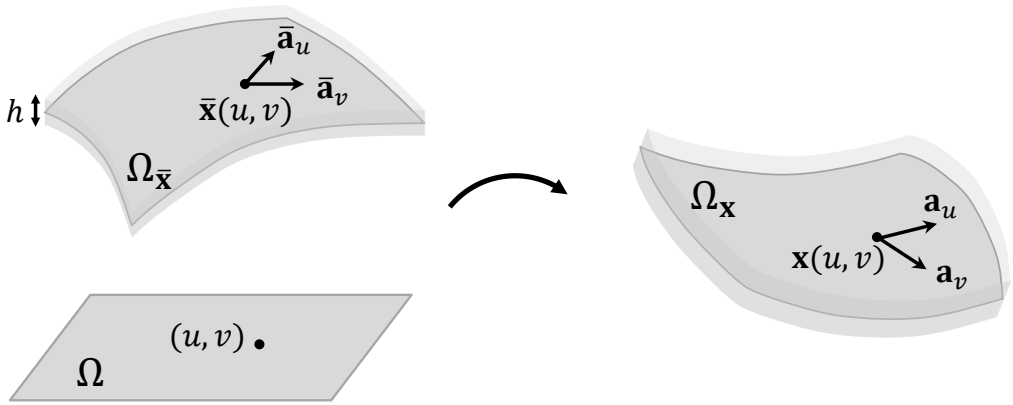


Figure 3.2: *Thin shell represented by its middle surface in its undeformed (*left*) and deformed (*right*) configurations and material domain (*bottom*).*

The inner products of the tangents give rise to the 2×2 metric tensors $\bar{\mathbf{C}}$ and $\tilde{\mathbf{C}}$ whose components are defined as

$$\bar{\mathbf{C}}_{uv} = \bar{\mathbf{a}}_u^T \bar{\mathbf{a}}_v \,, \quad \text{respectively} \quad \tilde{\mathbf{C}}_{uv} = \mathbf{a}_u^T \mathbf{a}_v \,. \tag{3.30}$$

For simplicity, we will assume that the parametrization for the undeformed configuration is isometric such that $\bar{\mathbf{a}}_u$ and $\bar{\mathbf{a}}_v$ are orthonormal and $\bar{\mathbf{C}}$ is the 2×2 identity matrix, $\tilde{\mathbf{C}} = \mathbf{I}$. This allows us to recover the 2×2 Green strain

$$\mathbf{E} = \tfrac{1}{2}(\tilde{\mathbf{C}} - \mathbf{I}) \,,$$

which describes the surface deformation. Although this measure does not explicitly account for deformations in the thickness direction, we can infer this information by introducing two *kinematic assumptions* [Bonet et al., 2000]: First, the surface should not exhibit transverse shearing, and second, the deformation should be volume-preserving. The first requirement is part of the Kirchhoff-Love assumptions for thin shell models, while the second one accounts for the incompressible nature of rubber materials. Based on these assumptions, we construct the 3×3 right Cauchy Green tensor as

$$\mathbf{C} = \begin{bmatrix} \tilde{\mathbf{C}}_{uu} & \tilde{\mathbf{C}}_{uv} & 0 \\ \tilde{\mathbf{C}}_{vu} & \tilde{\mathbf{C}}_{vv} & 0 \\ 0 & 0 & J^{-1} \end{bmatrix} , \tag{3.31}$$

where $J = \det \tilde{\mathbf{C}}$ is the determinant of the two-dimensional Cauchy Green tensor $\tilde{\mathbf{C}}$.

Discretization

We will now proceed to the spatial discretization. The derivations closely follow the general case described in Section 3.1.2 and we will use similar notations.

In accordance with the assumptions of plane stress and negligible bending forces, we settle for a finite element discretization centered around membrane elements. Similar to the general case, we assume constant deformation throughout each element and use linear triangular elements, so called constant strain triangles [Bathe, 1995]. We assume all elements have a constant thickness h and represent the geometry of the object's middle surface by a triangle mesh with n vertices. We denote by $\mathbf{x}_i$, respectively $\bar{\mathbf{x}}_i$, $1 \leq i \leq n$, the positions of the vertices in the deformed, respectively undeformed, configurations. Let $\bar{\mathbf{x}}_j^e$ and $\mathbf{x}_j^e$, $0 \leq j \leq 2$, denote the vertex positions of a given element e and define corresponding edge vectors $\bar{\mathbf{e}}_k = \bar{\mathbf{x}}_k^e - \bar{\mathbf{x}}_0^e$ and $\mathbf{e}_k = \mathbf{x}_k^e - \mathbf{x}_0^e$. We endow the undeformed configuration with an orthonormal material frame $\bar{\mathbf{T}} = [\bar{\mathbf{u}}\ \bar{\mathbf{v}}\ \bar{\mathbf{d}}] \in \mathbb{R}^{3\times 3}$, where

$$\bar{\mathbf{u}} = \frac{\bar{\mathbf{e}}_1}{\|\bar{\mathbf{e}}_1\|}\,, \quad \bar{\mathbf{d}} = \frac{\bar{\mathbf{u}} \times \bar{\mathbf{e}}_2}{\|\bar{\mathbf{u}} \times \bar{\mathbf{e}}_2\|}\,, \quad \bar{\mathbf{v}} = \bar{\mathbf{d}} \times \bar{\mathbf{u}}\,, \tag{3.32}$$

such that $\bar{\mathbf{u}}$ and $\bar{\mathbf{v}}$ span the plane of the element and $\bar{\mathbf{d}}$, the director, is its unit-length normal vector. Note that when using isotropic materials as in our case, the choice of the frame $\bar{\mathbf{u}}$, $\bar{\mathbf{v}}$ is not important and that frames do not have to be consistently oriented across elements. Consequently, there is no need for constructing (and tracking) a parametrization of the rest shape.

We can conveniently integrate the kinematic assumptions of no transverse shear and incompressibility into the definition of $\mathbf{F}^e$ by constraining the deformed director $\mathbf{d}$ as

$$\mathbf{d} = \frac{\mathbf{e}_1 \times \mathbf{e}_2}{\|\mathbf{e}_1 \times \mathbf{e}_2\|^2}\,. \tag{3.33}$$

This requires the deformed director to be normal to the deformed element (no transverse shear) and to be stretched such as to balance the change in area (incompressibility). The mapping between the undeformed and deformed elements is then given by

$$[\mathbf{e}_1\ \mathbf{e}_2\ \mathbf{d}] = \mathbf{F}^e \bar{\mathbf{T}}^{-1} \left[\bar{\mathbf{e}}_1\ \bar{\mathbf{e}}_2\ \bar{\mathbf{d}}\right]\,, \tag{3.34}$$

which allows us to write the deformation gradient as

$$\mathbf{F}^e = [\mathbf{e}_1\ \mathbf{e}_2\ \mathbf{d}] \left[\bar{\mathbf{e}}_1\ \bar{\mathbf{e}}_2\ \bar{\mathbf{d}}\right]^{-1} \bar{\mathbf{T}}\,. \tag{3.35}$$

See Figure 3.3 for an illustration of these transformations. Note that the multiplication by $\bar{\mathbf{T}}$ aligns the material frame of the element such that the first two columns of $\mathbf{F}^e$ describe in-plane deformation while the third column corresponds to the thickness direction.

Similar to (3.17), the Cauchy Green tensor is directly obtained by $\mathbf{C}^e = (\mathbf{F}^e)^T \mathbf{F}^e$. Note that, by construction, $\mathbf{C}^e$ has the same special structure as its continuous counterpart in (3.31).

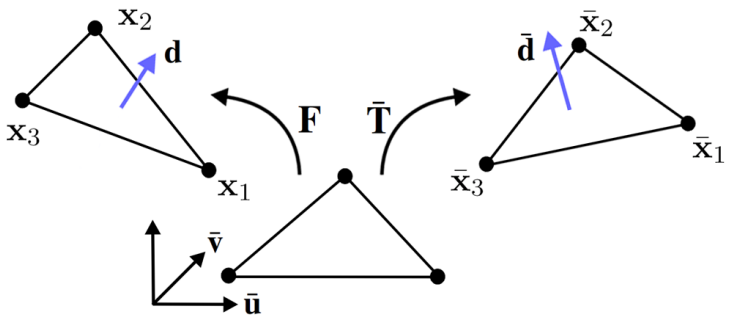


Figure 3.3: *Transformations between deformed* (left)*, undeformed* (right) *and material* (middle) *domains for a membrane element.*

Lastly, we obtain the elastic energy of the deformed element as

$$W^e(\mathbf{C}^e) = \int_{\bar{V}^e} \Psi(\mathbf{C}^e)\, dV = \Psi(\mathbf{C}^e) \cdot h\bar{A}_e\,, \tag{3.36}$$

where $\bar{A}^e$ is the area of the undeformed element.

Computation of the equilibrium position of the deformed membrane is performed as described in Section 3.1.3.

3.2 Generic Optimization Scheme

The first part of this chapter focused on the computation of an object's deformed shape when its undeformed state, material behaviour and applied forces were known parameters. This is commonly called the *forward problem*. Solving the forward problem is the concern of typical physics-aware modeling tools that allow the user to modify some system parameters – usually at interactive rates – to explore the space of feasible deformations. While these design tools offer great capabilities to create novel models, tuning forces or material variables can be extremely tedious when the ultimate goal consist in recovering a very precise shape. The design systems that we propose in the thesis rely on the opposite approach: our objective is to let the user play with the envisioned deformed shapes, possibly in a preprocessing stage and regardless of any physics consideration, and to automatically infer the values of the parameters which lead to the target deformations. In short, we focus on the *inverse problem*. With this aim in mind, we will explain in this section our strategy for computing optimal deformed shapes that match provided target shapes, which we cast as a nonlinear constrained optimization problem. We start by formalizing the problem (Section 3.2.1), then discuss the possible techniques to proceed to its solution via constrained nonlinear optimization (Section 3.2.2).

3.2.1 Problem Formulation

We assume that we want to design a deformable object whose desired behaviour is described by a set of n_p target shapes. To this end, we are free to modify some system variables that affect the object's deformed poses. These variables can be of any sort – base materials, applied loads, rest shape, etc. – and we will discuss several options in the following chapters of this thesis. For the sake of generality, we simply assume here that they have been clearly defined and parametrized and let $\mathbf{p}$ denote the vector of generic parameters to optimize. Our goal is now to find the values of the parameters $\mathbf{p}$ that lead to deformed states approximating the target shapes as closely as possible. As it is generally not possible to exactly match the target poses, we ask that the distance of each deformed pose $\mathbf{x}^i$ to its corresponding target pose $\mathbf{t}^i$ be minimized. We quantify *closeness* using a distance energy function that measures differences in positions on the model's boundary between deformed $\mathbf{x}^i$ and target poses $\mathbf{t}^i$ as

$$E_d = \sum_i E_d^i(\mathbf{t}^i, \mathbf{x}^i) \; . \tag{3.37}$$

Note that the distance energy is not a physical energy – we cannot simply add it to the energy of the system since this would lead to minima that do not correspond to force equilibrium configurations. Instead, we seek to find the physical solutions, i.e, configurations in force equilibrium, which are closest to the target shapes, i.e., minimize the distance energy. This can be formulated as a constrained optimization problem,

$$\begin{aligned} \min_{\mathbf{x}^i, \mathbf{p}} \quad & \sum_i^{n_p} E_d^i(\mathbf{t}^i, \mathbf{x}^i) \\ \text{s.t} \quad & \mathbf{f}_{\text{int}}^i(\mathbf{x}^i, \mathbf{p}) = -\mathbf{f}_{\text{ext}}^i(\mathbf{x}^i, \mathbf{p}) \quad \forall i \in 1 \dots n_p \; , \end{aligned} \tag{3.38}$$

where $\mathbf{f}_{\text{int}}^i$, respectively $\mathbf{f}_{\text{ext}}^i$, denote the internal, respectively external, forces corresponding to pose i. The objective function prefers deformed poses $\mathbf{x}^i$ that are close to their target counterparts, whereas the constraints require that each of the $\mathbf{x}^i$ is a physically-feasible solution, i.e., represents an equilibrium state in which the internal forces $\mathbf{f}_{\text{int}}^i$ are in balance with the externally applied forces $\mathbf{f}_{\text{ext}}^i$.

Letting $\mathbf{x} \in \mathbb{R}^{n_v \cdot n_p}$ denote the concatenation of the n_p deformed position vectors $\mathbf{x}^i$ and $\mathbf{f}(\mathbf{x}, \mathbf{p}) = \mathbf{f}_{\text{int}}(\mathbf{x}, \mathbf{p}) + \mathbf{f}_{\text{ext}}(\mathbf{x}, \mathbf{p})$, the sum of the concatenated force vectors, the minimization problem (3.38) can be more compactly rewritten as

$$\begin{aligned} \min_{\mathbf{x}, \mathbf{p}} \quad & E_d(\mathbf{x}) \\ \text{s.t} \quad & \mathbf{f}(\mathbf{x}, \mathbf{p}) = 0 \; . \end{aligned} \tag{3.39}$$

We describe in the next section how this problem can be solved numerically.

3.2.2 Nonlinear Constrained Minimization

The minimization problem described by (3.39) has a large number of degrees of freedom as well as numerous nonlinear constraints which have to be satisfied exactly. The most widely used approaches to solve such kinds of problems include the quadratic penalty method, sequential quadratic programming and the augmented Lagrangian method [Nocedal and Wright, 2000]. We will review and discuss below these different techniques.

Quadratic Penalty Method

The quadratic penalty method enjoy widespread popularity in many domains, largely because of its simplicity. In this method, the constraints are directly added to the initial objective to form an augmented objective function of the form

$$Q(\mathbf{x}, \mathbf{p}) = E_d(\mathbf{x}) + \frac{\mu}{2}||\mathbf{f}(\mathbf{x}, \mathbf{p})||^2 , \quad \mu > 0 . \tag{3.40}$$

The function (3.40) is then minimized for increasing values of μ using standard techniques for unconstrained optimization such as the Newton's method described in Section (3.1.3).

Although this technique is appealing at first, it actually suffers from major drawbacks. Indeed, it is worthwhile to note that a minimum of (3.40) coincide with a solution of the original problem (3.39) only when the satisfaction of the constraints does not conflict with a decrease in the original objective. In the opposite – and much more general – case, the amount of constraint violation depends on the penalty parameter μ, a feasible minimizer corresponding to an infinite value for μ. For want of getting a perfectly valid solution, one might be tempted to use large values of μ so as to decrease constraint violations. Unfortunately this also increase the ill-conditioning of the Hessian of Q leading to severe numerical issues when solving attendant linear systems and eventually results in poor convergence of most minimization algorithms. We will see later in this section that this fundamental issue can be largely reduced with very little computation and implementation overheads by adding an extra term to the augmented objective function (3.40).

Sequential Quadratic Programming

Sequential quadratic programming (SQP) is probably one of the most effective methods to solve constrained minimization problems, especially when the constraints exhibit significant nonlinearities. The core idea of SQP consists in generating a sequence of steps by solving successive quadratic programs that approximate the original problem.

We start by introducing the Lagrangian function $\mathcal{L}$ corresponding to our specific problem (3.39) as

$$\mathcal{L}(\mathbf{x}, \mathbf{p}, \boldsymbol{\lambda}) = E_d(\mathbf{x}) - \boldsymbol{\lambda}^t \mathbf{f}(\mathbf{x}, \mathbf{p}) \,, \tag{3.41}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^{3n_v}$ is a vector of additional variables named *Lagrange multipliers*. It is well-known from optimization theory [Nocedal and Wright, 2000] that the minimizers $\mathbf{y}^* = (\mathbf{x}^*, \mathbf{p}^*)$ of (3.39) should satisfy the so-called Karush-Kuhn-Tucker (KKT) conditions

$$\begin{array}{c} \nabla_{\mathbf{y}} \mathcal{L}(\mathbf{y}^*, \boldsymbol{\lambda}^*) = 0 \\ \mathbf{f}(\mathbf{y}^*) = 0 \end{array} \,, \tag{3.42}$$

where $\boldsymbol{\lambda}^*$ are the Lagrange multipliers corresponding to the solution $\mathbf{y}^*$. The application of the Newton's method to the root finding of the system of equations (3.42) allows us to compute iterates of the form

$$\begin{pmatrix} \mathbf{y}^{k+1} \\ \boldsymbol{\lambda}^{k+1} \end{pmatrix} = \begin{pmatrix} \mathbf{y}^{k} \\ \boldsymbol{\lambda}^{k} \end{pmatrix} + \alpha \begin{pmatrix} d\mathbf{y} \\ d\boldsymbol{\lambda} \end{pmatrix} \,, \quad 0 \leq \alpha \leq 1 \,, \tag{3.43}$$

where each Newton step solves the KKT system

$$\begin{bmatrix} \mathbf{H}(\mathbf{y}^k) & -\mathbf{J}^t(\mathbf{y}^k) \\ \mathbf{J}(\mathbf{y}^k) & 0 \end{bmatrix} \begin{pmatrix} d\mathbf{y} \\ d\boldsymbol{\lambda} \end{pmatrix} = - \begin{pmatrix} \mathbf{g}(\mathbf{y}^k) - \mathbf{J}^t(\mathbf{y}^k)\boldsymbol{\lambda}^k \\ \mathbf{f}(\mathbf{y}^k) \end{pmatrix} \,, \tag{3.44}$$

with $\mathbf{H} = \nabla^2_{\mathbf{yy}} \mathcal{L}$ the Hessian of the Lagrangian, $\mathbf{J}$ the Jacobian of $\mathbf{f}$ and $\mathbf{g} = \nabla_{\mathbf{y}} E_d$ the gradient of the objective. If $\mathbf{J}$ has full rank and the matrix $\mathbf{H}$ is positive definite on the tangent space of the constraints, i.e

$$\mathbf{d}^t \mathbf{H} \mathbf{d} > 0, \; \forall \mathbf{d} \;\text{ s.t. }\; \mathbf{J}\mathbf{d} = 0 \,, \tag{3.45}$$

the KKT system (3.44) has a unique solution. In this case, the Newton step generated by (3.44) actually corresponds to the solution of the minimization problem

$$\begin{array}{ll} \min\limits_{d\mathbf{y}} & E_d(\mathbf{y}^k) + \mathbf{g}(\mathbf{y}^k)^t d\mathbf{y} + \frac{1}{2} d\mathbf{y}^t \mathbf{H}(\mathbf{y}^k, \boldsymbol{\lambda}^k) d\mathbf{y} \\ \text{s.t} & \mathbf{J}(\mathbf{y}^k) d\mathbf{y} + \mathbf{f}(\mathbf{y}^k) = 0 \,, \end{array} \tag{3.46}$$

which locally models the original problem (3.39) by a quadratic program. Hence the name of the method.

As with unconstrained optimization, proper regularization of the matrix $\mathbf{H}$ and careful control of the stepsize α are crucial to guarantee global convergence of the iteration scheme (3.43). Since reduction of the objective function might increase the violation of the constraints, and vice versa, many line search strategies seek to decrease a merit function which combines both objective and constraints. A popular merit function, that we will use in this work, is

$$m(\mathbf{y}) = E_d(\mathbf{y}) + \mu \left\| \mathbf{f}(\mathbf{y}) \right\|_1 , \; \mu > 0 \,. \tag{3.47}$$

It can be shown that when the conditions (3.45) hold the merit function (3.47) can – after a proper update of μ – always be decreased. When $\mathbf{y}_k$ is not close enough to the final solution, this might not be the case and $\mathbf{H}$ needs to be modified. Checking for the conditions (3.45) may be difficult without factoring the constraint Jacobian $\mathbf{J}$ and the Lagrangian Hessian $\mathbf{H}$. However, regularizing the matrix $\mathbf{H}$ whenever it is not positive definite may unnecessarily hinder progress towards the solution. Byrd et al. recently tackled this problem and proposed an effective scheme to efficiently solve minimization problems with nonconvex objective and/or nonconvex constraints [2010]. The essence of the method consists in monitoring, at every iteration, the reduction of a local approximation of the merit function, defined as

$$m_k(d\mathbf{y};\mu) = E_d(\mathbf{y}_k) + \mathbf{g}(\mathbf{y}^k)^t d\mathbf{y} + \mu_k \left\| \mathbf{f}(\mathbf{y}_k) + \mathbf{J}(\mathbf{y}^k) d\mathbf{y} \right\|_1 , \tag{3.48}$$

in order to arbitrate between accepting the step, updating the merit function or correcting the Hessian. For each minimization step, the algorithm evaluates the convexity of the problem by estimating the norms – without computing the decomposition itself – of the tangential component $\mathbf{t}$ and the normal component $\mathbf{n}$ of the generated step direction $d\mathbf{y}$ satisfying

$$d\mathbf{y} = \mathbf{n} + \mathbf{t} , \;\; \mathbf{J}\mathbf{t} = 0, \; , \; \mathbf{t} \cdot \mathbf{n} = 0 , \tag{3.49}$$

and by checking the condition

$$\frac{1}{2} d\mathbf{y}^t \mathbf{H}(\mathbf{y}_k) d\mathbf{y} \geq \theta \gamma_k , \;\; \nu_k = \frac{\|\mathbf{J} d\mathbf{y}\|^2}{\|\mathbf{J}\|^2} , \;\; \gamma_k = \|d\mathbf{y}\|^2 - \nu_k , \tag{3.50}$$

where $\theta > 0$ is a small constant and ν_k, resp. γ_k, is a lower bound, resp. upper bound, for the squared norm of $\mathbf{n}$, resp. $\mathbf{t}$. If the problem is found to be sufficiently convex, i.e. if the condition (3.50) is satisfied, or if the step is sufficiently normal, i.e. if ν_k is significant, then μ_k is updated by requiring that

$$\mu_k \geq \tau \frac{\mathbf{g}(\mathbf{y}^k)^t d\mathbf{y} + \max\left\{ \frac{1}{2} d\mathbf{y}^t \mathbf{H}(\mathbf{y}_k) d\mathbf{y} \, , \theta \gamma_k \right\}}{\|\mathbf{f}(\mathbf{y}_k)\| - \left\| \mathbf{f}(\mathbf{y}_k) + \mathbf{J}(\mathbf{y}^k) d\mathbf{y} \right\|} , \;\; \tau > 1 , \tag{3.51}$$

so as to guarantee a decrease in the merit function. In the opposite case, the problem is considered to be insufficiently convex and $\mathbf{H}$ is progressively regularized until one of the two conditions above is satisfied. The stepsize α is then computed thanks to a standard backtracking approach using the merit function (3.48).

Note that when employing SQP, in Chapter 6, we used *Pardiso* [Schenk and Gärtner, 2006], a direct solver based on fast pivoting to solve the KKT system but that the approach presented by Byrd et al. can cope with significant residual errors when solving the linear system and is therefore compatible with the use of iterative solvers based on Krylov subspace methods .

Augmented Lagrangian Method

We have seen that solving the constrained minimization problem with SQP breaks down to finding roots of the gradient of (3.39) using Newton's method. This promises accurate constraint satisfaction but also leads to large indefinite systems of equations which are costly to solve. The quadratic penalty method, by contrast, does not increase the dimension of the system but strict constraint satisfaction, mandatory in our case, leads to ill-conditioned systems and thus numerical problems. As a hybrid between SQP and penalty methods, augmented Lagrangian methods (ALMs) strive to combine the advantages of both: they offer accurate constraint satisfaction without ill-conditioning (unlike the penalty method) but do not entail an increase in dimension (unlike SQP). ALMs are obtained by *augmenting* the Lagrangian of the constrained minimization problem (3.39) by an additional penalty term similar to the one used in quadratic penalty methods. For our specific optimization problem, the augmented objective function has the following form,

$$\mathcal{L}(\mathbf{x}, \mathbf{p}; \mu) = E_d(\mathbf{x}) - \boldsymbol{\lambda}^t \mathbf{f}(\mathbf{x}, \mathbf{p}) + \frac{\mu}{2} ||\mathbf{f}(\mathbf{x}, \mathbf{p})||^2 \,. \tag{3.52}$$

This new objective function is minimized iteratively by alternating between unconstrained minimization and multiplier update steps . In the first step of a given iteration, the Lagrange multipliers are kept fixed and $\mathcal{L}$ is minimized with respect to the free variables $\mathbf{x}$ and $\mathbf{p}$ using, e.g, the standard Newton-Raphson method with line search described in Section (3.1.3). After each minimization step, the Lagrange multipliers are updated according to

$$\boldsymbol{\lambda}^k = \boldsymbol{\lambda}^k - \mu^k \mathbf{f}^k \,, \tag{3.53}$$

provided that the decrease in the constraints is sufficient. Otherwise, the weight of the penalty term μ^k is increased while keeping the multipliers unchanged. The procedure stops when both the gradients of $\mathcal{L}$ and of the constraints $\mathbf{f}$ are smaller than given thresholds. It can be proven that when mu is sufficiently large, the Lagrange multipliers $\boldsymbol{\lambda}^k$ obtained from the update scheme (3.53) converges to $\boldsymbol{\lambda}^*$, the Lagrange multipliers satisfying the KKT conditions, without needing to further increase the penalty parameter [Nocedal and Wright, 2000]. This largely alleviates the problem of ill-conditioned Hessians of the quadratic penalty method presented above without introducing any additional adverse effects.

Dealing with Bound Constraints: The Gradient Projection Method

Many constrained minimization problems, like the one that we will treat in Chapter 5 additionally require that some of the parameters to optimize lie in a certain range.

These bound constraints can be easily integrated into the ALM procedure described above. In this case, the optimization problem that we consider is of the form

$$\begin{aligned} \min_{\mathbf{y}} \quad & \mathcal{L}(\mathbf{y}) \\ \text{s.t} \quad & \mathbf{l} \leq \mathbf{y} \leq \mathbf{u} \,, \end{aligned} \tag{3.54}$$

where $\mathbf{l}$, resp. $\mathbf{u}$, is the vector of lower bounds, resp. upper bounds, whose components may be set to $-\infty$, resp. $+\infty$.

A simple, yet efficient, method to solve the problem (3.54) is the so-called gradient projection method. The procedure allows for rapid changes in the set of working variables, and is consequently very well suited for large-scale problems. Each iteration of the gradient projection method consists of two steps. First, the set of active constraints $\mathcal{A}(\mathbf{y}^k)$ is estimated by computing the minimum of a quadratic approximation q of the Lagrangian along the steepest descent direction from the current iterate $\mathbf{y}^k$. This direction is projected onto the bound constraint box beforehand so as to obtain a feasible point $\mathbf{y}^c$, the so-called *Cauchy point*. In a second stage, the face of the feasible box on which the Cauchy point lies is explored by fixing the active constraints at their values at the Cauchy point and approximately solving the quadratic subproblem

$$\begin{aligned} \min_{\mathbf{y}} \quad & q(\mathbf{y}) \\ \text{s.t} \quad & y_i = y_i^c \,, \;\; i \in \mathcal{A}(\mathbf{y}^k), \\ & l_i \leq y_i \leq u_i \,, \;\; i \notin \mathcal{A}(\mathbf{y}^k). \end{aligned} \tag{3.55}$$

Global convergence of the method is guaranteed as long as the function value corresponding to the new iterate is no worse than the one of the Cauchy point [Nocedal and Wright, 2000].

Discussion

Both SQP and ALM have their own advantages and drawbacks and appeared to be complementary in the applications that we tackled in this thesis. In practice, we found ALM to be particularly robust. Solving the resulting linear systems was usually possible with most common solvers, well known regularization methods could be employed in case of indefinite matrices and building the merit function was straight forward since using the augmented objective function was a natural and very efficient choice. Also, redundant constraints do not seem to be very problematic with this method, as Lagrange multipliers are not explicitly treated as degrees of freedom and ill-ranked constraint Jacobians are naturally "regularized" by the Jacobian of the objective function. On the other side, ALM handles constraints by looking at their norm only, which is an issue when these constraints

are largely nonconvex. In such cases, SQP proved to be much more effective than ALM, which typically did not converged.

3.3 Conclusion

In this chapter, we have seen how the deformed state of a soft object, be it a solid or a membrane, could be accurately computed using a finite element method relying on principles of nonlinear elasticity and an efficient discretization scheme. We then showed how the simulation pipeline could be reversed in order to deduce the optimal values of the physical system's unknowns from a set of given deformed poses. This allowed us to introduce a generic optimization framework to design physically valid custom objects that will serve as a base for the applications treated in the next chapters. Finally, we reviewed different methods for efficiently solving the constrained minimization problem that we previously introduced.

In what follows, we will tailor the optimization procedure to different instances of the general problem of designing deformable objects. We will start by playing with the object's undeformed shape in the context of rubber balloon design (Chapter 4). Then we will address the question of optimal force locations and base material distribution arising when creating actuated deformable characters (Chapter 5). Finally we will combine our optimization component with an interactive tool in order to account for aesthetic aspects when fabricating panel-based inflatable structures (Chapter 6).

CHAPTER 4

Designing Custom Rubber Balloons

Deformable objects come in a variety of forms and materials. This richness allows for a large diversity of applications, in particular in the entertainment industry. In this chapter, we focus on the design of custom shaped balloons, which presents various advantages. First, balloons are relatively simple to fabricate, which make them suitable to experiments. Second, the only external forces exerted on them are gravity and pressure forces, that can be controlled by a single parameter, the amount of injected air. This allows us to isolate and explore the influence of rest shape optimization without having to deal with extra unknowns related to actuation. Finally, this example demonstrate the capabilities of our system to accurately predict and control the shape of largely deformed objects.

4.1 Introduction

Inflatable balloons are fascinating objects that attract the attention of both children and adults —or, in the words of A.A. Milne's character Winnie the Pooh, *'nobody can be uncheered with a balloon'*. Especially rubber balloons enjoy a high popularity, be it for advertisement and decoration or simply as toys. A primary reason for their cross-cultural ubiquity lies in the ease of manufacturing: the balloon mold, similar in shape to the inflated balloon, is briefly dipped into liquid rubber, e.g., latex. The rubber is then cured, removed from the mold—and the balloon is ready to deploy. This process is simple, efficient, and inexpensive, making it ideal for commercial production. But although more complex shaped molds could be used to obtain a wider range of balloon shapes, manually designing a mold that yields a

complex inflated shape is a formidable task. For this reason, we mostly see rubber balloons of simple shapes such as ellipsoids, wavy tubes, or coarse approximations of hearts and bunnies. Foil balloons, on this other side, can be produced with complex shapes such as those seen in Macy's annual Thanks Giving parade. Their design present different —but just as much interesting— challenges than the ones arising when designing rubber balloons and we will address their case in Chapter 6. However, foil balloons do not stretch noticeably during inflation. Rubber balloons, by contract, largely deform, which makes the experience of inflating them so unique.

Our goal in this chapter is to develop a method for designing balloons that, once fabricated, can be inflated into as complex shapes as foil balloons but are as deformable and as easy to manufacture as conventional rubber balloons. There are essentially two options for controlling the inflated shape of a balloon: varying its material properties locally and modifying its rest shape. Arbitrarily varying the material properties in a single production cycle seems very difficult but, although technically challenging, one could imagine a multi-layer fabrication process. A simpler way of locally controlling the material stiffness is to vary the thickness of a homogeneous material. But although intriguing at first, we found through experiments that this technique alone cannot provide sufficient shape variation. Furthermore, varying the thickness will most likely require a closed mold, which significantly complicates the fabrication process compared to the dip-molding used for conventional rubber balloons. Modifying the rest shape, on the contrary, enables us to approximate a wide range of target shapes with homogeneous (constant thickness) material and is compatible with dip molding fabrication. This is therefore, we believe, the enabling technology for mass fabrication of custom-shaped rubber balloons.

4.2 Overview

This chapter presents a process for automatic design and easy fabrication of balloons that can be inflated into desired shapes that are given as inputs to our system. The pipeline of the system, illustrated in Figure 4.1 is divided into three steps. We start by capturing and fitting the material properties of the material that we intend to use for the actual balloons, silicone in practice. We then run an optimization component which computes the optimal rest shapes of the balloons according to the scheme introduced in Chapter 3. This component is based on accurate simulation of the balloon behaviour under inflation. Finally, we use the rest shapes to 3D print the molds which will be employed to fabricate the real balloons. While the uninflated balloons produced by our method resemble their target shapes to

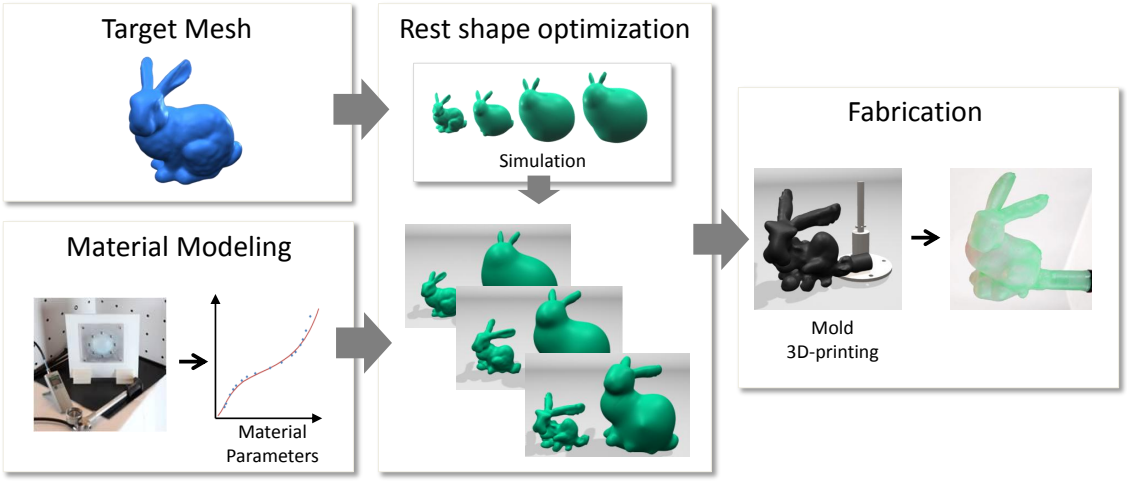


Figure 4.1: *Pipeline overview: Given a target shape, we use experimentally-acquired material properties to compute and fabricate an optimal balloon shape.*

some extent (see, e.g., Figure 4.6), they are not mere down-scaled versions of the target shapes and are very difficult to design manually.

Our main contributions in this chapter can be summarized as follows:

- We describe a complete process for designing and fabricating custom-shaped rubber balloons, including parameter acquisition, computational modeling, shape optimization and physical fabrication.
- We cast balloon shape optimization as a constrained minimization problem, combining strict enforcement of physical consistency with optimal shape approximation.
- We validate our method on a set of test cases and demonstrate its capabilities on a number of challenging balloon shapes.

The remainder of this chapter is organized as follows. We describe the mechanics of rubber balloons, present our computational model and motivate our choice of material model in Section 4.3. Section 4.4 describes the rest shape optimization, including the formulation of the problem as well as its numerical solution. We provide details on the fabrication process in Section 4.5 and present results in Section 4.6. This chapter concludes with a discussion of limitations and directions for future work.

4.3 Rubber Balloon Model

Our goal is to fabricate rubber balloons that, when inflated, assume a given target shape. To this end, we modify the rest shape of the balloons and use physical simulation to predict the resulting inflated shapes. This requires an adequate computational model, which is described in this section.

4.3.1 Mechanics

Elastic membrane

Rubber balloons can be inflated to several times their initial volume and still return to their original rest shape upon deflation. This observation suggests an elastic material model and, due to the large deformations that are observed during inflation, we turn to nonlinear continuum mechanics.

The geometry of balloons, which are essentially thin layers of rubber, motivates a surface-centered representation. Using the same notations as in Section 3.1.4, we describe the middle surface of the balloon in its undeformed and deformed configurations by the mappings $\bar{\mathbf{x}} : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^3$, respectively $\mathbf{x} : \Omega \to \mathbb{R}^3$ with $\Omega \subset \mathbb{R}^2$ denoting the surface's parametric domain with coordinates (u, v). For the sake of simplicity, we will assume in what follows that $\bar{\mathbf{x}}$ is an isometric parametrization.

During inflation, balloons undergo both stretching and bending deformation. But unlike typical thin shell materials the in-plane deformation largely dominates the bending contributions, that we will therefore neglect. Focusing only on the stretching of the membrane, we follow the derivations of Section 3.1.4 and introduce the tangent vectors $\mathbf{a}_u = \frac{\partial \mathbf{x}}{\partial u}$ and $\mathbf{a}_v = \frac{\partial \mathbf{x}}{\partial v}$ on the deformed surface. Assuming that the deformation is volume-preserving and exhibit no transverse shearing, we use the tangent vectors to define the 2×2 metric tensors $\tilde{\mathbf{C}}$, with components $\tilde{\mathbf{C}}_{uv} = \mathbf{a}_u^T \mathbf{a}_v$, from which we can construct the 3×3 right Cauchy Green tensor

$$\mathbf{C} = \begin{bmatrix} \tilde{\mathbf{C}} & 0 \\ 0 & (\det \tilde{\mathbf{C}})^{-1} \end{bmatrix} . \tag{4.1}$$

This canonical strain representation is amenable to standard elastic material models, which are often described in terms of the first three invariants of $\mathbf{C}$,

$$I_1 = \operatorname{tr}(\mathbf{C}) , \quad I_2 = \frac{1}{2} \left[I_1^2 - \operatorname{tr}(\mathbf{C}^T \mathbf{C}) \right] , \quad I_3 = \det \mathbf{C} . \tag{4.2}$$

Assuming incompressibility, we have $I_3 = 1$ and can write the balloon's strain energy density Ψ due to a given deformation $\mathbf{C}$ as

$$\Psi(\mathbf{C}) = \Psi(I_1, I_2) \, . \tag{4.3}$$

A discrete expression for the energy function (4.3) can be obtained as described in Section 3.1.4. We approximate the geometry of a balloon with n vertices and assume constant deformation throughout each triangle element. This allows us to define the membrane energy for each deformed triangle and eventually to compute the total elastic energy W^{mem} by summing up all elemental contributions.

Pressure

The forces due to an enclosed pressurized gas tend to increase the volume of its container. Since the pressure is constant throughout the container, the resulting normal force density is the same everywhere on its surface.

In the continuous setting, the pressure can be defined via the total forces $\mathbf{f}_A$ acting on a surface element dA with normal $\mathbf{n}$ as

$$p = \lim_{dA \to 0} \frac{d\mathbf{f}_A \cdot \mathbf{n}}{dA} \, . \tag{4.4}$$

This expression allows us to derive discrete nodal pressure forces as

$$\mathbf{f}_i^{\text{p}} = \sum_{j \in \mathcal{T}_i} w_{ij} \cdot p \cdot A_j \mathbf{n}_j \, , \tag{4.5}$$

where $\mathcal{T}_i$ is the set of triangles incident to vertex i, $\mathbf{n}_j$ and A_j are the area and normal of triangle j and w_{ij} are weights. We use constant weights of $w_{ij} = 1/3$, which is equivalent to computing the pressure forces from area-weighted normals.

As an aside, we note that the process of inflating a balloon is not a simple increase in pressure, but rather an increase in the number of gas molecules inside the balloon. The difference becomes clear when writing the ideal gas equation as

$$pV = NRT \, , \tag{4.6}$$

where N is the amount of gas contained in volume V, T is the temperature and R is the gas constant. Clearly, increasing the amount of gas (through pumping) increases the product of pressure and volume but, as we will see in Section 4.3.2, neither of them has to increase monotonically.

Equilibrium Configuration

As part of our shape optimization (Section 4.4) and parameter fitting (Section 4.3.2) methods, we have to compute the deformed shape that a balloon assumes under a given inflation pressure. Neglecting gravity, static equilibrium implies that elastic and pressure force densities cancel out in every point on the surface. In the discrete setting, this boils down to require nodal force balance, i.e.,

$$\mathbf{f}_k^{\text{mem}} + \mathbf{f}_k^{\text{p}} = 0 \qquad 0 \leq k \leq n \,, \tag{4.7}$$

where $\mathbf{f}_k^{\text{mem}} = -\nabla_{\mathbf{x}_k} W^{\text{mem}}$ are the elastic membrane forces and $\mathbf{f}_k^{\text{p}}$ denote the pressure forces as defined in (4.5).

Provided a set of position constraints (for the nozzle), the nonlinear equations (4.7) can be solved iteratively with the standard Newton-Raphson procedure described in Section 3.1.3. In order to improve robustness and convergence we employ line search and incremental loading on the pressure term. As for the expressions for the membrane forces and their Jacobian (tangent stiffness matrix), they are generated automatically using a computer algebra package.

Having established a computational model for simulating balloons, we can now address the modeling of their material behavior.

4.3.2 Material

Rubbers generally show a nonlinear stress-strain response. However, the extreme deformations observed during balloon inflation lead to a particularly complex behavior which, as shown below, most conventional material models cannot properly reflect. But whatever the material model, it is crucial to determine its parameters experimentally in order to obtain accurate agreement with physical reality. For this purpose, we use an experimental measurement setup that is close to our target application.

Measurements

Similar to the system described by Treloar [Treloar, 1944], we fix a rubber membrane to the supporting plane of a base apparatus using a clamp with a circular opening of 6 cm diameter (see Figure 4.2). Two holes on the back of the supporting plane allow us to connect a valve and an off-the-shelf pump for inflation as well as a digital manometer. The measurement process for a given sample membrane consists of at least 10 loading steps in each of which we inject an additional amount

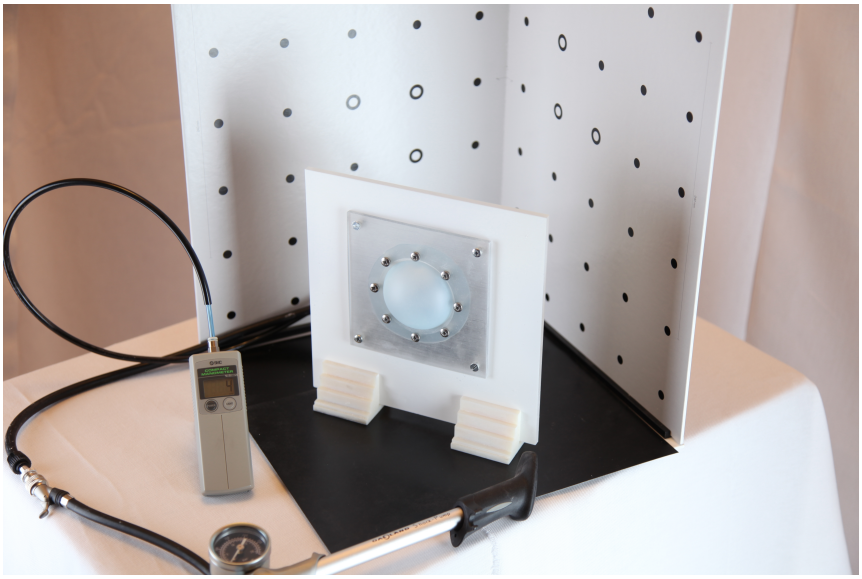


Figure 4.2: *Our measurement setup allows for capturing deformed shapes of silicone membranes subject to adjustable inflation pressures.*

of air, record the resulting pressure and capture the deformed geometry using a standard laser scanner. We then determine the spatially averaged extension ratio ρ_i^{avg} for step i from the area of the reconstructed geometry A_i as

$$\rho_i^{\text{avg}} = \sqrt{A_i / A_0}\,, \tag{4.8}$$

where A_0 is the initial area of the membrane.

We performed tests for silicone membranes with four different thicknesses (0.25, 0.5, 0.75 and 1.0mm), using three samples per thickness value to account for fabrication tolerance. Figure 4.3 shows the measured data (blue spades) for a thickness of 0.5mm averaged over the three samples. Our method is not restricted to a particular type of rubber, but for practical reasons (see Section 4.5), we exclusively used silicone for fabrication. However, in order to put these data in perspective, we also performed measurements on a latex membrane with 0.25 mm thickness (see Section 4.6).

Fitting

With this set of measurements, we determine the material coefficients of our computational model by minimizing the difference between simulated and captured shapes using the optimization algorithm described in Section 4.4. Since we do not have any information regarding local displacements, but only the global shape

of the deformed membrane, we choose a distance metric based on a smooth interpolation of the reconstructed geometry using radial basis functions (RBF). We define the distance energy between each captured sample $\mathbf{t}^i$ and the corresponding simulated membrane $\mathbf{x}^i$ as

$$E_d^i(\mathbf{t}^i, \mathbf{x}^i) = \frac{1}{2} \sum_{k=1}^{n} d_{RBF}^2(\mathbf{t}^i, \mathbf{x}_k^i) \ , \tag{4.9}$$

where $\mathbf{x}_k^i$, $1 \leq k \leq n$, denote the individual positions of the vertices of the simulated mesh and d_{RBF} is the function whose zero-level set defines the smooth RBF-surface, constructed using triharmonic kernel functions (see Carr et al. [Carr et al., 2001]). Minimizing the distance for all the pressure-geometry pairs simultaneously allows us to obtain material coefficients that provide the best average match for the entire deformation range of the experiment.

Evaluation

As can be seen from Figure 4.3, the measurements reveal an unusual deformation behavior. The average extension ratio first increases almost linear with respect to the pressure. At a certain point, however, there is a clear inflection in the curve indicating a second deformation regime of the material. In a third regime, the material stiffness increases again.

Inflated balloons will in general exhibit inhomogeneous deformations, most likely covering all three regimes. If good approximations are to be obtained, then this particular behavior must be reproduced by the material model. This is a challenging setting for a material model and we will investigate some candidates in the following. In order to facilitate comparison, we provide stress-strain curves for all materials considered in Figure 4.3. The energy functions of these models are provided in Appendix A.

The membrane formulation described in Section 3.1.4 guarantees volume preservation through geometric assumptions (thickness stretch compensates change in area) and thus avoids the numerical problems typically associated with incompressible elasticity. In this setting, the simplest nonlinear material model is the Neo-Hookean solid, which describes the strain energy as a linear function of the first invariant I_1. However, as can be seen in Figure 4.3, this model fails dramatically to approximate the deformation behavior of real rubber. The reason for this is that the Neo-Hookean material has a pressure peak at a rather small extension ratio beyond which the pressure drops with increasing deformation. Visually speaking, the peak value has to be high enough to match the pressure values for larger extension ratios from the experimental data. But as a result, the overall deformation behavior deviates wildly from the experiment.

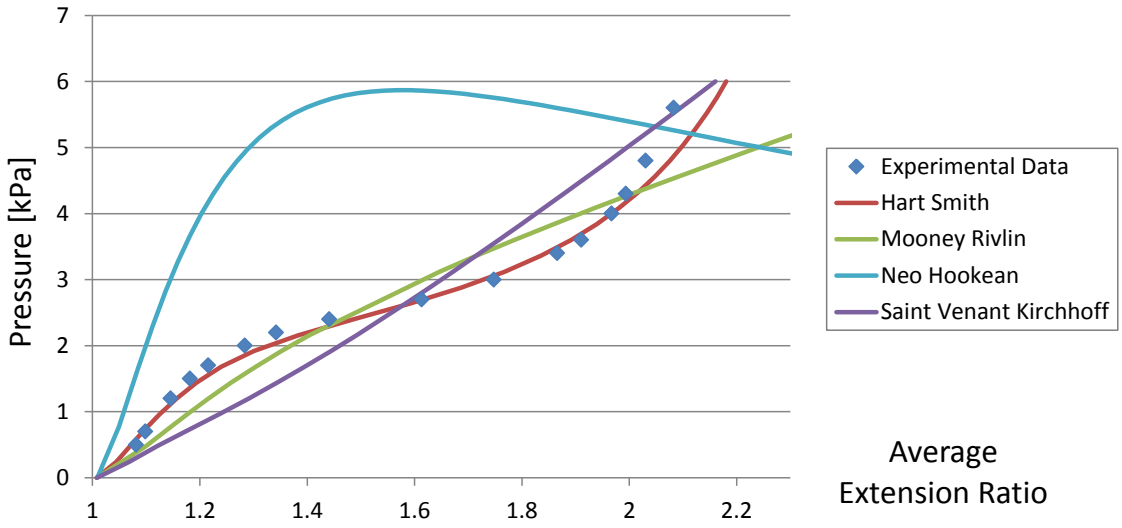


Figure 4.3: *Experimental data for the pressure-extension behavior of a silicone membrane and approximations with different material models.*

The second model that we considered is the St.Venant-Kirchhoff solid [Bonet and Wood, 1997] which, despite shortcomings [Irving et al., 2004], is widely used in graphics. But while the approximation is significantly better than for the Neo-Hookean material, it cannot reproduce the three characteristic deformation regimes of the experimental data. A similar behavior is observed for the Mooney-Rivlin material, which is a generalization of the Neo-Hookean solid that also considers the second invariant I_2. One option to improve the approximation quality is to include higher powers of the invariants or resort to Ogden-type models, which describe the energy in terms of powers of the principal stretches. However, each additional term entails an additional material constant, which increases the model complexity and thus complicates the process of parameter fitting.

The experimental observations by Treloar [1944] suggest that, for large strains, rubber material exhibits an energy growth which is an exponential rather than a polynomial function of the invariants. The empirical model of Hart-Smith [1966] takes this behavior directly into account, describing an exponential strain energy function through its derivatives as

$$\frac{\partial W}{\partial I_1} = G \cdot e^{k_1 (I_1 - 3)^2}, \quad \frac{\partial W}{\partial I_2} = G \cdot \frac{k_2}{I_2}, \tag{4.10}$$

where k_1, k_2 and G are material constants. Visually, the deformation behavior for small to moderate stretches is determined by G and k_2, while k_1 decides how quickly the exponential growth manifests. Despite the small set of material coefficients, the Hart-Smith material is capable of accurately reproducing the three deformation regimes observed in the experimental data.

The elastic forces for the Hart-Smith model can be computed directly from (4.10) as

$$\mathbf{f}^{\text{mem}} = -\frac{\partial W}{\partial \mathbf{x}} = -\frac{\partial W}{\partial I_1}\frac{\partial I_1}{\partial \mathbf{x}} - \frac{\partial W}{\partial I_2}\frac{\partial I_2}{\partial \mathbf{x}} \,. \tag{4.11}$$

The derivatives of the two invariants can be calculated by hand when using (4.1) in (4.2) and deriving component-wise. But since the involved expressions are rather lengthy and thus prone to error, we opt for a computer algebra software to compute first and second derivatives of W with respect to current as well as undeformed positions.

This completes our computational balloon model and we can now turn towards shape optimization.

4.4 Shape Optimization

This section explains our strategy for computing optimal rest shapes for rubber balloons, which we cast as a nonlinear constrained optimization problem. We start by formalizing the problem, then address some technical aspects.

4.4.1 Problem Setting

Our goal is to find a rest shape that, upon inflation, approximates the target shape as closely as possible. We measure *closeness* in terms of a distance energy $E_{dist}(\mathbf{t}, \mathbf{x})$ depending on the geometries of the inflated balloon, $\mathbf{x}$, and the target shape, $\mathbf{t}$. The distance measure should capture differences in first (stretching) and second (bending) fundamentals forms between the inflated balloon and the target shape. We use a variant of the discrete shell energy by Grinspun et al. [Grinspun et al., 2003] for this purpose and, using $\mathbf{x}$ and $\mathbf{t}$ as current respectively rest state, define

$$E_d = \sum_e \left[k_l (1 - \frac{l_e}{l_e^0})^2 + k_b (\theta_e - \theta_e^0)^2 \right] l_e^0 \,, \tag{4.12}$$

where l_e and θ_e (l_e^0 and θ_e^0) denote the deformed (undeformed) length and dihedral angle of edge e as illustrated in Figure 4.4, and k_l, k_b are stretching and bending coefficients. In comparison to simpler measures based on pairwise vertex distance, this metric has the advantage that the error caused by local deviations in shape remains local and does not propagate over the model.

With this distance metric established, we seek to find the physical solution, i.e., a configuration in force equilibrium, which is closest to the target shape, i.e.,

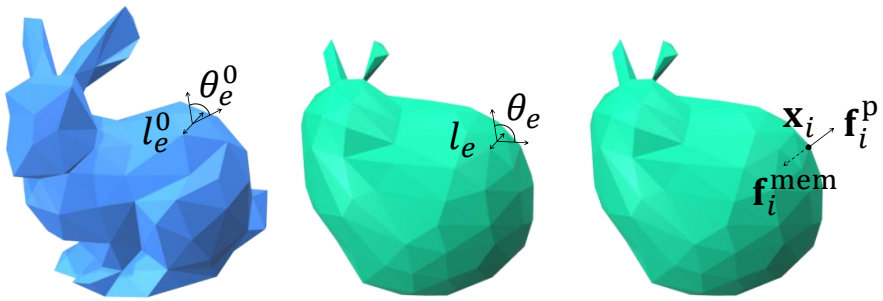


Figure 4.4: Left and Middle*: Edge length and dihedral angle on the target mesh and simulation mesh respectively.* Right*: Forces acting on a vertex of the simulation mesh.*

minimizes (4.12). As explained in Chapter 3, these requirements can be formulated as a constrained optimization problem with an objective function

$$\mathcal{L}(\mathbf{x}, \bar{\mathbf{x}}, \boldsymbol{\lambda}) = E_d(\mathbf{t}, \mathbf{x}) - \boldsymbol{\lambda}^t \mathbf{f}(\mathbf{x}, \bar{\mathbf{x}}) \,, \tag{4.13}$$

where $\mathbf{f}$ is the vector of constraints, each of which measures the deviation from force equilibrium for a given degree of freedom, i.e.,

$$\mathbf{f}_j(\mathbf{x}, \bar{\mathbf{x}}) = \mathbf{f}_j^{\mathrm{mem}} + \mathbf{f}_j^{\mathrm{p}} \,, \quad 1 \leq j \leq 3n \,. \tag{4.14}$$

Note that the system is in equilibrium only if $\mathbf{f}(\mathbf{x}, \bar{\mathbf{x}}) = \mathbf{0}$.

4.4.2 Numerical Solution

Solving the constrained minimization problem described by (4.13) requires some care. Since the force balance condition (4.14) is a fundamental requirement to obtain a physically valid result, we turn toward a method which guarantee strict constraint satisfaction and use the augmented Lagrangian method exposed in Section 3.2.2.

Regularization

The problem described by (4.13) is nonlinear as well as nonconvex and will therefore exhibit local minima. In particular, any region on the rest shape enclosed by a planar curve can be mirrored without altering the inflated shape. In order to obtain a well-posed problem, we favor convex shapes by asking for the solution with maximum volume.

Moreover, a numerical problem arises in regions with coplanar elements, for which the hessian of the membrane energy exhibits a null-space in the normal

direction. We solve this issue with an additional energy term based on the bending component of (4.12), thus penalizing curvature deviations from the initial rest shape.

Inflation Pressure

The optimization process starts with a downscaled version of the target shape. For practical reasons, we do not directly prescribe an extension ratio between uninflated and inflated balloons but determine an inflation pressure which achieves the desired increase in volume. To this end, we first increase the inflation pressure in a sequence of static equilibrium solves, until the volume of the inflated shape matches the volume of the target shape. We then use this pressure to compute the optimal balloon shape.

Rest Shape Intersections

If not explicitly prevented, the optimization process is likely to introduce intersections in the rest shapes. We use a collision handling strategy based on penalty forces to avoid such unphysical settings. In each iteration of the optimization, we first detect all pairs of vertices that are closer than a given threshold ε_c. In order to improve runtime performance, we use a kD-tree and cull vertex-pairs whose (approximate) geodesic distance is smaller than a threshold value ε_{gd}. For each vertex pair $(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j)$ returned by the detection step, we introduce a penalty potential as

$$W_p(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j) = \left(|\bar{\mathbf{x}}_i - \bar{\mathbf{x}}_j| - \varepsilon_c\right)^3 , \tag{4.15}$$

which is active for $|\bar{\mathbf{x}}_i - \bar{\mathbf{x}}_j| - \varepsilon_c < 0$ and set to zero otherwise. Note that the cubic power of the potential ensures continuity of its second derivatives. The potentials for all points are then added to (4.13) such that, upon convergence, all vertex-vertex constraints are satisfied. Although this simple approach cannot resolve all intricate collisions in general, we found it to be sufficient for all our examples.

Nozzle Attachment

We are primarily interested in approximating the target shape when designing/optimizing the rest shape for a balloon, but we also have to include the nozzle attachment in this process. The nozzle is particular in that it does not noticeably deform during inflation, i.e., it has the same size in the undeformed and inflated configuration. However, this same size covers different regions in the two configurations such that simply attaching it to the two geometries will interfere

with the triangle correspondence. We solve this problems by attaching the nozzle in the inflated configuration (in fact, we ask the user to do so) resulting in a set $\mathcal{N}_i$ of vertices (on the target surface) that are constrained to remain at positions $\mathbf{p}_i$ in both configurations. While the target shape trivially satisfies these constraints, further action is required to obtain a high-quality rest shape mesh which accounts for the nozzle constraints and has the same topology as the target mesh. We use a nonlinear variant of Laplacian mesh smoothing (see, e.g., [Nealen et al., 2006]) for this purpose in which we impose the nozzle vertices as hard position constraints and require all other vertices to remain on the original surface. For the latter, we construct an implicit representation of the surface using radial basis functions, similar to the one that we used for the material fitting (see Section 4.3.2) and compute the smoothed mesh positions as the minimizer of

$$E_s(\bar{\mathbf{x}}) = E_c(\bar{\mathbf{x}}) + E_{rbf}(\bar{\mathbf{x}}) \;\text{ s.t. }\; \bar{\mathbf{x}}_i = \mathbf{p}_i,\; i \in \mathcal{N}\,. \tag{4.16}$$

The first term E_c measures the distance of a given vertex to the centroid of its one-ring neighborhood,

$$E_c(\bar{\mathbf{x}}) = \bar{\mathbf{x}}_i - \frac{1}{N_i}\sum_{j\in\mathcal{T}_i} \bar{\mathbf{x}}_j \tag{4.17}$$

with N_i denoting the valence of vertex i and $\mathcal{T}_i$ its set of one-ring neighbours. The second term penalizes deviations from the original (RBF-) surface and has the same form as in (4.9).

4.5 Fabrication

This section briefly outlines our process for fabricating custom-shaped balloons.

4.5.1 Mold Design

The optimization method returns a triangle mesh for the balloon's rest shape. We complete the balloon mold by attaching a cylindrical tube to the previously defined nozzle vertices (see Section 4.4) as well as some additional geometry for fixing the mold to a support as illustrated in Figure 4.5. The mold can then be fabricated using, e.g., a 3D printer. Even though the silicone balloon can be stretched during unmolding, this process can be quite cumbersome for complex shapes such as the bunny. Fortunately, this problem can be avoided when using a printer supporting soft materials such as our Objet Connex.

Figure 4.5: Left: *Balloon Mold.* Right: *Silicone balloon.*

4.5.2 Balloon Fabrication

Industrial balloon fabrication usually relies on latex due to its durability and low material costs. For manual fabrication, however, we found silicone more convenient to handle: unlike latex, it is not sensitive to temperature, cures without volume loss and can be evacuated to remove air inclusions. The silicone has to be mixed with an activator and be processed rapidly. Since remains cannot be stored, a dip-molding process is not attractive and we simply brush the rubber onto the mold. We then put the coated mold in a vacuum chamber to remove air inclusions. We found that this also leads to smoother coatings.

The coating thickness obtained by this process is in the range of 0.25mm-0.3mm. In order to increase the durability of the balloons we add another another layer using the same process. Finally, the balloon needs to cure for one day before removing it from the mold and an additional 5 to 7 days to obtain its final material properties. Figure 4.5 shows one of our silicone balloons after removal from the mold.

4.6 Results

In order to explore the capabilities of our method we experimented with a variety of different target shapes. For each example, we computed an optimized rest shape, printed corresponding molds and fabricated the balloons using silicone. For comparison, we also fabricated balloons corresponding to the downscaled target shapes. Figure 4.6 shows all results in a compact overview.

Figure 4.6: *Results obtained with our method. The columns (from left to right) show target shapes, optimized balloons, simulated inflated balloons, fabricated optimized balloons, simulated inflated balloons corresponding to downscaled targets and corresponding fabricated balloons.*

The last column clearly indicates that the downscaled targets do not lead to acceptable approximations. Indeed, the original shape is indiscernible in most cases. By contrast, the simulated shapes of the optimized balloons (column 3) as well as the fabricated counterparts (column 4) are in good agreement with the targets for the majority of the examples and most of the characteristic features are clearly discernible.

Comparing the target shapes against the simulated inflated balloons, several observations can be made. First, it is impossible to accurately reproduce flat regions as the pressure forces tend to push them outwards. This effect is clearly visible for the faces of the cube (row 2) but can also be identified on the legs of the elephant (row 5) and the arms of the armadillo (row 6). Furthermore, sharp transitions such as the borders around the bumps on the sphere (row 1) or the edges of the cube (row 2) remain visible but are smoothed out. However, isolated sharp features such as the corners of the cube can be reproduced with fair accuracy since they do not have to stretch during inflation and can thus be built into the undeformed shape. Finally, high frequency detail such as the fur of the bunny is lost. This is partly due to the limited accuracy of the fabrication process, but also owed to the fact balloons cannot have local concave features since these would be *popped out* by the pressure forces. This observation also explains why the snout of the armadillo and the ears of the bunny (row 4) are rather roundish.

As a side note, it can be seen from the cube and the sphere example that our method preserves the overall symmetry of optimized shapes for symmetric targets, even for non-symmetric (irregular) meshes.

The inflated real balloons (columns 4 and 6) are in good agreement with their simulated counterparts, only the armadillo and the hand (row 7) examples show significant deviations. This can be attributed to inaccuracies in the fabrication process combined with the complexity of the shapes: the highly viscous silicone tends to accumulate in the troughs of the mold, leading to significant thickness variations. This effect could be reduced by using thinned rubber and a higher number of layers, albeit at the expense of increased fabrication times.

Larger Inflations

The average extension ratio for the inflated balloons stays below 2.5 in all our examples, whereas values around 5 are often observed for latex balloons. The reason for this difference is that latex is far more durable than silicone, which in our experiments allowed only a stretching of 2.5 before rupture. But while this limits the increase in volume for silicone balloons, this limitation is not inherent to our approach: we simulate the teddy example with a latex material whose parameters we determined in the same way as for silicone. Using latex, we can attain extension

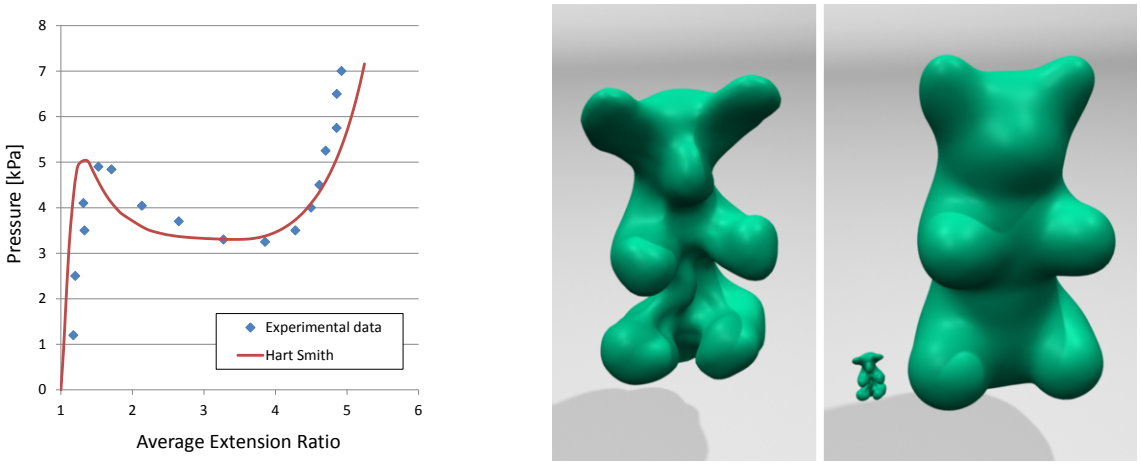


Figure 4.7: Left*: measured data for a latex membrane and the Hart-Smith model fit.* Right*: optimized and inflated shape for a teddy bear balloon with an average stretch of ≈5.*

ratios larger than 5 with a pressure of 7kPa (Figure 4.7, left). This in turn allows us to simulate balloons which dramatically increases in volume during inflation, while still providing good approximation to the target shape (Figure 4.7, right).

Performance

We provide computation times for all examples shown in this chapter in Table 4.1. It can be seen that for most balloons, our method takes less than 15 minutes to compute optimized shapes. With slightly less than 90 minutes, the bunny example takes by far the longest time to finish, which is mostly due to the large number of intersections that slow down convergence.

Model	#vertices	t_{stat} [ms]	t_{nwtn}[ms]	t_{tot} [s]	#its
Armadillo	6027	24494	5897	569	81
Bumpsphere	3265	20359	2993	201	55
Bunny	4975	61138	7212	5000	879
Cube	5153	39094	6543	243	20
Elephant	3818	26317	3031	3060	131
Gingerbread	2238	5329	1776	66	29
Hand	2253	24403	2696	3229	1137
Mouse	2633	22277	3200	627	180
Squirrel	5995	151447	7744	105	103
Teddy	3527	85936	3492	484	106

Table 4.1: *Computation times breakdown: static equilibrium (t_{stat}), a single Newton step (t_{nwtn}), total computation time (t_{tot}) and number of ALM iterations (#its).*

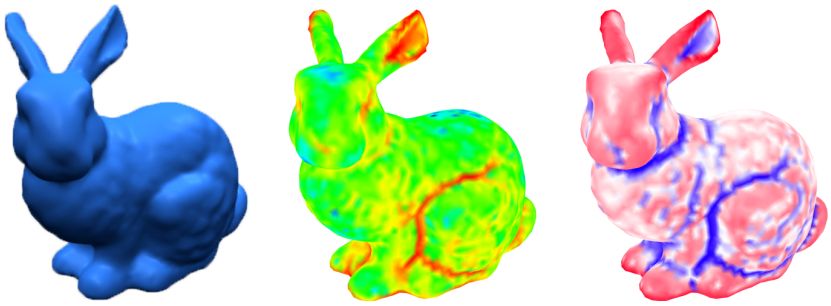


Figure 4.8: *Approximation error for the bunny with green/red indicating lowest/highest error* (middle)*. Red/blue color indicates positive/negative mean curvature* (right)*.*

Accuracy & Validation

In order to quantify the accuracy of our optimization method, we measured the difference between input target shapes and simulated optimized shapes. Figure 4.8 shows an error plot for the bunny example using the distance metric of (4.12). For easier interpretation, we measured for each vertex of the inflated shape the distance to the closest point on the target surface, leading to an average/maximum error of 0.23/0.89cm (the inflated balloon has a maximum inter-vertex distance of 14.6cm). As expected, the regions of highest error coincide with those of minimal mean curvature.

To estimate the impact of the inaccuracies in the fabrication process, we applied random thickness perturbations of different magnitudes to the bunny model and investigated the impact on the inflated shape using again the vertex-based distance measure. A variation of 10% led to an average deviation of 0.22cm, whereas 20% of variation already results in 0.52cm average error. This agrees with our observations: small thickness variations have only little effect while larger variations such as as those seen in the armadillo and the hand example can lead to significant shape deviations.

We furthermore verified the importance of the material model experimentally. In a simple test, we compared the inflated shape of a real bunny balloon to the results produced by simulations using the Mooney-Rivlin and the Hart-Smith material. The bunny result in Figure 4.6 shows that the Hart-Smith material results in good approximation with clearly visible ears, whereas these features were not discernible for the Mooney-Rivlin material, which led to an almost spherical shape.

Finally, we analyzed the stability of the numerical solutions computed by our method. Using the bear example, we let our algorithm run on a perturbed version of the optimized rest shape and compared the result to the original solution. For an average vertex displacement of 0.25cm, the difference was below 10^{-4}cm. This

indicates that minima are well separated and we found this to be true for all examples considered.

4.7 Discussion and Outlook

In this chapter, we presented a method for computational design of rubber balloons with desired inflated shapes. Our approach drastically increases the space of possible balloon shapes as demonstrated with a set of challenging examples.

Although our method is able to produce a large variety of shapes, certain features are difficult or even impossible to reproduce, in particular flat parts, locally-bounded concave regions, and sharp edges. These limitations do, however, apply to inflatable structures in general and are not specific to our method. By adding internal connections one could overcome some of these limitations and we will explore this direction in Chapter 6 when studying foil balloons.

There are evidently shapes that can be well-approximated by our method while others are difficult to reproduce with satisfying quality. However, many of these difficult shapes can be changed into models that lend themselves well to balloon fabrication—see, e.g., the elephant. Currently, we rely on artists to design geometrically similar but feasible target shapes. An interesting direction for future work would be to develop a method for automating this shape abstraction process.

Shape optimization is very powerful in the context of balloon design. However, this involves only a limited subset of all the possible design parameters that affect the final shape of a deformable body. External forces and materials, in particular, also have a large influence on the deformation. In the next chapter, we will turn toward a very different class of objects and investigate the question of optimal placement of pin- and string-actuators, as well as best material distribution, when designing custom animated characters.

CHAPTER 5

Creating Actuated Deformable Characters

In this chapter, we focus on actuated deformable characters whose design presents particularly interesting challenges. First, these characters are deformed using actuators that apply forces at localized positions on the surface. This naturally raises the question of where and how much force needs to be applied in order to yield a desired deformation. Second, actuated characters are typically made of several materials, which allows us to tackle the problem of finding the optimal material distribution when two base materials are available.

5.1 Introduction

Character design is a vital part of animated movie production, game development and other applications of computer graphics. Many virtual characters are rigidly articulated, others are very deformable, and most of them show properties between these two extremes ranging from humanoid virtual actors with bulging muscles, to invertebrate figures like jelly monsters and stylized background characters such as plants, buildings and other man-made objects. Digital characters are typically created solely for the virtual worlds they live in. However, many other applications such as theme park attractions, exhibitions, artistic installations or next-generation games require real, physical embodiments of these figures. While there is an extensive set of tools for digital character design and animation, translating animated

characters to the real world is an extremely difficult task. This problem is made even more evident by the quickly growing availability of rapid manufacturing devices and services that might soon lead to a switch from mass fabrication to personalized design of characters such as action figures.

Realizing this technological trend, recent work by Bächer et al. [2012] and Calì et al. [2012] proposed solutions for transforming articulated digital characters into 3D-printed figures that can be posed in various ways. While this is a significant advancement in fabrication-oriented character design, these methods are restricted to rigidly articulated characters and, more importantly, do not address the problem of how to animate the resulting figures.

Motivated by these observations, we propose a method for fabrication-oriented design of deformable characters that can be actuated using pins, strings, or posed by hand. Given a digital representation of an animated (or animatable) character as input, we seek to find a system of external actuators as well as an internal material distribution that allow us to fabricate a physical prototype whose range of deformation and movements closely approximate the input. Our solution to this problem is a dedicated algorithm that combines finite-element analysis, sparse regularization, and constrained optimization. We demonstrate our method on a set of two- and three-dimensional example characters. We present results in simulation as well as physically-fabricated prototypes with different types of actuators and materials.

5.2 Overview

In this chapter, we present a method for fabrication-oriented design of actuated deformable characters that allows a user to automatically create physical replicas of digitally designed characters using rapid manufacturing technologies. The input of our system is a mesh describing a deformable character in its neutral state as well as a set of target shapes that represent desired deformations. We then optimize for the actuation parameters (number, placement, and forces) and an internal material distribution that allow us to fabricate a physical character whose range of deformation closely approximates the target shapes. As summarized in Figure 5.1, our pipeline for deformable character design consists of three main stages that we briefly introduce below.

Initial Actuation The first stage determines an estimate of how many actuators should be used and in which regions they should be placed. We consider two variants: by default, we let the user select actuation points on the model. This is convenient when the user wants to have a particular number of actuators and/or

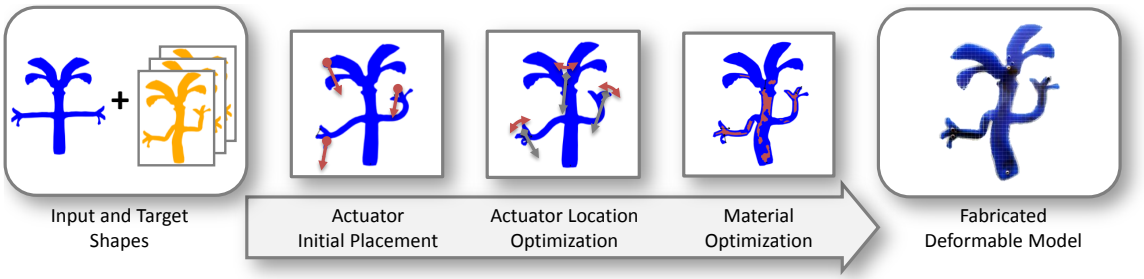


Figure 5.1: *An overview of our optimization scheme: the user provides an input character and target shapes, the number of actuators and their initial locations are determined, the positions of the actuators are optimized, and an internal material distribution is computed. Finally, the physical prototype is fabricated.*

knows roughly where they should be placed. In other cases, in particular for characters without apparent articulation structure, it can be difficult for the user to make a suggestion. In this case, we automatically suggest a number of actuators and their locations using sparse regularization.

Actuator Locations Given the number of actuators and an initial estimate for their locations, we next adjust their placement such as to minimize the overall distance of the model to the individual target poses. The problem is solved using constrained optimization.

Material Optimization With the actuator positions fixed on the model, the third stage computes an internal material distribution that further optimizes the matching for the individual target poses. We assume that there are two materials available and initially allow each element of the model to assume an arbitrary convex combination of the base materials. As the optimization progresses, we drive the elements' materials toward one of the base materials, which eventually results in a discrete material layout that is ready for fabrication.

Finally, we use the optimized actuator locations and material distribution to fabricate a physical prototype of the deformable character using rapid prototyping technology.

Each of the stages of the pipeline raises specific challenges that we address using novel techniques. In particular, and to summarize the contributions of this chapter,

- We automatically infer the number of external actuators needed to replicate a target animation using sparse regularization. Starting with a dense set of actuators on the boundary of the model, we discard unnecessary pins or strings by penalizing the L_1-norm of the forces that they exert on the object. The result of the proposed optimization procedure also provides an approximation of the actuators optimal locations.

- We refine actuators positions by decoupling actuators' attachment point locations and object's discrete representation. The technique we present improves the matching quality between the actuated simulated poses and the target deformed shapes by allowing the actuators to slide on the surface of the model, which is defined using a distance field.
- We introduce a goal-based method for automatically computing rigid structures to be embedded in multi-material actuated deformable objects. We address the problem from a material distribution perspective but instead of solving the discrete problem directly, we opt for a relax formulation which allows us to use continuous optimization techniques.

The outline of the remaining part of this chapter is the following. After introducing the model that we use to simulate our actuated deformable characters in Section 5.3, we expose the approach that we take to optimize the actuation system and the material distribution of the object in Section 5.4. Results are presented and discussed in Section 5.6. Finally, Section 5.7 concludes this chapter and provides directions for potential future work.

5.3 Simulation

Our system aims at automating the design of actuated deformable characters whose deformed poses approximate as closely as possible a set of provided target shapes. We assume that the type of actuators is known, and that the object is made of two base materials, a soft one and a stiff one. The optimization of the actuation system and the material distribution of the figures requires proper simulation of the object deformations. In this section, we describe the elastic model that we use for this purpose as well as the types of actuation forces that we consider.

5.3.1 Elastic Model

The computation of the deformed shape of a body subject to external forces was addressed in Chapter 3. In what follows, we quickly recap the basic concepts and the main steps of the procedure.

We assume that a deformable character in its neutral pose is given as input to the system. We distinguish between objects with an arbitrary rest shapes, represented by tetrahedron meshes, and objects with a constant thickness that we model using 2D triangle meshes. We let n_v denote the number of vertices in the mesh and use $\bar{\mathbf{x}} \in \mathbb{R}^{\text{dim}\cdot n_v}$ to refer to the vector of undeformed positions, where dim denotes the

number of dimensions. Likewise, we let $\mathbf{x} \in \mathbb{R}^{\dim \cdot n_v}$ denote the position vectors in the deformed configuration.

We start by constructing an elastic model of the input shape using finite elements. Since the characters that we consider exhibit large deformations, we use a nonlinear deformation measure but retain linear elements for the sake of efficiency. For simplicity, we first focus on the three-dimensional case. Let $\bar{\mathbf{x}}^e \in \mathbb{R}^{12}$, resp. $\mathbf{x}^e \in \mathbb{R}^{12}$, denote the vectors of concatenated undeformed, resp. deformed, nodal positions $\bar{\mathbf{x}}_i^e$, $0 \leq i \leq 3$, resp. $\mathbf{x}_i^e$, $0 \leq i \leq 3$ of a single tetrahedral element e. We first compute the deformation gradient $\mathbf{F}^e(\bar{\mathbf{x}}^e, \mathbf{x}^e) = \mathbf{d}\mathbf{D}^{-1}$ where $\mathbf{d}$ is the 3×3 matrix whose columns hold the edge vectors $\mathbf{d}_i = \mathbf{x}_i^e - \mathbf{x}_0^e$. Analogously, the edge vector matrix $\mathbf{D}$ is defined through $\mathbf{D}_i = \bar{\mathbf{x}}_i^e - \bar{\mathbf{x}}_0^e$. We define the elastic energy density of the element using a Neo-Hookean material model,

$$\Psi^e(\mathbf{F}^e) = \mu(\mathrm{tr}(\mathbf{C}^e) - 3) + \frac{\kappa}{2}(\det(\mathbf{F}^e) - 1)^2 \,, \tag{5.1}$$

where μ and κ are material parameters and $\mathbf{C} = (\mathbf{F}^e)^t\mathbf{F}^e$ is the right Cauchy-Green tensor. The elastic energy of the deformed element is then obtained by integrating (5.1) over its domain. Since we use linear finite elements, the deformation gradient is constant across the element such that we have $W^e = \Psi^e V^e$, where V^e denotes the volume of the tetrahedron. The global deformation energy $W^{\mathrm{int}}(\bar{\mathbf{x}}, \mathbf{x})$ between the rest state $\bar{\mathbf{x}}$ and an arbitrary deformed state $\mathbf{x}$ is obtained by summing up elemental contributions as $W^{\mathrm{int}}(\bar{\mathbf{x}}, \mathbf{x}) = \sum_e W^e(\bar{\mathbf{x}}_e, \mathbf{x}_e)$. The elastic energy gives rise to internal forces $\mathbf{f}^{\mathrm{int}} \in \mathbb{R}^{\dim \cdot n_v}$ as

$$\mathbf{f}^{\mathrm{int}} = -\frac{\partial W^{\mathrm{int}}(\bar{\mathbf{x}}, \mathbf{x})}{\partial \mathbf{x}} \,. \tag{5.2}$$

Finally, letting $\mathbf{f}^{\mathrm{ext}}$ denote the vector of external nodal forces acting on the discretized body, the deformed pose of the object is obtained as a solution of the partial differential equations

$$\mathbf{f}^{\mathrm{int}} = -\mathbf{f}^{\mathrm{ext}} \,, \tag{5.3}$$

which we solve using a Newton-Raphson procedure as described in Section 3.1.3.

For the two-dimensional case, we use linear triangle elements, and the derivation of the elastic energy is largely similar. As detailed in Section 3.1.4 we expand the two-dimensional Cauchy-Green tensor to three dimensions by inferring the thickness stretch from the assumption of volume-preserving deformations. This allows us to also use (5.1) for the two-dimensional case.

We can now have a closer look at the external forces acting on the physical system. Besides gravity forces, which are trivially incorporated to $\mathbf{f}^{\mathrm{ext}}$ after lumping the mass of the object to the vertices of the mesh, these external forces include the forces exerted by the actuators.

5.3.2 Actuation forces

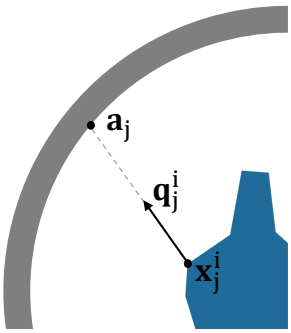
Our design framework supports three kinds of actuators: pin-type actuators, string-type actuators and clamp-type actuators. As we detail below, the handling of the external actuation forces is different for each of these variants.

Pin-based actuators

Pin-based actuators exert traction forces on the surface of the object which act on very small areas, that we equate with points. For simplicity, we will assume here that these points coincide with vertices of the simulation mesh. The forces due to pins can be arbitrary, and are different from one pose to the other, but they act at the same location on the boundary for all the poses of the object. Letting n_a be the number of actuators, we will denote by $\mathcal{Q} = (\mathbf{l}, \mathbf{q}^1, \ldots, \mathbf{q}^{n_p})$ the actuator system where $\mathbf{l} \in \mathbb{R}^{\dim \cdot n_a}$ holds the locations of the actuators in the rest state and each vector $\mathbf{q}^i \in \mathbb{R}^{\dim \cdot n_a}$ holds the actuation forces for a given pose $\mathbf{x}^i$. We account for the forces due to pin-based actuators in a given state $\mathbf{x}^i$ by simply adding the coordinates of the vector $\mathbf{q}^i$ to the corresponding entries of the vector of external forces $\mathbf{f}^{\text{ext}}$.

String-based actuators

A string actuator is defined through an attachment point $\mathbf{a}_j$ located on an external support structure as well as another attachment point on the surface of the character (see figure on the right). Like a pin actuator, it acts on a fixed isolated point on the boundary of the object, the string attachment point, that remains the same throughout the deformation. However, there are two major differences. First, the direction of the force due to a string matches the direction of the string at all time. Second, the string can only pull and not push. These requirements can be enforced by modeling the actuation force corresponding to the string as



$$\mathbf{q}_j^i = k_j^i \frac{(\mathbf{a}_j - \mathbf{x}_j^i)}{||\mathbf{a}_j - \mathbf{x}_j^i||}\,, \tag{5.4}$$

where $k_j^i > 0$ corresponds to the tension of the string and $\mathbf{x}_j^i$ is the actuator's location on the deformed surface.Using the formulation described above, the deformed pose of the object can be found by treating the tensions k_j^i as additional degrees of freedom.

Clamp-based actuators

Clamp-based actuators prescribe the positions for sets of vertices, thus emulating the process of posing characters by hand. In our discrete settings, these hard constraints are easily enforced by modifying the force Jacobian $\mathbf{J} = \frac{\partial \mathbf{f}(\mathbf{x}^i)}{\partial \mathbf{x}}$ and the vector of total forces $\mathbf{f} = \mathbf{f}^{\text{int}} + \mathbf{f}^{\text{ext}}$ before solving the linear system corresponding to the Newton step (3.26) such that for all constrained degrees of freedom $\mathbf{x}_c^i$, we have

$$\begin{aligned} &\mathbf{f}_c = 0 \\ &\mathbf{J}_{cc} = 1, \\ &\mathbf{J}_{cj} = \mathbf{J}_{jc} = 0\ ,\ \ j \neq c \end{aligned} \tag{5.5}$$

Having introduced the different physical quantities needed to compute the deformed state of an actuated figure, we address the optimization of the system's design variables in the next section.

5.4 Design Optimization

5.4.1 Basic Formulation

Our goal is to compute the internal material structure of the deformable model and the sets of actuation forces that, when applied to the model, lead to deformed states, $\mathbf{x}^i$, $1 \leq i \leq n_p$, that are as close as possible to n_p target states, denoted by $\mathbf{t}^i$, $1 \leq i \leq n_p$. Letting $\mathbf{p}$ denote the vector of generic parameters that we want to optimize for, we have seen in Chapter 3 that this objective could be cast as a constrained minimization problem

$$\begin{aligned} &\min_{\mathbf{x}^i, \mathbf{p}} \sum_i^{n_p} E_d^i(\mathbf{t}^i, \mathbf{x}^i) \\ &\text{s.t} \quad \mathbf{f}^{\text{int}}(\mathbf{x}^i, \mathbf{p}) = -\mathbf{f}^{\text{ext}}(\mathbf{x}^i, \mathbf{p}) \quad \forall i \in 1 \ldots n_p\ , \end{aligned} \tag{5.6}$$

where $E_d^i(\mathbf{t}^i, \mathbf{x}^i)$ penalizes the deviation of each deformed pose $\mathbf{x}^i$ from its corresponding target pose $\mathbf{t}^i$. For this application, we found sufficient to use a distance energy function that measures the Euclidean distance between the positions of the model's boundary vertices in the deformed configuration and the target configuration as

$$E_d^i(\mathbf{t}^i, \mathbf{x}^i) = \sum_{j \in \mathcal{B}} ||\mathbf{x}_j^i - \mathbf{t}_j^i||^2\ , \tag{5.7}$$

where $\mathcal{B}$ denotes the set of boundary vertices. Since some of our design parameters $\mathbf{p}$ are subject to additional bound constraints, we solve the minimization problem (5.6) with the Gradient Projection Method detailed in Section 3.2.2.

We will now explain how we apply this formulation to the optimization of number and locations of actuators and object's material distribution.

5.4.2 Initial Actuation

Given a deformable character and its desired range of deformation, we first have to determine a number of actuators and an initial estimate for their locations. In some cases, the structure of a character will largely imply the number of actuators. In other cases, the user can have a specific idea of how many actuators should be used and where they should be placed. Yet, for many characters, in particular those without apparent skeletal structure, the answer to this question is far from obvious.

In order to treat both of these cases, we support two variants of actuator layout in our system: the user can hand-select a desired number of points on the undeformed model, in which case we directly proceed to the next stage of our pipeline. Otherwise, we ask the user to specify an admissible range for the number of actuators (e.g. 5-10) and automatically compute a suggestion as described subsequently.

The underlying reasoning of our approach is that it is generally desirable to have the smallest number of actuators that yield a sufficiently good approximation of the target poses. Indeed, the mechanical complexity of real-world actuation systems typically imposes strict bounds on the number of actuators, as is the case for our string-based setup (see Figure 5.4) and animatronic figures in general (see, e.g., [Bickel et al., 2012]). Finding the optimal number of actuators is an inherently discrete problem that does not directly lend itself to continuous optimization. Instead of turning toward specialized optimization methods such as mixed-integer programming, we draw inspiration from sparse regularization techniques used in, e.g., machine learning and image processing. Starting with a dense set of actuators on the boundary of the model, we introduce a regularizer that prefers sparse solutions, i.e., a small number of actuators.

A widely used approach for sparse regularization is to penalize the L_1-norm of the design variables, i.e., the sum of absolute values. In our case, the design variables are force vectors comprising two or three components. Since all of these components have to be zero simultaneously in order to obtain a zero net force, we introduce a regularizer that penalizes the sum of force magnitudes. Considering

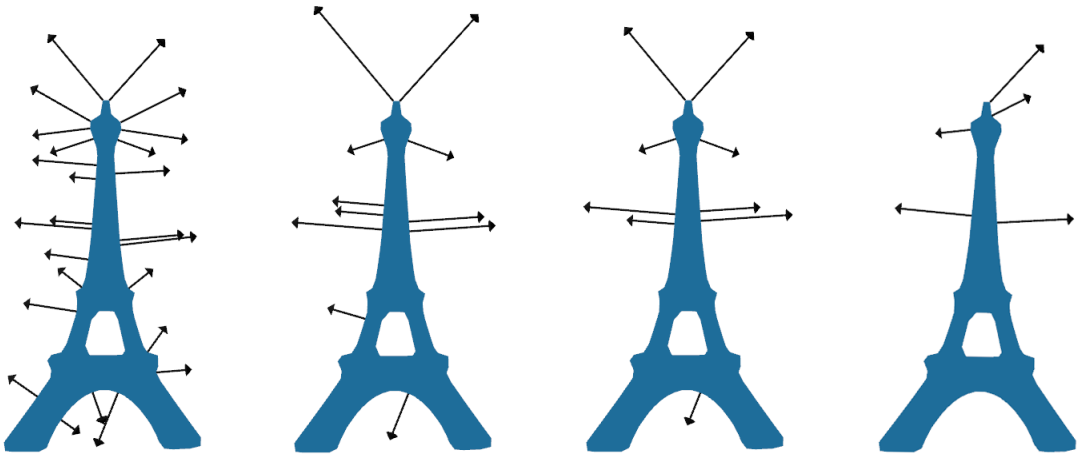


Figure 5.2: *Sparse regularization demonstrated on the* TourEiffel *example. The number of actuators (black arrows) is effectively controlled by the coefficient* k_{sparse} *(indicated below the figures).*

only a single target pose i for the moment, we define a sparse regularizer as

$$R_{\text{sparse}}(\mathbf{q}^i) = k_{\text{sparse}} \sum_j^{n_v} \left(\sum_k^{\dim} (\mathbf{q}_j^i)_k^2 \right)^{1/\alpha}, \tag{5.8}$$

where k_{sparse} is a scaling parameter and $\alpha > 2$ generalizes the L_1-norm to more strongly penalize small values. In order to incorporate multiple target poses, we ask that a given actuator should have a zero value only if its force vectors vanish for all target poses. This requirement can be modeled as

$$R_{\text{sparse}}(\mathbf{q}^1, \ldots, \mathbf{q}^{n_p}) = k_{\text{sparse}} \sum_j^{n_v} \left(\sum_i^{n_p} \sum_k^{\dim} (\mathbf{q}_j^i)_k^2 \right)^{1/\alpha}. \tag{5.9}$$

We add this regularizer to the basic optimization problem (5.6) with $\mathbf{p} = (\mathbf{q}^1, \ldots, \mathbf{q}^{n_p})$ and, depending on the value of k_{sparse}, obtain solutions with different numbers of actuators (see Figure 5.2). Since it is not possible to choose k_{sparse} *a priori* in order to obtain a desired number of actuators n_a, we solve a sequence of problems with different values of k_{sparse} until we find a solution within the admissible range specified by the user.

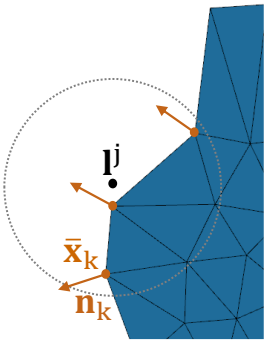
5.4.3 Actuator Locations

The initial actuation step provides us with a number of actuators and their initial locations. Next, we improve the matching quality, i.e., the correspondence between

deformed poses $\mathbf{x}^i$ and target poses $\mathbf{t}^i$, by allowing the actuators to slide along the boundary of the character. In order to ensure that actuators can move freely along the surface but not in their normal directions, we introduce a penalty term that attracts the actuators to the zero-levelset of a local distance field Φ around the surface.

Using the implicit moving least squares (IMLS) method of Öztireli et al. [2009], the distance field is defined as

$$\Phi(\mathbf{x}) = \frac{\sum_k \mathbf{n}_k \cdot (\mathbf{x} - \bar{\mathbf{x}}_k)\phi_k(\mathbf{x})}{\sum_k \phi_k(\mathbf{x})} , \tag{5.10}$$



where $\phi_k(\mathbf{x}) = \left(1 - \frac{||\mathbf{x}-\bar{\mathbf{x}}_k||_2^2}{h^2}\right)^4$ are locally-supported kernel functions that vanish beyond their support radius h. Using this formulation, we define a penalty energy

$$E_{\text{IMLS}} = \sum_i \Phi(\mathbf{l}_i)^2 \tag{5.11}$$

that attracts the actuator locations $\mathbf{l}_i$ to the surface. In order to ensure that the actuation forces vary smoothly as the actuators move along the surface, we distribute them to a local neighborhood of vertices using the IMLS-kernels

$$\mathbf{f}_k^i = \frac{\mathbf{q}_j^i \phi_k(\mathbf{l}_j)}{\sum_{l \in \mathcal{S}_j} \phi_l(\mathbf{l}_j)} . \tag{5.12}$$

Here, $k,l \in \mathcal{S}_j$ where $\mathcal{S}_j$ denotes the set of vertices whose kernel functions are nonzero at $\mathbf{l}_j$ (see also the figure above). In addition to this penalty term, we also want to prevent actuators from moving too close together as coinciding actuators would lead to a null-space during optimization. To this end, we define a C^2 continuous repulsion term

$$E_{\text{rep}}(\mathbf{l}_j, \mathbf{l}_k) = k_{\text{rep}}(\varepsilon_{\text{rep}} - ||\mathbf{l}_j - \mathbf{l}_k||)^3 , \tag{5.13}$$

which is activated whenever the distance between actuators is smaller than a threshold value ε_{rep}.

String-Based Actuation

The formulation above assumes that the actuators can apply forces $\mathbf{q}_i$ in arbitrary directions. In the case of strings, this is no longer true (see Section 5.3.2) and a few adaptations have to be made. First, instead of optimizing for the vectors $\mathbf{q}_j^i$ directly, we optimize for the string tensions k_j^i and the string attachment points

$\mathbf{a}_j$, in addition to the actuator's location $\mathbf{l}_j$ on the rest state of the object. We enforce the condition that $\mathbf{a}_j$ be on the support structure using a formulation similar to (5.11). Assuming a circular support, we formulate the penalty energy as

$$E^j_{\text{string}}(\mathbf{a}_j) = (||\mathbf{a}_j - \mathbf{c}|| - r)^2 \,, \tag{5.14}$$

where r is the radius of the support and $\mathbf{c}$ denotes its center.

Second, in order to limit the likelihood of strings intersecting with the model or tangling up during animation, we privilege string directions that are close to the boundary normals at the attachment point. We express this preference with a further penalty term

$$E_{\text{dir}} = k_{\text{dir}} \sum_i^{n_p} \sum_j^{n_a} \left(1 - \frac{\mathbf{q}^i_j}{||\mathbf{q}^i_j||} \cdot \mathbf{n}_i(\mathbf{l}_j) \right)^2 \,, \tag{5.15}$$

where $\mathbf{n}_i(\mathbf{l}_j) = \frac{\sum_k \mathbf{n}^i_k \phi_k(\mathbf{l}_j)}{\sum_k \phi_k(\mathbf{l}_j)}$ denotes the interpolated normal at the string location $\mathbf{l}_j$ in pose i.

5.5 Material Optimization

Even with the location optimization described in the previous section, characters can have poses that are difficult to achieve. Such difficult poses arise, e.g., from conventional articulation such as the sharp bending of an arm. Approximating such poses with a homogeneous material would require a large number of actuators. We address this problem by allowing the material properties to vary spatially, thus building preferences for deformation directly into the model.

As a basis for material optimization, we will assume that there is a library of non-miscible base materials (such as silicone and printable plastics) described by energy density functions W^i. For simplicity, we restrict considerations to two material types per character, typically a soft and a stiff one. We allow the material properties to vary among the elements but assume that each element consists of a homogeneous material. If we directly constrain each element to take on only material properties from the library, we arrive at a discrete optimization problem and its associated difficulties. In order to avoid the need for more complex optimization methods, we convert the discrete problem into a continuous one by allowing the per-element materials to be interpolations of the base materials. We start the optimization by allowing arbitrary (convex) combinations of the base materials and then progressively drive the interpolation weights to the boundary of the intervals, thus enforcing a discrete material distribution. On a technical level,

we do not interpolate material descriptions but the elastic energies that would result from the different base materials. The effective deformation energy of a given element e is defined as

$$W(\mathbf{F}^e, \rho^e) = \rho^e \, W^1(\mathbf{F}^e) + (1 - \rho^e) \, W^2(\mathbf{F}^e) \, , \tag{5.16}$$

where ρ^e are interpolation weights. Adopting this interpolated material model and adding the interpolation weights per element as free variables to the optimization problem, we can solve for a material distribution that leads to an optimal approximation of the target poses by optimizing for the parameters $\mathbf{p} = (\mathbf{q}^1, \ldots, \mathbf{q}^{n_p}, \rho^{e_1}, \ldots, \rho^{e_n})$. However, in order to obtain a physically meaningful solution, we have to drive the interpolation weights to 0 or 1. We achieve this with a penalty energy of the form

$$R_{\text{mat}} = k_{\text{mat}} \sum_e \rho^e (1 - \rho^e) \, , \tag{5.17}$$

where k_{mat} is a scaling parameter that is progressively increased until a solution is found that satisfies $\rho^e(1 - \rho^e) < \varepsilon$, where ε denotes a small threshold value. While the penalty term R_{mat} will eventually ensure that all interpolation weights are either 0 or 1, they can potentially assume values out of this range in earlier iterations. We therefore have to explicitly enforce the bounds on the interpolation variables since we could otherwise encounter non-physical material combinations that would hinder convergence.

The material optimization scheme computes material distributions that are optimal in the sense of approximating the target poses. However, multiple solutions can lead to equivalent approximations of the target poses. In our approach, we aim to find macroscopic material distributions rather than micro-level structures. We therefore favor larger material clusters rather than small, isolated islands by adding the regularizer,

$$R_{\text{smooth}} = k_{\text{smooth}} \sum_j (\rho^j - \frac{1}{n_j} \sum_{k \in \mathcal{T}_j} \rho^k)^2 \, , \tag{5.18}$$

where $\mathcal{T}_j$ denotes the set of n_j elements adjacent to element j. Finally, since our focus is on deformable characters, we also prefer soft materials over stiff ones in regions where the difference in approximation quality is small. We model this preference with a second regularization term,

$$R_{\text{soft}} = k_{\text{soft}} \sum_i^{n_e} (\rho^e)^2 \, , \tag{5.19}$$

assuming that the soft material corresponds to $\rho = 0$.

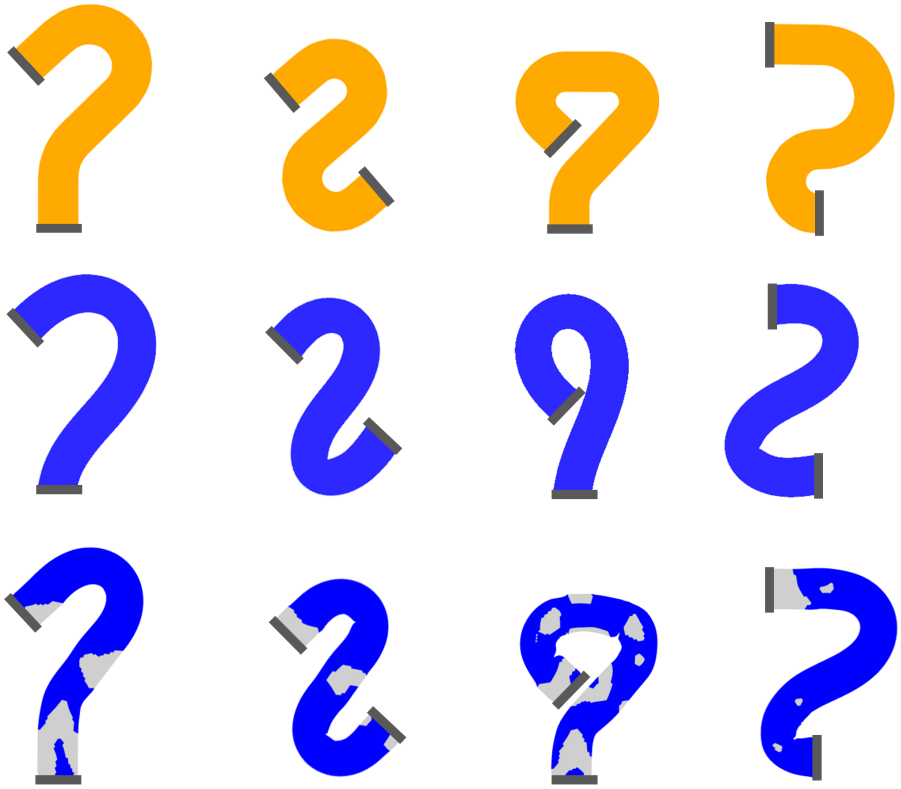


Figure 5.3: *A straight bar is deformed into four different question mark shapes by imposing position constraints on its end caps. The first row shows the target shapes, the second row the result without material optimization and the third our results after material optimization (stiff/soft material shown in grey/blue color).*

5.6 Results

We evaluated our method by designing and fabricating six characters with different types of actuators and materials. For fabrication, we used an Objet Connex350 multi-material printer with two base materials of significantly different stiffnesses. We chose VeroClear, a rigid and transparent material, and TangoBlack+, a black material with properties similar to soft rubber. Three of the characters were printed directly ("Palmy3D", "Questionmark", "Grampolo"), whereas the remaining three were fabricated using silicone injection molding with 3D-printed rigid parts ("TourEiffel", "Palmy2D", "WormEye"). We simulated the material behavior during the design process using measured data for silicone [Bickel et al., 2012] and data provided by Objet for the 3D-printed materials. In the following paragraphs, we discuss our results and highlight the roles played by the individual stages of our design pipeline.

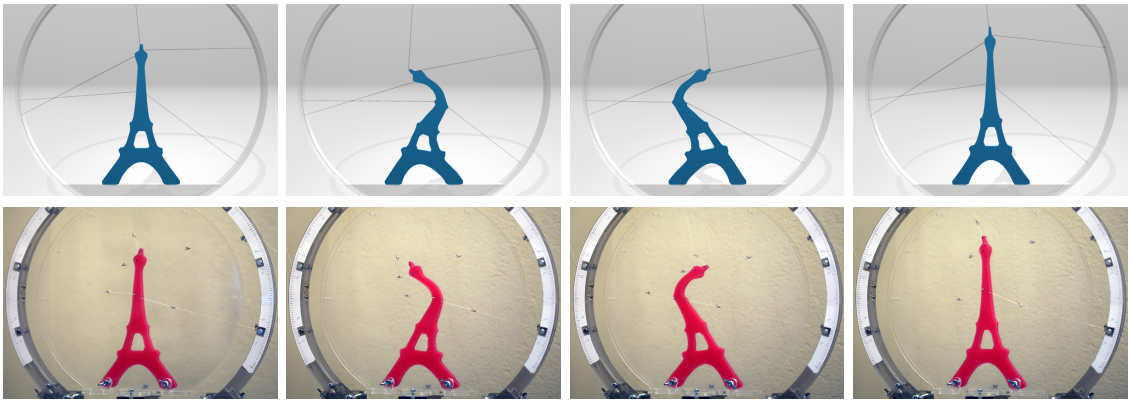


Figure 5.4: *"TourEiffel" model. From a given animation, we extracted several key frames and automatically optimized the actuator locations. The top row shows the resulting deformations in simulation. The bottom rows shows the corresponding poses when using the fabricated model.*

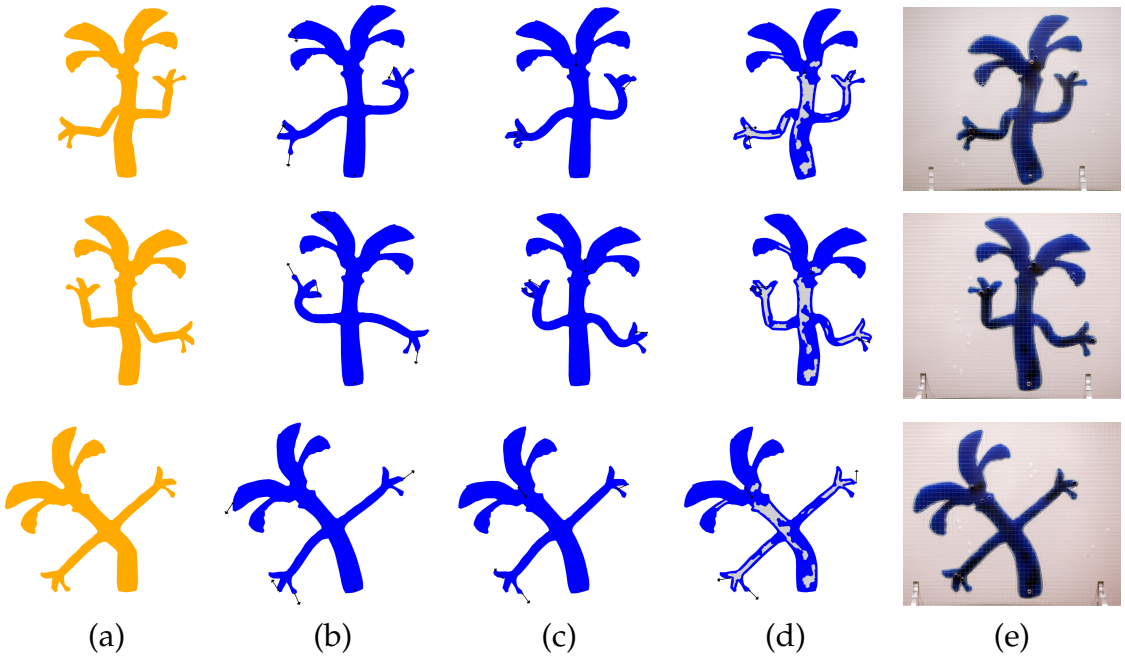


Figure 5.5: *"Palmy2D." Given 3 target poses (a), we use sparse regularization to find an initial location of 4 actuation points (b). We then refine the location of the actuation points by allowing them to slide on the surface (c) and optimize for an internal material distribution that allows us to better approximate the target poses (d). Columns (b-d) show the resulting simulated deformations of each optimization stage. Column (e) shows the fabricated character that we posed using pins.*

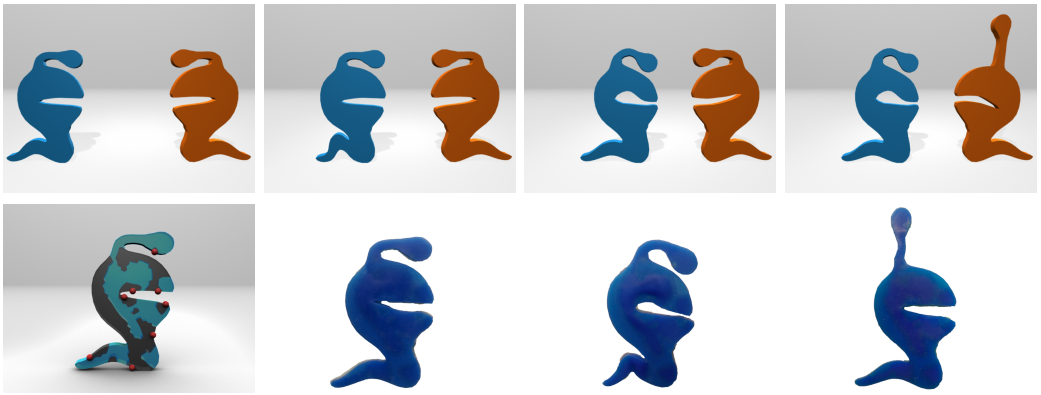


Figure 5.6: *"WormEye." Top row: Frames from an animation of two WormEye characters. Bottom row: Optimized material distribution and actuator locations for reproducing poses present in the animation (left) and real fabricated deformed character (right).*

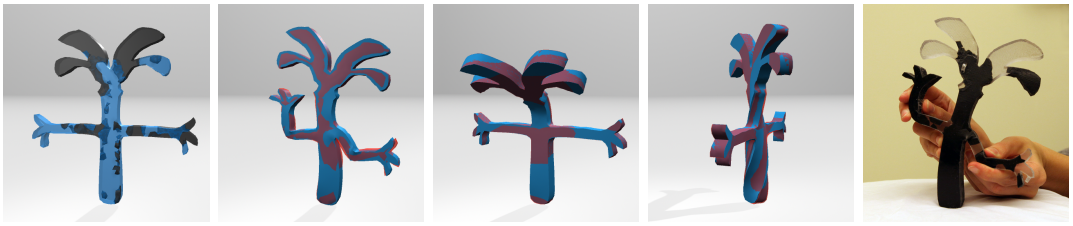


Figure 5.7: *"Palmy3D." From left to right: Rest pose with optimized material distribution (black/stiff, blue/soft), simulated poses (blue) overlaid with input target poses (red), 3D printed character.*

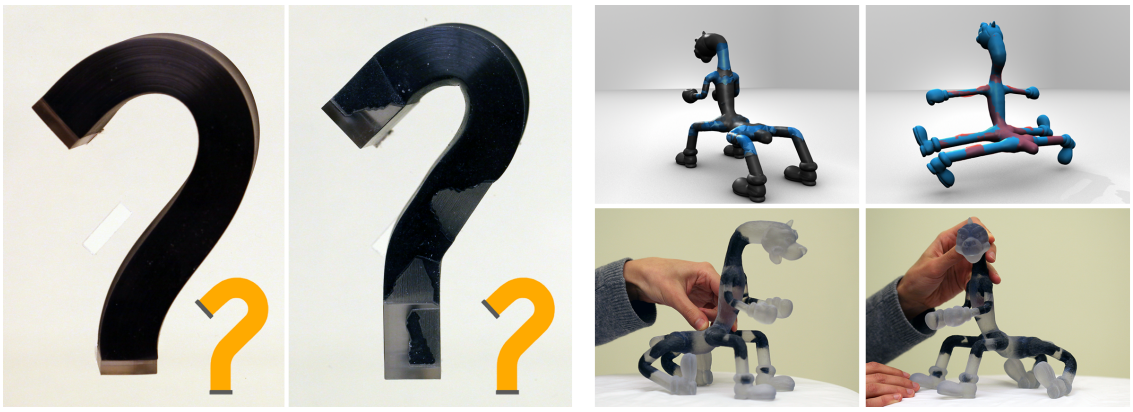


Figure 5.8: Right: *A straight bar with uniform (left) and optimized (right) material distribution is bent into a question mark shape by applying force only on its end caps. The small orange icon shows the target shape.* Left: *"Grampolo". Top row: Illustration of the rest pose with optimized material distribution (black/stiff, blue/soft) on the left and the simulated deformed mesh (blue) overlayed with the input target pose (red). Bottom row: Fabricated character actuated by hand.*

Model	Pose1	Pose2	Pose3	Pose4	Pose 5	Size
TourEiffel (1)	4.8/0.5	3.0/0.4	2.8/0.5	6.4/0.8	4.4/0.6	202
TourEiffel (2)	3.6/0.5	2.0/0.4	3.6/0.5	5.2/0.6	3.8/0.6	202
Palmy2D (1)	7.1/0.7	11.0/0.9	5.7/0.4	-	-	100
Palmy2D (2)	7.6/0.4	7.4/0.5	4.8/0.2	-	-	100
Palmy2D (3)	5.1/0.5	5.1/0.3	2.6/0.1	-	-	100
Questionmark (2)	8.3/0.4	-	-	-	-	80
Questionmark (3)	3.1/0.2	-	-	-	-	80
Grampolo (1)	17.1/3.7	-	-	-	-	102
Grampolo (2)	13.6/2.6	-	-	-	-	102
Grampolo (3)	2.7/0.4	-	-	-	-	102

Table 5.1: *Error statistics. Max/mean Euclidean distance (in mm) between vertices of the simulated poses and vertices of the input poses after initial actuation (1), optimization of actuation locations (2), and optimization of material distribution (3). The size corresponds to the maximum length of the character's bounding box.*

Actuator Optimization

For all examples shown in this chapter, we employed the sparse regularization approach for computing an appropriate number of actuators. We observed that, especially for characters that do not have an obvious internal structure, manually selecting the number of actuators and their placement proved to be difficult. Our automatic approach significantly simplified this design task.

Figure 5.4 shows an example of string-based actuation. We extracted five key frames from an artist-generated input animation and used the sparse regularization method to automatically determine the number of actuators as well as their initial locations. We fabricated the model with silicone and attached five strings driven by servo motors to playback the animation. The strings are routed through pulleys at a ring around the model, whose locations are optimized as well.

The "QuestionMark" example (Figure 5.8, *left*) is posed using clamp-type actuators that fix both the position and orientation of the end caps of the model. The remaining examples were designed using pin-type actuators. We pose the planar characters by attaching small pins at the actuators locations. These pins are then plugged into precision-drilled holes in an acrylic plate to reproduce the target poses. The 3D characters, "Palmy3D" and "Grampolo", were designed using the same framework as for the 2D examples, but for simplicity, we pose and animate these models by hand.

The first stage of our pipeline provides an initial guess for the actuator locations and already leads to fair approximation quality in some cases. As can be seen from Table 5.1, however, the subsequent actuator location optimization is able to significantly reduce the error for all examples.

Model	#Elem.	#Actuation Points	Computation Time	Weights
TourEiffel	525	5	20min02s	1/-/-
Palmy2D	2745	4	5h26min	$0.01/10^{-5}/0.01$
Palmy3D	8362	4	7h17min	$0.1/10^{-7}/0.01$
WormEye	808	8	3h10min	$1.5\,10^{-4}/10^{-6}/0.1$
Questionmark	4536	caps fixed	1h09min	$-/10^{-7}/1$
Grampolo	17709	7	4h44min	$500/10^{-3}/1$

Table 5.2: *Example statistics. From left to right: Number of elements, number of actuation points, total computation time, weights used for the regularization terms* R_{sparse}, R_{soft} *and* R_{smooth}.

Material Optimization

Allowing material properties to vary spatially further improves the visual and quantitative error of all characters. A particularly striking example can be seen in Figure 5.3, where material optimization allowed us to create very different deformation styles with only two actuators. Note that our scheme leads to intuitive solutions if the character exhibits mostly rigid articulation, as is the case, e.g., for "Palmy2D" (Figure 5.5) and "Grampolo" (Figure 5.8, *right*). Although we used only one example pose for the "Grampolo" character, the optimization scheme was able to infer a meaningful material distribution, putting soft material at joint locations and rigid material at limbs. For characters with more complex deformations, however, the material distribution can be significantly more complex as shown in Figure 5.6 and Figure 5.3.

Weight Selection

Our method uses a number of different penalty terms to enforce soft constraints or drive the solution toward a preferred subspace. The weights that we used for the different examples are listed in Table 5.2. For most of the weights, determining an appropriate value posed no difficulty, since the corresponding penalty terms were not directly competing with other objectives. However, the weights of the regularization terms (5.18) and (5.19) have a significant impact on the final material distribution. We set these weights by first selecting a value for (5.19) which, together with the target poses, determines the overall material structure. We use a default value for (5.18) and, if necessary, adjust it with 1-2 iterations to suppress small material islands without changing the boundary of larger structures too much.

Accuracy, Robustness and Performance

Our fabricated prototypes show good agreement with the simulation. Figure 5.5 illustrates the progressive improvement during each step of our pipeline including the final fabricated character.

A variation in the position of the actuators affects the resulting material distributions. However, in practice we observed that a small variation leads to a solution with similar quality. We also investigated whether multiple iterations of our pipeline would lead to improved results but did not observe any significant improvement after the first cycle in our experiments.

Statistics for each example including the number of elements, number of actuation points and computation times can be found in Table 5.2. The largest fraction of the computation time is spent on material optimization. This is mostly due to the fact that the optimization scheme uses several outer iterations to increase the parameter k_{mat} that eventually enforces each element to assume one of the base materials.

5.7 Discussion and Outlook

This chapter described a method for creating physically fabricated prototypes of animated digital characters. Our approach automatically finds a sparse set of actuation locations on the surface and optimizes the internal material distribution such that the resulting character exhibits the desired deformation behavior. We demonstrated our method on a set of simulated as well as physically-fabricated characters with different types of actuators and materials.

We optimize for sets of actuation forces corresponding to equilibrium states that are as close as possible to the target poses. However, knowing the forces at the deformed state does not imply that there is a unique way of getting to that state from the undeformed configuration as there can be bifurcation points (buckling) along the way. While we did not encounter this sort of problems in our examples, further treatment might be necessary in order to ensure robust tracking of the input animation in between target poses.

One possible direction for future work could be to explore the possibility of using a larger number or range of materials, e.g., by printing micro-level structures. An interesting related problem is also the design of more elaborate actuation systems that would enable us to animate more complex characters in an automated way.

With the applications considered so far, we demonstrated that automatic computation of rest shapes, material distributions and actuation parameters was not only possible but also effective in order to design physically valid objects with

desired deformation behaviour. In the next chapter we will furthermore show that automation does not necessarily means no room for the designer to express his creativity and that user-input and optimization can be combined in order to guide the resulting designs towards aesthetically pleasing solutions.

CHAPTER 6

Conceiving Inflatable Structures

This chapter investigates inflatables and presents a dedicated tool combining automatic pattern generation and an intuitive user interface to design this type of structures. The design of inflatable balloons was already explored in Chapter 4. However, the latter focused on rubber balloons which present notable differences with the structures that interest us in the chapter. In particular, rubber balloons stretch significantly during inflation and have 3D rest shapes. By contrast, inflatables have great membrane stiffness and negligible stretching strain, and are manufactured by assembly of piecewise *flat* rest shapes. These two characteristics largely affects the simulation and the final appearance of the structure – seams between adjacent panels are generally visible – and must be taken into account when developing a proper design tool. In this chapter, we present our solution to address these specific challenges.

6.1 Introduction

Inflatables are structures made of flat membrane elements that assume complex curved shapes when pressurized. Thanks to their lightweight nature, rapid deployment, and cost efficiency, they enjoy widespread popularity in entertainment, advertisement, engineering, and architecture. Made of planar panels, they are easy to maintain and repair, allow for large physical dimensions and permit decoration with paint, ink, or other appliques using standard printing methods. Moreover, they can be quite stiff when inflated, expanding the potential functionality be-

yond the realm of purely decorative: inflatable furniture and portable architectural structures stand alongside foil balloons and parade floats.

Designing inflatable structures requires solving a complicated *patterning* problem: what is the shape of the flat panels that we must cut, and how must we interconnect the panels, such that the final assembly inflates to the desired curved shape? This task is extremely challenging since the designer must anticipate, and invert, the effects of pressure on the shape of the structure, while simultaneously taking into account the aesthetics of seams. The combination of functional and aesthetic requirements make patterning the most difficult aspect of current manual design processes.

Our goal in this chapter is to provide a tool to help the designer to make plans for an inflatable structure that corresponds to a given target shape. Our approach ensures that the designer retains full control over aesthetic considerations by discarding fully automated processes in favor of an interactive, optimization-in-the-loop methodology. As the designer sketches the proposed placement of seams, the underlying optimizer alleviates the iterative and error-prone tasks of reverse-engineering the physics of inflation, proposing a set of panels that best accommodate the desired seams and target shape.

6.2 Overview

We propose an interactive system to easily design inflatable structures that correspond to desired shapes. The workflow of this system is the following: We assume that the designer already has a certain target shape at hand, perhaps acquired from real world data, designed via modeling software, or provided as a specification by a client. After loading the shape in the tool, the user draws a network of seams defining desired segment boundaries in 3D. As the user creates new segments or modifies existing seams, an underlying optimization component automatically generates flat panels for the segments such that the inflated structure is as close as possible to the target while satisfying the desired seam positions. Once the user is satisfied with the result, the generated panels can be used to fabricate a physical prototype. We demonstrate the suitability of our tool on a varied set of simulation examples, some of which we have fabricated, demonstrating excellent agreement with the design intent.

Our main contributions in this chapter can be summarized as follows:

- We describe a tool to design custom inflatable structures that allows interactive exploration of different seam layouts, including internal connections and

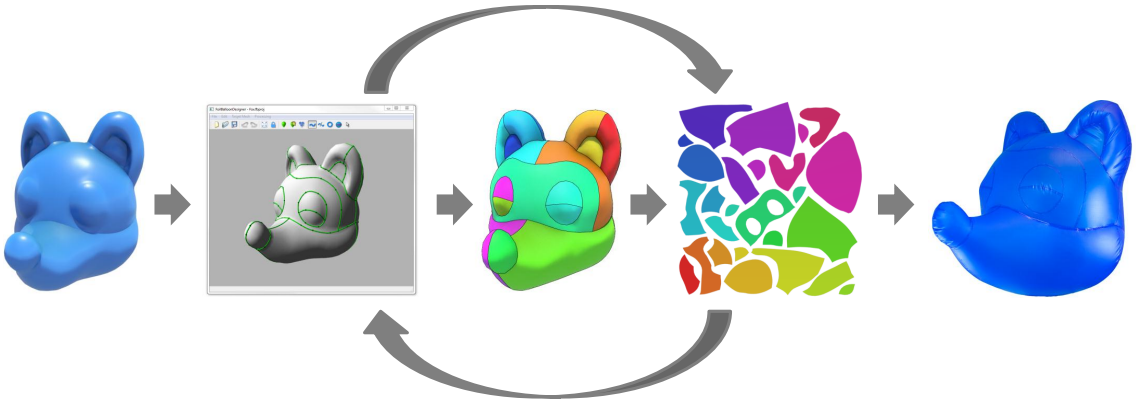


Figure 6.1: *Pipeline Overview: Given a target shape* (left)*, the user sketches seams defining desired segment boundaries on the target mesh* (second from left)*. The system automatically flattens and optimizes the segments such that the inflated structure* (middle) *is as close as possible to the target. The generated flat segments* (second from right) *can then be used to fabricate a physical prototype* (right)*.*

their effects on the inflated shape. Our tool does not require any modeling expertise since the seams are directly specified on the target mesh.

- We propose a fast physics-based pattern optimizer with integrated remeshing which supports quasi-inextensible materials. Our optimizer relies on accurate coarse-scale simulation of wrinkling inflatable membranes motivated by tension field theory and on a dedicated constrained optimization method that allows adjustable balance between expected shape approximation and seam quality.
- We demonstrate the capabilities of our method by designing a representative set of inflatable structures with complex shapes and elaborate seam layouts and we validate the feasibility of our designs on three physical prototypes.

The remainder of this chapter is organized as follows. After precisely defining the notion of inflatable structure in Section 6.3, we describe the interface of our design tool in Section 6.4. Section 6.5 presents our model for simulating inflatable membranes and clarifies its connection with tension field theory. Section 6.6 describes our pattern optimizer and provides information on technical details. Our results are discussed in Section 6.7. and are followed by final remarks in Section 6.8.

Figure 6.2: *Internal connections* (left) *allow the designer to realize shapes with sharp creases and concave features* (middle) *that would be poorly approximated without internal structures* (right).

6.3 Anatomy of Inflatable Structures

Inflatable structures are made from flat *panels*, i.e., thin layers of metallic foil, vinyl, or textile. While our approach does not exclude stretchy materials such as rubber per se, we target structures that show little stretch but large bending deformations. We therefore focus on quasi-inextensible materials that exhibit a high resistance to stretching, but are compliant to bending.

A connection between two panels is called a *seam*. Depending on the type of material, seams are created through gluing, heat sealing, or stitching. As an important design constraints, the segments forming the seam should have the same length on both panels. Otherwise, the design must be altered during manufacturing using cuts or pleats.

The shape of inflatable structures is governed by the requirement that they have to be *stable under pressure*: the pressure forces must be balanced by membrane forces in every point on the surface. This equilibrium constraint puts limits on what kind of shapes can be obtained with a structure that consists of single closed surface. However, the space of possible designs can be significantly enlarged by allowing for *internal connections* (see Figure 6.2). Such internal connections can be used to attach parts of the surface to each other that would otherwise be pushed apart by the pressure forces. They can also be used to create creased feature curves.

Having defined an inflatable structure, we now focus on the different features of our design tool.

6.4 Design Tool Interface

This section presents the design system from a user perspective. We start by describing the different windows of the interface (Section 6.4.1) then explain how seams can be drawn on the surface of the mesh (Section 6.4.2) and possibly connected to create internal panels (Section 6.4.3).

6.4.1 Views

The design interface consists of three views: the *inflate* view, *target* view, and *pattern* view (see Figure 6.3). Each new design session starts by loading a closed triangle mesh that represents the 3D target shape. The user works in the *target* view where he incrementally builds a seam layout that partitions the target mesh into a set of segments. Once a seam layout or edit is committed, our system flattens the corresponding segments and begins pattern optimization in the background. The *inflate* view immediately shows a preliminary shape for the resulting inflated structure, obtained by inflating the model with the current patterns, that are shown in the *pattern* view. Both views are continuously updated while the optimization proceeds in the background.

Additionally, three different display modes are available in the *inflate* view: the *segments* mode, the *compression field* mode and the *internal panels* mode. The *segments* mode allows to easily identify the different pieces of the inflatable structure and to match them with the corresponding flat patterns. In the *compression field* mode, compressed areas are overlaid in red, thus providing the user with an indication of the location and amplitude of the expected wrinkles (see Figure 6.6). Finally, the *internal panels* mode serves to visualize the shape of the internal structures in the simulation (see Figure 6.4) .

6.4.2 Seam Design

The user draws seams directly on the target model using a spline tool that implements a *geodesics* metaphor, i.e., connects seam points by taking an approximately shortest path on the surface. We represent seam curves using cubic Hermite splines that are defined through a coarse set of 3D control points. For segmentation, we simply project the spline curve onto the surface mesh.The seam tool supports snap-on functionality in order to link new seams to existing ones. In addition, the user can also edit existing seams by simply dragging control points, as well as delete selected seams.

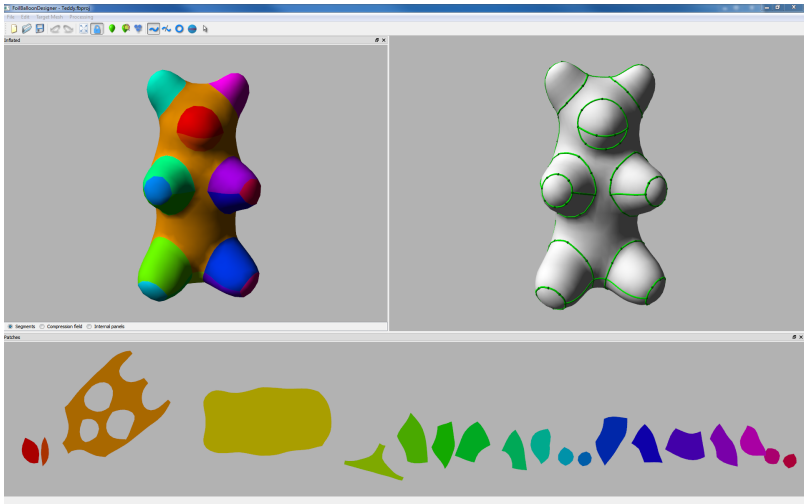


Figure 6.3: *Our design interface, showing the inflated view* (top left)*, target view* (top right)*, and pattern view* (bottom)*.*

Generally, some of the seams will have aesthetic purposes or requirements, while others simply subdivide a larger region in order to increase the shape approximation quality. For each of the seams, the user can therefore specify a weight that indicates how important it is for the seam to remain in its original location with respect to the target shape. These conditions are then enforced through corresponding objectives during pattern optimization.

6.4.3 Internal Connections

As a necessary condition for a given shape to be a feasible balloon, it has to be stable under pressure. Clearly, this requirement limits the space of shapes that can be realized as balloons. However, the design space can be significantly enlarged by allowing for internal connections, i.e., panels that are not visible from the outside and serve a purely functional purpose. Internal connections are created by connecting existing seams on the surface as indicated by the user (see Figure 6.4). Technically, internal connections are no different from the other, visible patches. However, paired with our optimization, internal connections provide a powerful tool for creating complex shapes with distinct features (see Figure 6.2).

In the next section, we describe our approach to accurately simulate the inflated mesh from a technical point of view.

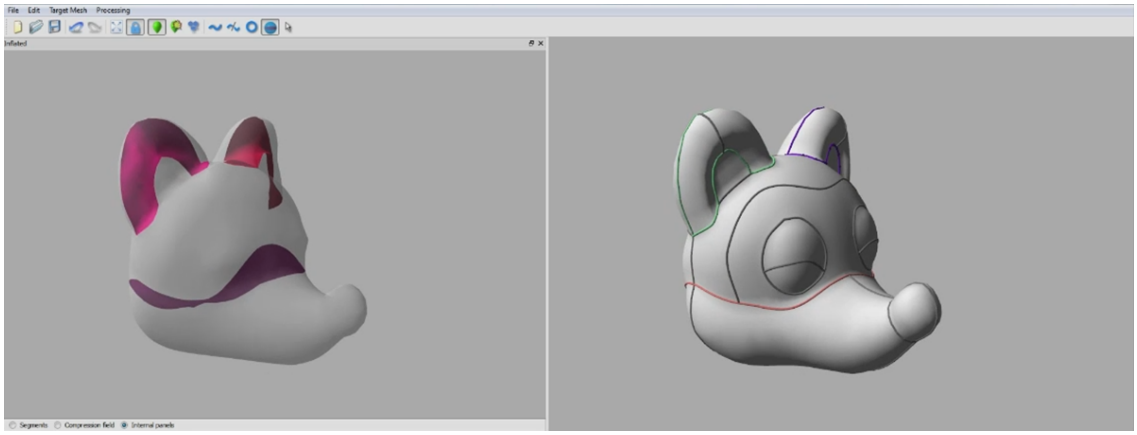


Figure 6.4: *Internal panels, visible in the* inflate *view (*left*) can be easily created by selecting sets of seams in the* target *view (*right*).*

6.5 Simulation

As a core component of our design system, we must be able to rapidly compute the deformed shape of inflatable structures. Like many other thin structures, inflatables exhibit a strong resistance to stretching but will wrinkle at the onset of compression. This behavior poses challenges that our simulation must confront efficiently. This section explains the strategy that we employ to address these difficulties. We start by investigating the cause of compressive deformations in inflatable structures (Section 6.5.1), then present our solution based on tension field theory that allows us to approximate wrinkling regions by a smooth surface (Section 6.5.2). Our method relies on a modified membrane energy formulation that is used in our discrete treatment of the membrane mechanics as described in Section 3.1.2.

6.5.1 Origins of Compression

As a didactic example, consider the simple foil balloon depicted in Figure 6.5, assembled from two disc-shaped panels. When inflating the balloon, we expect that (i) the distance between the centers of the two panels increases due to *pressure*; (ii) the seam remains on its original plane due to *symmetry*; (iii) each radial line, extending from the center to the seam, will remain unstretched due to *inextensibility*. To meet these requirements the diameter *must* decrease during inflation. Correspondingly, the circumference must shrink, implying a compressive deformation on the seam that is resolved through the typical wrinkles observed in foil balloons (Figure 6.5, *right*).

Wrinkling is a characteristic trait of thin surface structures, but a major struggle

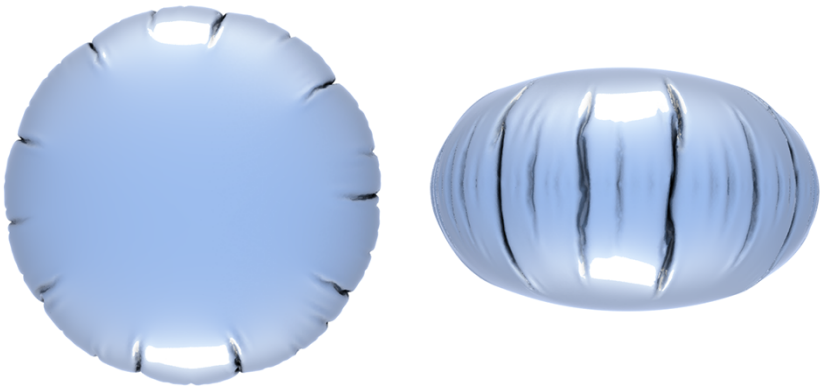


Figure 6.5: *Compression-induced wrinkling in a simple foil balloon.*

for simulation codes. First, compressions give rise to negative eigenvalues in the energy Hessian, thus breaking the fundamental assumption of most fast linear solvers, i.e., a positive-definite matrix. Second, since the location of wrinkles can usually not be predicted, a simulation mesh with uniformly high resolution is required. Clearly, both these properties are highly detrimental to efficiency.

The problem of compressions in thin surfaces is not new to graphics. For example, Choi and Ko [2002] proposed a modified mass-spring system that allows for stable animations of buckling cloth. Both the original work and its extension to triangle meshes [Choi and Ko, 2003] handle compressions along buckling springs that connect pairs of particles that are at topological distance two. However, in order to provide accurate results even for coarse simulation meshes, we would like a model that is able to handle compressions along arbitrary directions, irrespective of mesh structure.

6.5.2 Tension Field Theory

The wrinkling of membranes has been intensively studied in mathematical and physical sciences [Pipkin, 1986; Steigmann, 1990]. The difficulty of wrinkling analysis stems from the fact that the elastic energy density is not convex in the presence of compressions, jeopardizing the uniqueness and existence of solutions. Tension field theory offers a solution to this problem by postulating a *relaxed* energy density that reflects the average energy value in wrinkled regions and gracefully fades to zero before compressive stresses can occur. The underlying reasoning is that while wrinkled regions can carry longitudinal loads (so called tension trajectories), they do not exhibit resistance to transverse deformations. This formulation enables a macroscopic treatment of wrinkling that accurately captures

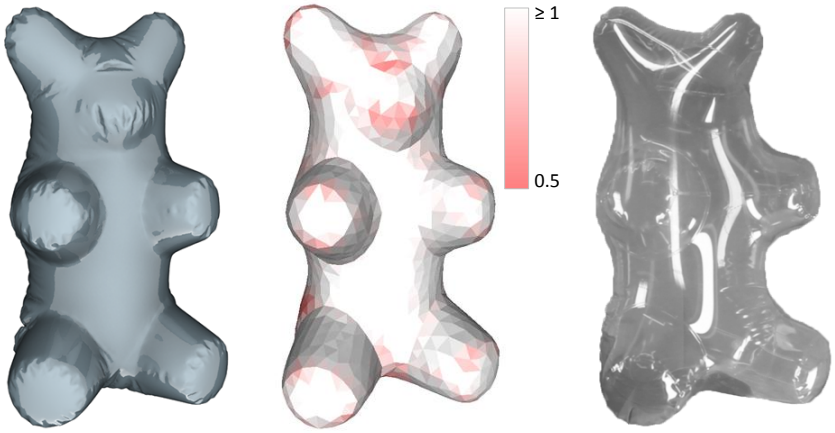


Figure 6.6: *Wrinkling analysis using the full model (*left*), compressive deformations from the tension field model (*middle*), real-world prototype (*right*). Colors indicate compressed elements with deformations ranging from $\lambda_2 = 0.5$ to $\lambda_2 = 1$.*

the deformation behavior on the coarse level but abstracts away geometric detail. There are two important advantages of this approach, both of which translate directly into computational efficiency: it requires fewer elements and it removes the problems due to indefiniteness. We will make this concept precise below.

Although the tension field approach does not directly provide geometric information on the wrinkles, the compression field does give strong indications on the locations and the directions of the expected wrinkles. We found these regions to be in very good correspondence with both the locations predicted by a high-res simulation of the full model and our actual, fabricated prototypes (see Figure 6.6).

Relaxed Energy Density

As detailed in Section 3.1.4, the deformation around a given point on the surface can be measured by a 2×2 matrix, the right Cauchy Green tensor $\tilde{\mathbf{C}}$, as defined by (3.30). The spectral decomposition of $\tilde{\mathbf{C}}$ allows us to write the right Cauchy Green tensor in terms of its principal stretches λ_1, λ_2 and corresponding eigenvectors $\mathbf{N}_1$, $\mathbf{N}_2$ as

$$\tilde{\mathbf{C}} = \lambda_1 \mathbf{N}_1 \mathbf{N}_1^t + \lambda_2 \mathbf{N}_2 \mathbf{N}_2^t \,. \tag{6.1}$$

Without loss of generality, we assume that $\lambda_1 \geq \lambda_2$. Assuming that the material is incompressible and does not exhibit transverse shearing, we can expand $\mathbf{C}$ to the 3×3 tensor

$$\mathbf{C} = \begin{bmatrix} \tilde{\mathbf{C}} & 0 \\ 0 & J^{-1} \end{bmatrix} , \tag{6.2}$$

where $J = \lambda_1\lambda_2$ is the determinant of $\tilde{\mathbf{C}}$. This deformation measure is amenable to standard material models and we opt for a Neo-Hookean material, whose strain energy density is defined in terms of the principal stretches as

$$\psi = \kappa\left(\operatorname{tr}(\mathbf{C}) - 3\right) = \kappa\left(\lambda_1 + \lambda_2 + \frac{1}{\lambda_1\lambda_2} - 3\right), \tag{6.3}$$

where κ is the stiffness coefficient. As described above, the essence of tension field theory can be condensed into a relaxed strain energy density

$$\tilde{\psi}(\lambda_1,\lambda_2) = \begin{cases} \tilde{\psi}_{\text{sl}}(\lambda_1,\lambda_2) = 0 & \lambda_1 < 1, \lambda_2 < 1 \\ \tilde{\psi}_{\text{tf}}(\lambda_1,\lambda_2) = \psi(\lambda_1,\tilde{\lambda}_2(\lambda_1)) & \lambda_1 \geq 1, \lambda_2 < \tilde{\lambda}_2(\lambda_1) \\ \tilde{\psi}_{\text{fm}}(\lambda_1,\lambda_2) = \psi(\lambda_1,\lambda_2) & \lambda_1 \geq 1, \lambda_2 \geq \tilde{\lambda}_2(\lambda_1) \end{cases} \tag{6.4}$$

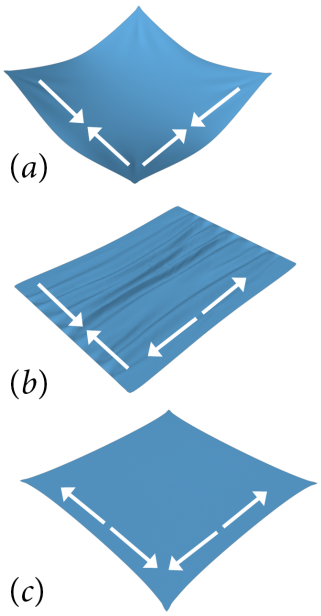
where $\tilde{\lambda}_2$ is the energetic minimum of λ_2,

$$\tilde{\lambda}_2(\lambda_1) = \underset{\lambda_2}{\operatorname{argmin}}\, \psi(\lambda_1,\lambda_2) = \frac{1}{\sqrt{\lambda_1}}. \tag{6.5}$$

This relaxed energy density can then be explicitly written as

$$\tilde{\psi}(\lambda_1,\lambda_2) = \begin{cases} \tilde{\psi}_{\text{sl}}(\lambda_1,\lambda_2) = 0 & \lambda_1 < 1, \lambda_2 < 1 \\ \tilde{\psi}_{\text{tf}}(\lambda_1,\lambda_2) = \kappa\left(\lambda_1 + \frac{2}{\sqrt{\lambda_1}} - 3\right) & \lambda_1 \geq 1, \lambda_2 < \frac{1}{\sqrt{\lambda_1}} \\ \tilde{\psi}_{\text{fm}}(\lambda_1,\lambda_2) = \kappa\left(\lambda_1 + \lambda_2 + \frac{1}{\lambda_1\lambda_2} - 3\right) & \lambda_1 \geq 1, \lambda_2 \geq \frac{1}{\sqrt{\lambda_1}} \end{cases}. \tag{6.6}$$

The three cases listed in (6.4) are illustrated in the inset figure. For the first case (*a*), the surface is assumed to be *slack*, i.e., both stretches are negative and the energy is set to zero. The second case (*b*) corresponds to wrinkling and the original model is applied with the compressive stretch replaced by its energetically optimal value. The third case (*c*) corresponds to a *taught* surface with both stretches positive for which the original model can be applied without modifications. Since the first and second cases are energetically optimal with respect to compressive stretch, the material model will never give rise to compressive stresses.



It is worth noting that the modified energy density $\tilde{\psi}$ is convex but only C^1 continuous. The discontinuities in the relaxed energy's second derivatives are clearly visible in Figure 6.7. Although we did not notice any adverse

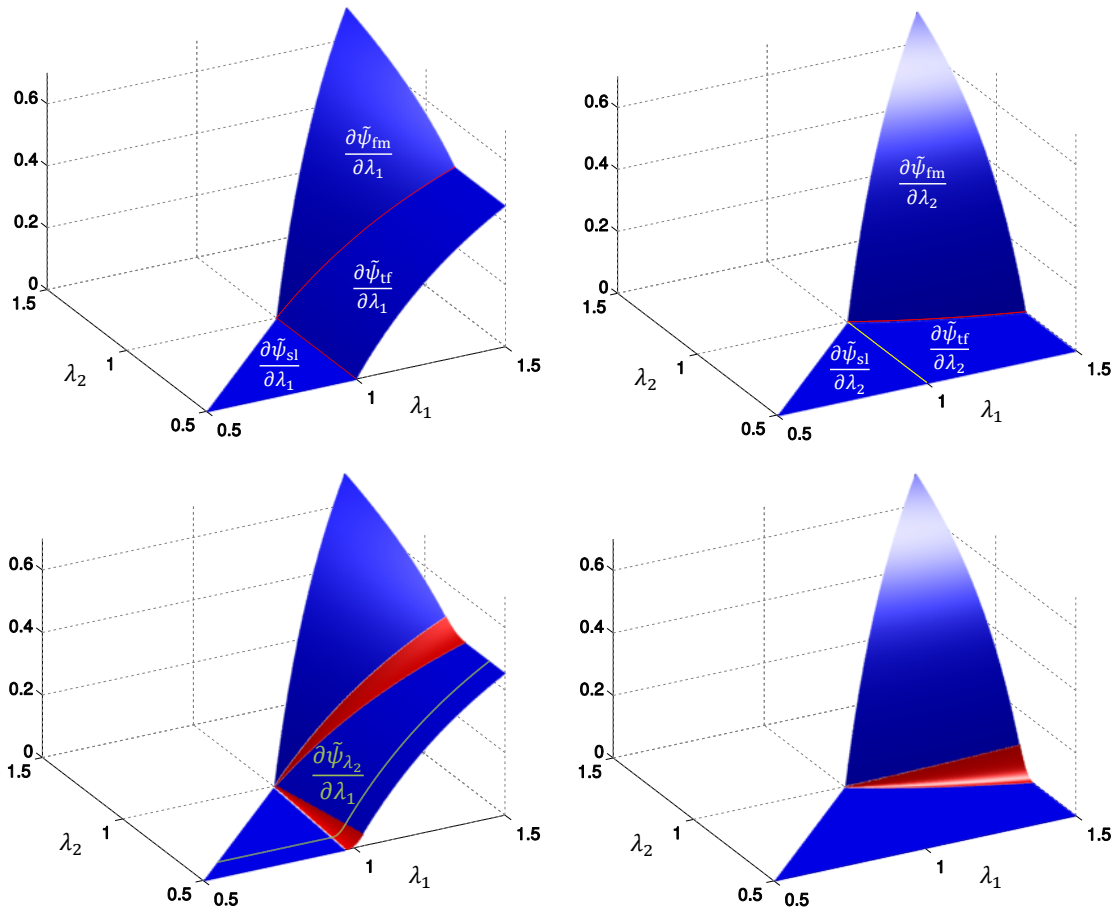


Figure 6.7: *Landscape of the relaxed energy derivatives. Before smoothing* (top)*, discontinuity lines (red) are clearly visible at the transition between the different regimes. After smoothing* (bottom)*, the discontinuities have been replaced by interpolating transition zones (red) based on quadratic functions.*

effects when solving for equilibrium states, the discontinuous force derivatives pose a significant problem for optimization since the formulation of the constraints directly depends on these forces (see Section 6.6). Instead of turning toward sophisticated methods for nonsmooth optimization, we preferred to smooth the transitions between the different regimes using quadratic interpolation. Our approach is presented below.

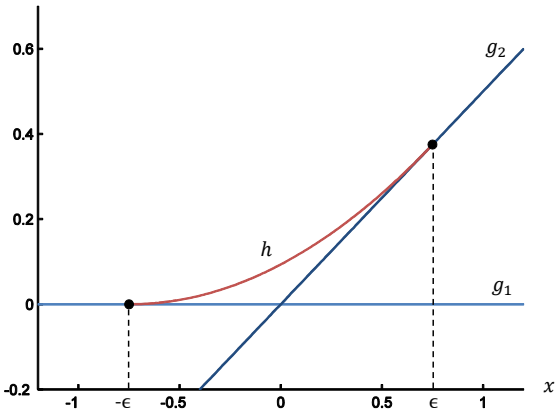


Figure 6.8: *Illustration for the construction of a transition function h interpolating the two scalar functions g_1 and g_2 of Equation (6.7) and their first derivatives.*

Smoothing the Relaxed Energy Density Derivatives

We start by considering the simple one-dimensional problem depicted in Figure 6.8, where c is a constant, $\epsilon > 0$ and g_1 and g_2 are two scalar functions defined by

$$g_1 : x \mapsto 0 \quad \text{and} \quad g_2 : x \mapsto cx \,. \tag{6.7}$$

Our goal is to define a quadratic transition function h over $[-\epsilon, +\epsilon]$ that interpolates g_1 and g_2 and their first derivatives such that

$$\begin{array}{rcl} g_1(-\epsilon) & = & h(-\epsilon)\,, \\ g_1'(-\epsilon) & = & h'(-\epsilon)\,, \\ g_2(\epsilon) & = & h(\epsilon)\,, \\ g_2'(\epsilon) & = & h'(\epsilon)\,. \end{array} \tag{6.8}$$

This function is defined by

$$h : x \mapsto \frac{c}{4\epsilon}x^2 + \frac{c}{2}x + \frac{c\epsilon}{4}, \quad x \in [-\epsilon, \epsilon]. \tag{6.9}$$

This result can be applied with little modification to the slightly more general case where g_1 and g_2 are defined by

$$g_1 : x \mapsto a \quad \text{and} \quad g_2 : x \mapsto cx + d\,, \tag{6.10}$$

with a, b and $c \neq 0$ given constant parameters.

Letting $x_I = \frac{a-d}{c}$ denote the abscissa of the intersection between the graphs of g_1 and g_2, we can define the transition function h over $[x_I - \epsilon, x_I + \epsilon]$ which interpolates g_1 and g_2 at $\{x_I - \epsilon\}$ and $\{x_I + \epsilon\}$ by

$$h : x \mapsto \frac{c}{4\epsilon}t^2 + \frac{c}{2}t + \frac{c\epsilon}{4} + a,\ t = x - x_I,\ x \in [x_I - \epsilon, x_I + \epsilon]. \tag{6.11}$$

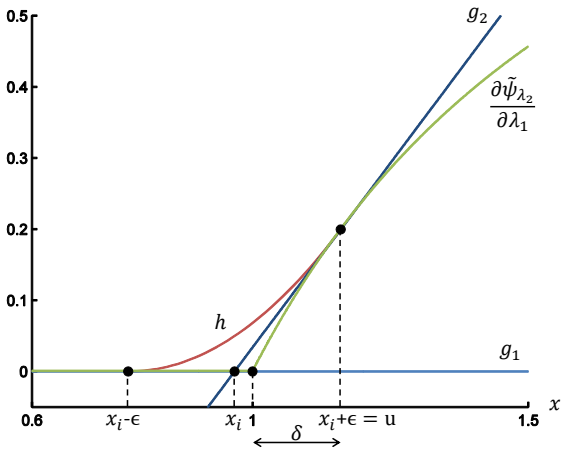


Figure 6.9: *Illustration for the construction of a transition function h smoothing the first derivative of the univariate function* $\tilde{\psi}_{\lambda_2}$.

We can now focus on our original problem, i.e. the smoothing of the derivatives of the relaxed energy $\tilde{\psi}$. The discontinuities in its second derivatives appears between $\frac{\partial\tilde{\psi}_{\text{sl}}}{\partial\lambda_1}$ and $\frac{\partial\tilde{\psi}_{\text{tf}}}{\partial\lambda_1}$, $\frac{\partial\tilde{\psi}_{\text{tf}}}{\partial\lambda_1}$ and $\frac{\partial\tilde{\psi}_{\text{fm}}}{\partial\lambda_1}$, and $\frac{\partial\tilde{\psi}_{\text{tf}}}{\partial\lambda_2}$ and $\frac{\partial\tilde{\psi}_{\text{fm}}}{\partial\lambda_2}$. In what follows, we detail the construction of the transition function between $\frac{\partial\tilde{\psi}_{\text{sl}}}{\partial\lambda_1}$ and $\frac{\partial\tilde{\psi}_{\text{tf}}}{\partial\lambda_1}$. The approach can be directly adapted to smooth out the two other discontinuity lines.

For every λ_2, we keep λ_2 fixed and smooth the first derivative of the univariate function $\tilde{\psi}_{\lambda_2} : \lambda_1 \mapsto \tilde{\psi}(\lambda_1, \lambda_2)$. In order to make the size of the transition interval converge to 0 as λ_1 and λ_2 converge to $(1,1)$, we first define its upperbound u by

$$u = 1 + \delta(1 - \lambda_2) \,, \tag{6.12}$$

where δ is a given parameter.

We then define the function interpolating $\tilde{\psi}_{\lambda_2}$ using the previous result (equation (6.11)) with

$$\begin{aligned} a &= \frac{\partial\tilde{\psi}_{\text{sl}}}{\partial\lambda_1}(1,\lambda_2) \,, \\ c &= \frac{\partial^2\tilde{\psi}_{\text{sl}}}{\partial\lambda_1^2}(u,\lambda_2) \,, \\ \epsilon &= \frac{1}{c}\frac{\partial\tilde{\psi}_{\text{tf}}}{\partial\lambda_1}(u,\lambda_2) \,, \\ x_I &= u - \epsilon \,, \end{aligned} \tag{6.13}$$

as illustrated on Figure 6.9.

6.5.3 Discretization

We proceed to the discretization of the relaxed energy function and its derivatives using Constant Strain Triangles as described in Section 3.1.4.

We let m denote the number of external and internal panels of the inflatable structure. The geometry of each panel in its undeformed state is described by a triangle mesh $\mathcal{P}_i$. The deformed mesh, comprising all panels, is denoted by $\mathcal{M}$. Furthermore, we let $\mathbf{x} \in \mathbb{R}^{3n}$ denote the vector of deformed positions for the n nodes of the surface. Likewise, $\bar{\mathbf{x}} \in \mathbb{R}^{2N}$ holds the positions of the N undeformed panel vertices in their two-dimensional domain. Note that $N > n$ since, for each deformed vertex on a panel boundary in $\mathcal{M}$, there are at least two corresponding undeformed vertices from boundaries of different panels.

The internal forces $\mathbf{f}_i^{\text{int}}$ at each node are obtained as

$$\mathbf{f}_i^{\text{int}} = -\sum_{e \in \mathcal{F}_i} \frac{\partial \tilde{\psi}^e}{\partial \mathbf{x}_i} V_e = -\sum_{e \in \mathcal{F}_i} \left(\frac{\partial \tilde{\psi}^e}{\partial \lambda_1^e} \frac{\partial \lambda_i^e}{\partial \mathbf{x}_i} + \frac{\partial \tilde{\psi}^e}{\partial \lambda_2^e} \frac{\partial \lambda_2^e}{\partial \mathbf{x}_i} \right) h A_e$$

where $\mathcal{F}_i$ denotes the set of triangle elements incident to the vertex i, A_e is the initial area of element e, h is the thickness of the panels and $V_e = hA_e$ is the volume of e. It is evident from this expression that the gradient and Hessian of (6.4) require the first and second derivatives of the principal stretches. We provide the corresponding derivations in Appendix B.

We assume that the inflatable is made of a light-weight material and that the pressure inside the structure is much higher than external pressure. Under these assumptions, we can safely ignore effects due to self weight and only consider pressure forces, which we define directly in the discrete setting as

$$\mathbf{f}_i^{\text{p}} = p \frac{\partial V}{\partial \mathbf{x}_i} = \sum_{e \in \mathcal{F}_i} \frac{1}{3} p A_e \mathbf{n}_e \,, \tag{6.14}$$

where p is the pressure value and $\mathbf{n}_e$ and A_e are the outward normal and the area of element e. The shape of the inflated structure can then be computed by solving the static equilibrium problem

$$\mathbf{f}_i^{\text{int}}(\mathbf{x}, \bar{\mathbf{x}}) + \mathbf{f}_i^{\text{p}}(\mathbf{x}) = 0, \; 1 \leq i \leq n \,. \tag{6.15}$$

6.6 Automatic Pattern Generation

Our system combines user-guided seam design with automatic pattern generation. During seam design, the user will repeatedly invoke the pattern optimization

scheme in order to explore the effect of a given layout or edit on the inflated structure.

Formally, given a target mesh $\mathcal{T}$ and a set of seam lines segmenting the mesh into m parts, we seek to find optimal panel shapes $\bar{\mathbf{x}}$ for each part such that the distance between the inflated mesh $\mathcal{M}$ and the target mesh $\mathcal{T}$ is as small as possible. We cast this goal into the form of a constrained minimization problem,

$$\min_{\mathbf{x},\bar{\mathbf{x}}} E(\mathbf{x},\bar{\mathbf{x}}) \quad \text{s.t. } \mathbf{f}(\mathbf{x},\bar{\mathbf{x}}) = \mathbf{0}\,, \tag{6.16}$$

where the constraints $\mathbf{f}(\mathbf{x},\bar{\mathbf{x}}) = \mathbf{0}$ require force equilibrium in every node and $E(\mathbf{x},\bar{\mathbf{x}})$ summarizes various objective terms, which we detail in the next section. We solve this optimization problem using the Sequential Quadratic Programming (SQP) method presented by Byrd et al. [2010] and summarized in Chapter 3, that guarantees progress even in non-convex regions.

6.6.1 Objectives

Distance to Target

In order to quantify the distance between the inflated mesh $\mathcal{M}$ and the target mesh $\mathcal{T}$, we construct a distance field on $\mathcal{T}$ using implicit moving least squares [Öztireli et al., 2009]. The distance penalty is defined as

$$E_{\text{target}}(\mathbf{x}) = \sum_i \frac{\sum_k \mathbf{n}_k \cdot (\mathbf{x}_i - \mathbf{c}_k)\phi_k(\mathbf{x})}{\sum_k \phi_k(\mathbf{x})}\,, \tag{6.17}$$

where $\phi_k(\mathbf{x}) = \left(1 - \frac{||\mathbf{x}-\mathbf{x}_k||_2^2}{h^2}\right)^4$ are locally-supported kernel functions that vanish beyond their support radius h, that we set to twice the average length of the target mesh edges, while $\mathbf{c}_k$ and $\mathbf{n}_k$ denote the vertex positions and normals of $\mathcal{T}$, respectively. This measure allows the vertices of $\mathcal{M}$ to slide freely over $\mathcal{T}$, whereas a simpler pair-wise vertex distance would lead to bias and thus unnecessarily restrict approximation quality.

Seam Locations

Seams are critical to the aesthetics of inflatable structures. Our interface provides tools that allow the user to rapidly create seam layouts on the target surface. Some of these seams simply split larger areas into smaller parts, in which case their exact location is not critical. Our optimization can leverage such freedom for better shape approximation. Others, however, serve an important aesthetic role that the

final inflatable structure has to respect by adhering to the shape and location of the seams. We therefore let the user specify the importance of a given seam $\mathcal{S}_i$ by assigning a weight σ_i to it that determines how strongly the corresponding seam vertices on $\mathcal{M}$ are attracted to their target locations on $\mathcal{S}_i$. We define the penalty function based on vertex-wise L_2-distance as

$$E^i_{\text{seam}}(\mathbf{x}) = \sigma_i \sum_{j \in \mathcal{S}_j} ||\mathbf{x}_j - \mathbf{s}_j||_2^2 \,, \tag{6.18}$$

where $\mathbf{s}_j$ denotes the target position for $\mathbf{x}_j$ on the seam. It is worth noting that the seam vertices are not restricted to be a subset of the vertices of $\mathcal{M}$—seams can run freely across the target surface.

Fabrication Constraints

In order for two panels $\mathcal{P}_j$ and $\mathcal{P}_k$ to join in a seam, the corresponding boundary segments must have the same length on both panels. Otherwise, discrepancies have to be corrected *a posteriori* using cuts or pleats, which increases fabrication time and degrades the visual quality of the product. We enforce this equal-length requirement per seam as

$$E_{\text{length}}(\bar{\mathbf{x}}) = \sum_{i=1}^{s_e} (L^i_j(\bar{\mathbf{x}}) - L^i_k(\bar{\mathbf{x}}))^2 \,, \tag{6.19}$$

where s_e is the number of seam edges in $\mathcal{M}$ and L^i_j and L^i_k are the lengths of corresponding edge vectors on the boundaries of $\mathcal{P}_j$ and $\mathcal{P}_k$. While the lengths of the boundary segments have to be the same, they can, and generally will, exhibit different curvatures. Nevertheless, boundaries should still remain smooth, which we encourage with a corresponding penalty term,

$$E_{\text{smooth}}(\bar{\mathbf{x}}) = \sum_{i=1}^{m} \sum_{j=1}^{b_i} ||\bar{\mathbf{q}}^i_l - 2\bar{\mathbf{q}}^i_j + \bar{\mathbf{q}}^i_r||_2^2 \,, \tag{6.20}$$

where b_i is the number of non-corner boundary vertices $\bar{\mathbf{q}}^i$ of panel $\mathcal{P}_i$, whereas $\bar{\mathbf{q}}^i_l$ and $\bar{\mathbf{q}}^i_r$ denote the left and right neighbors of boundary vertex $\bar{\mathbf{q}}^i_j$.

Regularization

The distance energy (6.17) allows the vertices $\mathbf{x}$ to slide on the target surface, but this freedom comes at the price of a nullspace: for any displacement of a given internal panel vertex $\bar{\mathbf{x}}_i$, there is a corresponding world-space displacement $\mathbf{x}_i$ such

that neither objectives nor constraints change. In order to obtain a well-posed problem, we use a Laplacian regularizer that asks for a smooth distribution of internal panel vertices as

$$E_{\text{laplace}}(\bar{\mathbf{x}}) = \sum_{i=1}^{m} \sum_{j=1}^{n_i} \mathcal{L}(\bar{\mathbf{x}}_j) \,, \tag{6.21}$$

where $\mathcal{L}(\bar{\mathbf{x}}_j)$ is the Tutte Laplacian [Tutte, 1963]. As a desirable side effect, this regularizer also promotes well-shaped elements.

6.6.2 Initial Flattening

The final shape of the patterns is obtained by solving the constrained minimization problem (6.16). However, a good initial guess is crucial for rapid convergence. In principle, any mesh parametrization method can be used to create an initial guess. A particular aspect of our setting is, however, that the shape of the panels is entirely defined by their boundary vertices—the shape of interior elements is, to a large extent, an afterthought. Among the many existing methods, we therefore took inspiration in one that preserves the lengths of the segment boundaries [Wang, 2008].

We start by converting the user-provided seams, represented by smooth spline curves, into sets of edge vectors, defining a segmentation of the target mesh $\mathcal{T}$. For each segment, we first flatten its boundary $\mathbf{q}^i$ by minimizing an objective function that penalizes squared differences in edge lengths and internal angles as

$$E_{\text{fl}}(\bar{\mathbf{q}}^i) = \sum_{j=1}^{b_i} \tfrac{1}{l_{ij}} (\bar{l}_{ij}(\bar{\mathbf{q}}^i) - l_{ij})^2 + l_j (\bar{\theta}_j(\bar{\mathbf{q}}^i) - \theta_j)^2 \,, \tag{6.22}$$

where $\bar{l}_{ij}$ are the lengths of the boundary edges of panel $\mathcal{P}_i$ and l_{ij} are the corresponding lengths in $\mathcal{T}$. Moreover, $\bar{\theta}_j$ is the sum of internal angles around $\bar{\mathbf{q}}^i_j$, θ_j is the corresponding quantity on $\mathcal{T}$, and l_j is the average length of the two edges incident to $\mathbf{q}^i_j$. The resulting nonlinear problem is solved with a few iterations of Newton's method. Keeping the boundary vertices fixed, the positions of the internal vertices are then computed by minimizing a Laplacian energy analogous to (6.21), which amounts to a single linear solve.

We note that, although a good initial guess helps speed convergence, our optimization scheme is not very sensitive to this choice. The example shown in Figure 6.10 puts this robustness to a test, using *D-Charts* and *ABF++* (see [Julius et al., 2005]) as initial guess for the three panels of a spherical balloon. Here, the lengths of the panel boundaries are very different from the corresponding lengths on the target

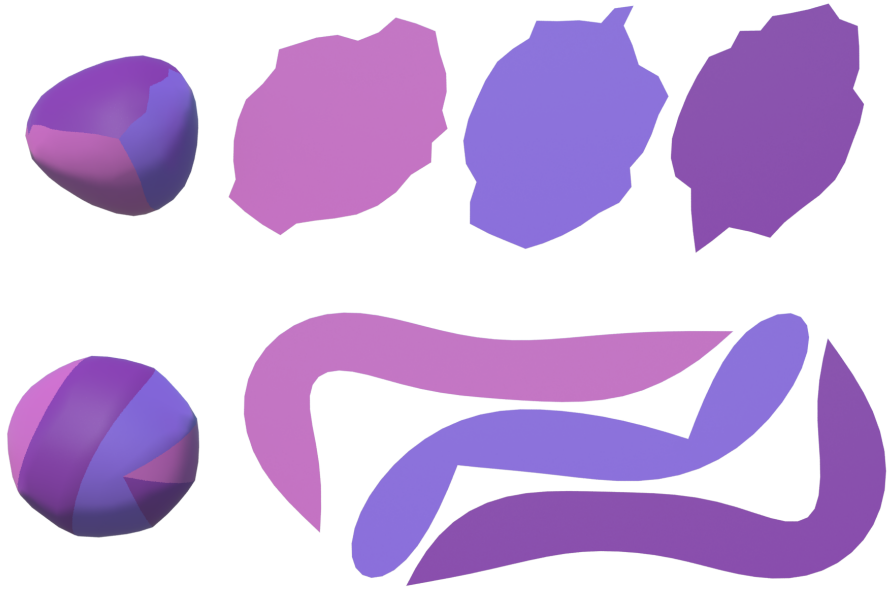


Figure 6.10: *Pattern optimization for a spherical balloon. The balloon (left) and its patterns (right) are shown for the initial guess (top), and after optimization (bottom).*

shape. Nevertheless, our method is able to find panel shapes that allow for a close approximation of the target. It is also worth noting that, since the seams were not restricted to stay in place, the final patterns are very different from the initial guess, revealing a surprisingly symmetric and elegant solution.

6.6.3 Remeshing

Figure 6.10 exemplifies the potential difference between initial and final patterns. In order to robustly handle such extreme changes in size and shape, we integrate the optimization with a remeshing method that maintains well-shaped elements at all times. A number of works in graphics have explored the integration of remeshing and simulation, either globally [O'Brien and Hodgins, 1999; Bargteil et al., 2007; Wojtan and Turk, 2008] or in a locally adaptive manner [Wicke et al., 2010; Narain et al., 2012]. Since our mesh sizes are comparatively small, we opt for a global remeshing scheme that builds on *Triangle* [Shewchuk, 1996]. Remeshing is invoked whenever the aspect ratio of an element falls below a given threshold. We first resample the boundaries of the patches to satisfy a minimum- and maximum-length criteria on the edges, maintaining correspondence between adjacent patches. We then invoke *Triangle* to remesh the interior of each patch and carbon-copy all changes to the inflated mesh as well as the target mesh.

6.7 Results

We used our design system to create a diverse set of inflatable structures, seven of which we present and discuss in this section. Information regarding the number of elements and the number of panels of these models is provided in Table 6.2. For validation, we also created physical prototypes for three of these examples. The design interface and the results are demonstrated in Figures 6.12 and 6.13. The fabricated models are made out of PVC plastic sheets. The optimized patches generated by our system were cut with a computer-controlled cutting machine and then manually hot sealed. In the following section, we validate some of our design decisions and discuss our results in more detail.

Simulation

We compared our relaxed energy density based on tension field theory to a full simulation using the strain energy density of an incompressible Neo-Hookean material as stated in (6.3). The full simulation for the Teddy model shown in Figure 6.11 with ≈59k elements took more than two hours, whereas our relaxed energy density is computationally more efficient because it requires fewer elements and avoids problems due to indefiniteness. We tested our simulation approach with several different resolutions, ranging from ≈3k elements to ≈59k elements, and report the approximation error in Figure 6.11. The computation for the three meshes of ≈3k, ≈15k and ≈59k elements took 4.5, 77 and 157 seconds, respectively. In practice, we observed that already with a coarse mesh we are able to obtain satisfactory accuracy and therefore choose for all our models a low mesh resolution while still resembling the shape of the original object.

Performance of Optimization

To evaluate the performance of our optimization, we start from a given seam layout and measure the required time for convergence when no later edit is performed. This comprises the initial flattening, initialization of all data structures, remeshing operations and subsequent data structure updates, as well as optimization. The optimization is the most expensive step of the process but the evaluation of the different involved quantities largely dominates the linear solves.. All computations were done on a standard desktop computer with 3.20 GHz and 12 cores. Our research prototype is written in C++ and, except of solving the system of equations, not yet parallelized, leaving room for significant speed-ups. As a stopping criterion, we require the infinity-norm of the gradient of the Lagrangian corresponding to (6.16) to be smaller than $\text{tol} = 10^{-3}$, whereas a tighter tolerance of $\text{tol} = 10^{-5}$

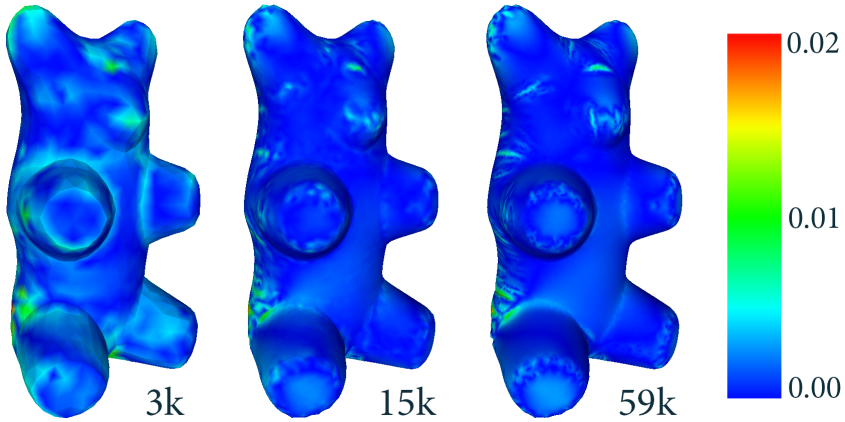


Figure 6.11: *Comparison of simulations with relaxed energy model on meshes with resolutions of 3k, 15k, and 59k elements to a reference simulation using the original energy on the 59k mesh. Colors indicate per-vertex difference for a unit size model.*

Model	#Remeshing steps	#Iterations / Time[s] preview		#Iterations / Time[s] full convergence	
Sphere	5	8	4	128	21
Teddy	2	23	17	193	258
Fox	4	11	35	398	420
Twisty	2	15	33	247	430
Flower (bottom)	3	51	60	298	546
Flower (top)	3	47	69	197	506
Tentacle	2	9	23	175	131
Elephant	3	30	137	401	1768

Table 6.1: *Processing time for our results. Timings are given for preview quality (relative error between displayed mesh and final result $< 5\%$ of the size of the model) and full convergence (gradient of objective $< 1.e^{-3}$ and forces $< 1.e^{-5}$)*

is used for the forces. We use the same thresholds for all models presented in this chapter. Table 6.1 lists detailed performance numbers of our algorithm for all examples, the number of required remeshing steps, and the number of SQP iterations. In practice, we observed that already after a few iterations the result is close to the final solution. We therefore opted to visualize the incremental steps of the optimization to the user, providing fast visual feedback and intuition about the quality of the seam placement. As shown in Table 6.1, after less than a minute, the relative error measured between the starting point of the optimization and the final converged result is below 5%, and therefore provides sufficient accuracy for a pre-visualization.

Seam Placement

The approximation quality, i.e., how well a given inflatable structure matches its target shape, depends on the number of patches as well as the seam layout. In our experiments we observed that there is a tradeoff between shape approximation and aesthetic requirements. As illustrated in Figure 6.10, seams can slide significantly during the optimization on the target surface. We allow the user to intuitively control the admissible amount of sliding by adjusting the weight of the corresponding penalty term (6.18). If seams are assigned a small weight, they can move such as to optimize the overall shape approximation. If seams are assigned a high weight, they stay close to their target location in 3D. As the location of the seams is generally very important for the aesthetics of the inflatable structure, we used relatively high weights for all models, except for the "Sphere". This approach proved particularly important and effective for modeling the "Fox" (Figure 6.12), for which characteristic features such as the eyes, and eyelids were delineated by corresponding seams.

Our interface allows non-expert users to efficiently add, edit, and replace seams and explore the impact of these operations on the inflated shape in an interactive manner. On average, designing a foil balloon took between 8 minutes for simple models ("Teddy") and less than half an hour for sophisticated models with internal connections ("Fox").

Internal Connections

Several of our models ("Fox", "Elephant", "Flower", "Twisty") rely on internal connections, which are created by connecting existing seams on the surface as indicated by the user. As shown in Figure 6.2, internal connections can be used, for example, to create sharp concave creases—a salient feature for many models. Figures 6.12 and 6.13 visualize the internal patches generated by our method and demonstrate that the resulting inflated shapes are in very good agreement with the desired behavior.

6.8 Discussion and Outlook

In this chapter, we presented a design system for creating inflatable structures made from flat panels. The enabling technology of our system is an automatic physics-based pattern generation method, combining fast simulation based on tension field theory and robust constraint optimization. Bound to an intuitive user interface, even non-expert users are able to design and explore intricate structures by simply

drawing and editing seams on an input model. As demonstrated by our results, our system also supports internal connections, thus significantly broadening the range of shapes that can be designed.

The shape of the flat panels is automatically computed by the system. However, input meshes that flatten to exceedingly thin panels pose challenges for subsequent remeshing and optimization. It would be helpful to automatically update the segments' connectivity during the optimization and merge thin panels to adjacent panels automatically. Also, we make no attempt at inferring the location of internal connections in an automatic way nor automatically consider geometric properties of our input model such as ridges, symmetries or curvature. Future work could include higher-level tools that exploit this information for supporting the seam placement or for auto-completion of partially drawn seams.

Finally, even with internal connections, there is a limit on what kind of shapes can be obtained with an inflatable structure. For example, planar regions and sharp convex edges (as shown in the "Twisty" example, Figure 6.13) are inherently difficult to reproduce. For future work, it would be interesting to indicate at the beginning of the design process infeasible regions and limits on achievable approximations.

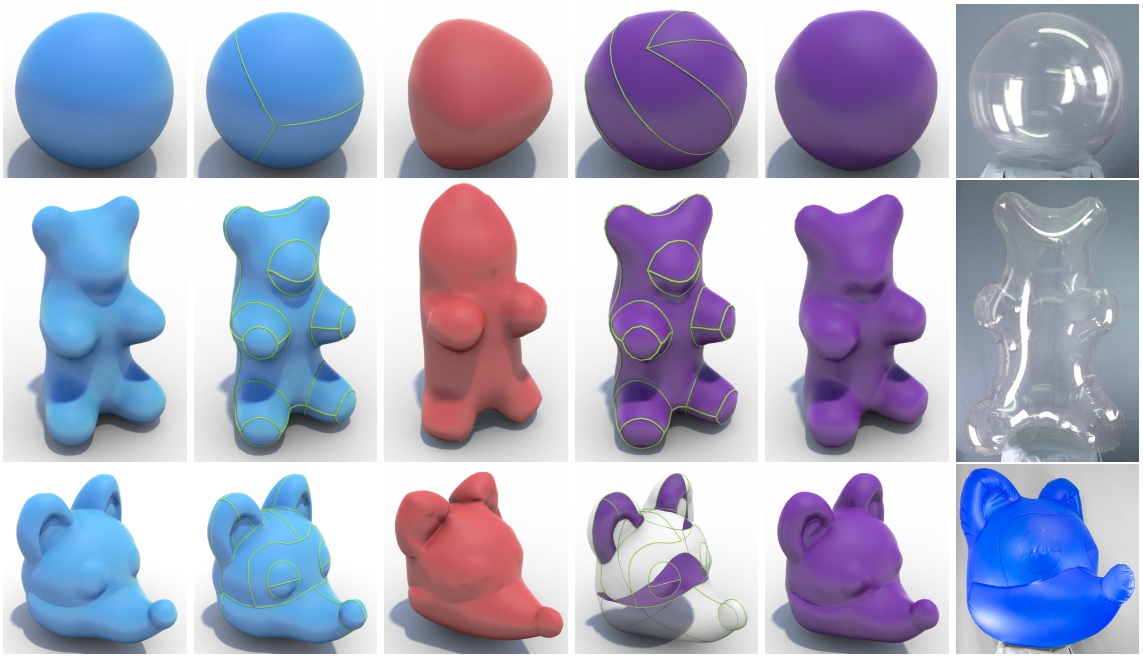


Figure 6.12: *Overview of our results (from left to right): input model, input model with seam layout, nonoptimized inflated shape, optimized result with seams, optimized result, fabricated prototype. The rows show (from top to bottom) the Sphere, Teddy, and Fox examples.*

Model	#Elements	#Total Panels	#Internal Panels
Sphere	692	3	0
Teddy	2974	17	0
Fox	7544	22	3
Twisty	6084	34	3
Flower (bottom)	6069	12	1
Flower (top)	7888	23	0
Tentacle	4458	21	0
Elephant	14436	40	2

Table 6.2: *Model Statistics.*

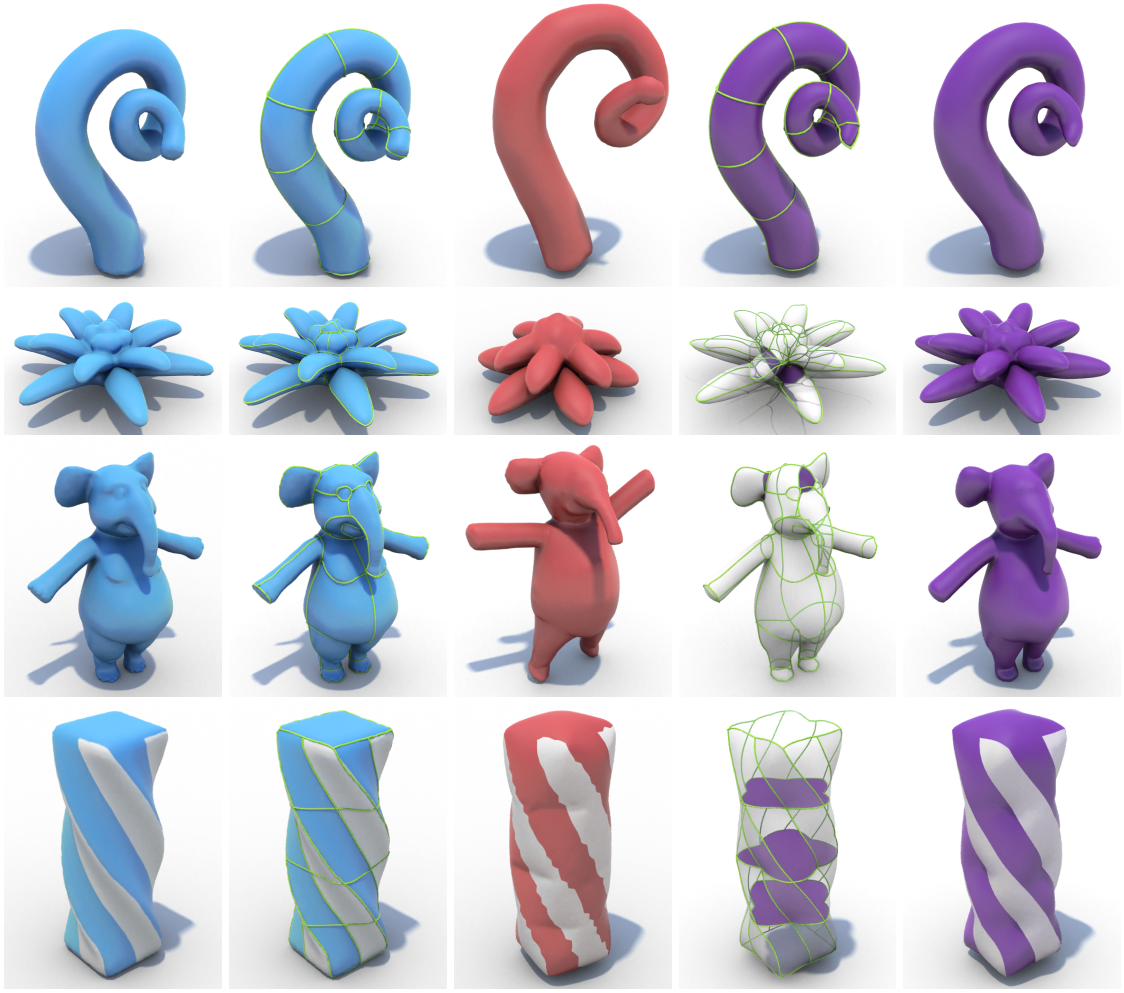


Figure 6.13: *Overview of results (from left to right): input model, input model with seam layout, nonoptimized inflated shape, optimized result with seams, optimized result without seams. The rows show (from top to bottom) the Tentacle, Flower, Elephant, and Twisty examples.*

CHAPTER 7

Conclusion

This chapter concludes the thesis by summarizing and discussing its principal contributions and by suggesting directions for potential future work.

7.1 Discussion

This thesis addressed a general problem which can be succinctly formulated: if we want to fabricate an object whose behaviour can be described by a set of deformed poses, what are the best values for the parameters that we are free to modify to best approximate the target shapes? Although the question is concise, the solution is not. It is hard, not to say impossible to come up with a black box which simply takes the inputs and automatically computes the outputs without specialized implementations. What degrees of freedom should we actually consider? How should we parametrize the design space? How to best measure optimality? What optimization scheme should we use? All these aspects are closely related to the specific instance of the problem which is considered and affect the capabilities of the final design system.

In this context, contributions of this work are two-fold. First, we demonstrated that for the three applications that we investigated, the design space in which we worked, as well as the proposed design workflow, allow sufficient freedom to fabricate objects of a large variety of shapes with limited user input. Second, we presented effective solutions to simulate and optimize the associated physical systems.

More specifically, in the context of custom rubber balloon design, we showed that a Hart-Smith material model with experimentally fitted parameters accurately captured the behaviour of silicone and latex; we demonstrated that complex inflated balloon shapes could be matched by automatically optimizing the balloons' rest shapes and that the resulting meshes could be used to 3D-print balloon positive molds for actual balloon fabrication.

We then presented a comprehensive process to create actuated deformable characters actuated by pins, strings or posed by hand. Our approach, centered around a three-stage optimization component, allows automatic computation of the actuation system and the material distribution of a figure expected to physically replicate some target animation. We first estimate the number of needed strings or pins by considering a dense set of actuators on the boundary of the model and discarding unnecessary ones using sparse regularization. We then refine the locations of the resulting actuators by letting them slide on the surface which is defined by an implicit distance field. We complete our approach by eventually computing rigid structures to embed in the object in order to further improve the final matching quality. This last problem is addressed from a material distribution perspective but instead of solving the discrete problem directly, we opted for a relaxed formulation which allows us to use continuous optimization techniques.

Finally, in the last part of the thesis, we focused on the design of inflatable structures made of flat panels and introduced an accurate coarse-scale simulator for inflatable membranes using tension field theory. We combined it with a physics-based pattern optimizer with integrated remeshing to compute optimally-shaped panels, such that the inflated structure matches a provided target shape with panels boundaries being at manually sketched locations. By letting the user interactively update the desired seam positions on the target shape, our tool warrant easy exploration of different seam layouts with limited effort and expertise.

This last application allowed us to tackle the tricky question of the amount a control required by a design tool. In the case of panel-based inflatable structures, seam placement is crucial both for aesthetics, and for structural design, it was therefore a strong candidate for an iterative interactive approach. However, properly balancing between user input and automation is generally delicate. The solution is not unique and depends on the familiarity of the targeted users with the problem. While computer-aided systems are unanimously welcomed, experts tend to expect fine tuning tools whereas novice users typically prefer high level control or fully automated systems.

7.2 Future Directions

This thesis proposed novel tools and algorithms to design and fabricate diverse types of custom objects. While we looked at the design problem from various angles, several aspects were not covered in this work. Also, many applications, with their associated challenges, remain unexplored, offering many exciting avenues for future research.

The design framework that we propose is based on concepts stemming from various disciplines, belonging themselves to active research fields. Any progress in related areas would automatically benefit to our system. In particular, many of the optimization problems that we derived involve non linear objectives with non linear constraints, most of them being non convex. Exploration of novel optimization techniques and alternative formulations would be consequently worthwhile. Likewise, our physics based optimization component relies on accurate simulation of the behaviour of the object to design. Although the approach that we take, based on standard finite element techniques, performed effectively, it would be valuable to analyse the behaviour of the physics-based optimization unit when novel simulation methods such as recent point-based schemes [Müller et al., 2004; Martin et al., 2010] are employed.

Combinatorial problems, known to be NP hard, naturally arise in many applications but techniques to approach specific instances exist. In this thesis, we looked into some problems of this class and proposed novel formulations allowing to use efficient continuous optimization methods. For example, in Chapter 5, we solved for a discrete material distribution problem by turning towards a relaxation technique. Similarly, we optimized for actuators locations by incorporating sparse regularization to our objective function. However, in many cases, combinatorial problems still remain challenging to solve. In Chapter 6, we increased the number of feasible inflatables' designs by allowing for internal connections between existing panels. Automatic creation and placement of such structures is not trivial and we let the user solve the combinatorial task of selecting the pieces to be connected. In general, because of their intrinsic difficulties, many combinatorial problems remain open and would deserve proper investigation.

In this thesis, we focused on the design of deformable solids and inflatable shells but the framework that we presented could be extended in many ways. First, control of other physical systems could be investigated. Exploration of cloth design, in particular, would be an exiting direction for future work. Handling of additional types of materials, such as anisotropic fabrics, would be thus very useful. In the context of solids' design, heterogeneous structures were investigated in Chapter 5, but our study was limited to two-material objects. Generalizing the method to an

arbitrary number of materials, or translating the approach to a micro-structure level would enlarge the range of emulated material behaviours and certainly increase the number of design possibilities. The effects of additional types of forces could also be investigated. In this thesis, we studied the actions of — possibly moving — fixed attachment points, but torque was not considered. Impulses, friction, contact forces due to collisions in general, were also discarded in this work. Mordatch et al. [2012] lately introduced a method to synthesize complex human behaviour involving manipulation of the environment. By introducing auxiliary decision variables, they simultaneously optimize for contacts and character motions. Their contact-invariant optimization scheme is an interesting option to cope with the challenges related to collision handling. Further exploration of such approaches would be helpful for the design of mutually interacting deformable systems. Finally, effects of external forces could be incorporated to the optimization objective function as desirable properties of the object to manufacture: the user could specify whether an inflatable should float, fly, support a certain weight, or if a character should walk, jump, dance, etc. Taking such general characteristics into account will most likely increase the dimensionality of the design solution. How to best analyse and explore the resulting Pareto front are interesting related research questions.

Designing dynamic systems remains a challenging task. We broached this topic in Chapter 5 by extracting key frames from an input animation and by optimizing the system variables for all the target deformed poses simultaneously, assuming quasi-static deformations. However, while this appeared to be sufficient for animations with limited secondary motion, physical replication of highly dynamic sequences would most likely require a better suited approach. Simulation of such systems typically implies solving an initial value problem, arising from the spacial discretization of the equations of motion in time. Moreover, the level of accuracy required in the context of fabrication-oriented design means using an implicit integration scheme, and thus involves nonlinearly coupled variables. Although it would be possible to adapt the feasibility conditions of our physics-based optimizer and take into account velocity unknowns for all the time steps, the size of the system to solve would quickly become prohibitive. A promising approach is to work in a lower dimensional space to decrease the problem complexity. While such a methodology has successfully been applied to the control of key-frame based animations [Barbič et al., 2009], finding the most adapted reduced space for physical replication of dynamic systems is still an open — but very exciting — problem.

The tools we developed in this work require very little knowledge and expertise in shape modeling and can be used by a large range of users. By relying on data-driven approaches, we reduced the design process to the import of geometry and to the specification of high level features. However, any target mesh is not necessarily a feasible shape from the physics perspective. Design optimality in

such case is not a well defined concept. Analysis and correction, when necessary, of the provided inputs, automatically or using physics-aware editing tools, are complementary and still challenging tasks, conducive to future research.

APPENDIX A

Energy Functions for Material Models

This appendix lists the constitutive equations of the hyperelastic materials that we used in this work, either for modeling or comparison.

As in the main part of this thesis, we let $\mathbf{F}$ denote the deformation gradient, $\mathbf{C} = \mathbf{F}^t\mathbf{F}$, the right Cauchy Green tensor and I_1, I_2 and I_3, the three invariants of $\mathbf{C}$ defined as

$$\begin{array}{rcl} I_1 & = & \mathrm{tr}(\mathbf{C}) \;=\; \lambda_1^2 + \lambda_2^2 + \lambda_3^2 \,, \\ I_2 & = & \frac{1}{2}(\mathrm{tr}(\mathbf{C})^2 - \mathrm{tr}(\mathbf{C}^t\mathbf{C})) \;=\; \lambda_1^2\lambda_2^2 + \lambda_2^2\lambda_3^2 + \lambda_3^2\lambda_1^2 \,, \\ I_3 & = & \det(\mathbf{C}) \;=\; \lambda_1^2\lambda_2^2\lambda_3^2 \,, \end{array} \tag{A.1}$$

where λ_1, λ_2 and λ_3 are the eigenvalues of $\mathbf{F}$.

Many models use distinct terms to distinguish between the change in shape and the change in volume of a material's infinitesimal element. To this end, the material energy function is often described in terms of the new invariants

$$\begin{array}{l} J = \sqrt{I_3} \,, \\ \bar{I}_1 = J^{-\frac{2}{3}} I_1 \;\;, \\ \bar{I}_2 = J^{-\frac{4}{3}} I_2 \;\;. \end{array} \tag{A.2}$$

In what follows, we formulate the equations for the compressible versions of the material models. When the change in volume is very small, it is convenient to assume that the material is perfectly incompressible. This requirement can be directly integrated into the formulas below by setting J to 1. In this case, we also have $\bar{I}_1 = I_1$ and $\bar{I}_2 = I_2$.

Saint Venant-Kirchhoff

The energy density of a Saint Venant-Kirchhoff material, which extends linear elastic materials to nonlinear regimes, has the form

$$\Psi = \frac{\lambda}{2}[\mathrm{tr}(\mathbf{E})]^2 + \mu \mathrm{tr}(\mathbf{E}^2)\ ,\ \ \mathbf{E} = \frac{1}{2}(\mathbf{C} - \mathbf{I})\ , \tag{A.3}$$

where λ and μ are material constants, the so-called Lamé's parameters.

Neo-Hookean

When dealing with relatively large deformations, another popular choice is the Neo-Hookean material model, capable of reproducing nonlinear stress-strain behaviours. Its energy density is given by

$$\Psi = \frac{\mu}{2}(\bar{I}_1 - 3) + \frac{\kappa}{2}(J-1)^2\ , \tag{A.4}$$

where μ and κ are the material parameters.

Mooney-Rivlin

The Neo-Hookean model is actually a special case of the more general Mooney-Rivlin model, which uses the tree invariants $\bar{I}_1$, $\bar{I}_2$ and J as

$$\Psi = \frac{\mu_1}{2}(\bar{I}_1 - 3) + \frac{\mu_2}{2}(\bar{I}_2 - 3) + \frac{\kappa}{2}(J-1)^2\ , \tag{A.5}$$

with μ_1, μ_2 and κ, the material constants.

Ogden

For complex hyperelastic materials, the material behaviour can be accurately described using an Ogden model, that is defined in terms of the eigenvalues of $\mathbf{F}$ as

$$\Psi = \sum_{j=1}^{N} \frac{2\mu_j}{\alpha_j^2}(\bar{\lambda}_1^{\alpha_j} + \bar{\lambda}_2^{\alpha_j} + \bar{\lambda}_3^{\alpha_j} - 3) + \frac{\kappa}{2}(J-1)^2\ , \tag{A.6}$$

where $\bar{\lambda}_i = J^{-\frac{1}{3}}\lambda_i$, $1 \leq i \leq 3$, and μ_j, α_j and κ are material parameters.

Hart-Smith

For large strains, rubber materials typically exhibit an energy growth which is an exponential rather than a polynomial function of the invariants. The empirical model of Hart-Smith takes this behavior directly into account, describing an exponential strain energy function through its derivatives as

$$\frac{\partial W}{\partial I_1} = G \cdot e^{k_1 (I_1 - 3)^2} \,, \quad \frac{\partial W}{\partial I_2} = G \cdot \frac{k_2}{I_2} \,, \tag{A.7}$$

where k_1, k_2 and G are material constants.

APPENDIX B

Derivatives of Principal Stretches

In this appendix, we derive the expressions for the derivatives of the principles stretches with respect to the coordinates of the mesh vertices in the deformed configuration.

Again, we let $\mathbf{C}$ denote the right Cauchy-Green tensor $\mathbf{C} = \mathbf{F}^t\mathbf{F}$, where $\mathbf{F}$ is the deformation gradient. Here we restrict considerations to the two-dimensional setting, noting that an extension to three dimensions is straightforward through kinematic assumptions [1]. We express $\mathbf{C}$ in terms of principal stretches (eigenvalues) λ_i and corresponding principal directions (eigenvectors) $\mathbf{N}_i$ as

$$\mathbf{C} = \sum_i \lambda_i \mathbf{N}_i \mathbf{N}_i^t \,, \quad \text{where} \quad \lambda_i = \mathbf{N}_i^t \mathbf{C} \mathbf{N}_i \,. \tag{B.1}$$

Without loss of generality, we assume that $\mathbf{N}_1$ corresponds to the direction of maximum stretch λ_1. In our two-dimensional setting, the unit-length vector $\mathbf{N}_1$ can be parametrized by a single scalar α as

$$\mathbf{N}_1(\alpha) = (\cos(\alpha), \sin(\alpha))^t \tag{B.2}$$

The second eigenvector $\mathbf{N}_2$ of $\mathbf{C}$ is orthogonal to $\mathbf{N}_1$

$$\mathbf{N}_2(\alpha) = (-\sin(\alpha), \cos(\alpha))^t \,. \tag{B.3}$$

We note that

$$\mathbf{N}_2 = \frac{\partial \mathbf{N}_1}{\partial \alpha} \quad \text{and} \quad \mathbf{N}_1 = -\frac{\partial \mathbf{N}_2}{\partial \alpha} \,. \tag{B.4}$$

[1] the stretch in the thickness direction is inferred from in-plane stretches assuming constant volume, i.e., $\det(\mathbf{C}) = 1$.

For computing the derivatives of the principal stretches, we will focus on λ_1 for the sake of conciseness. Analogous expressions hold for λ_2. The first derivative of the maximum stretch is determined as

$$\begin{aligned}
\frac{\partial \lambda_1}{\partial \mathbf{x}} &= \frac{\partial}{\partial \mathbf{x}}(\mathbf{N}_1^t \mathbf{C} \mathbf{N}_1) && \text{(B.5)}\\
&= \frac{\partial \mathbf{N}_1^t}{\partial \mathbf{x}} \mathbf{C} \mathbf{N}_1 + \mathbf{N}_1^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1 + \mathbf{N}_1^t \mathbf{C} \frac{\partial \mathbf{N}_1}{\partial \mathbf{x}} && \text{(B.6)}\\
&= \left(\frac{\partial \mathbf{N}_1}{\partial \alpha} \frac{\partial \alpha}{\partial \mathbf{x}} \right)^t \mathbf{C} \mathbf{N}_1 + \mathbf{N}_1^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1 + \mathbf{N}_1^t \mathbf{C} \left(\frac{\partial \mathbf{N}_1}{\partial \alpha} \frac{\partial \alpha}{\partial \mathbf{x}} \right) && \text{(B.7)}\\
&= \frac{\partial \alpha}{\partial \mathbf{x}}^t \mathbf{N}_2^t \mathbf{C} \mathbf{N}_1 + \mathbf{N}_1^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1 + \mathbf{N}_1^t \mathbf{C} \mathbf{N}_2 \frac{\partial \alpha}{\partial \mathbf{x}} && \text{(B.8)}\\
&= \mathbf{N}_1^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1 \,, && \text{(B.9)}
\end{aligned}$$

where the last transformation follows from the fact that $\mathbf{C}\mathbf{N}_1 = \lambda_1 \mathbf{N}_1$ and $\mathbf{N}_2^t \mathbf{N}_1 = 0$. It is evident from these expressions that the only derivatives required are those of $\mathbf{C}$, which are easily computed[2].

The second derivative follows as

$$\begin{aligned}
\frac{\partial^2 \lambda_1}{\partial \mathbf{x}^2} &= \frac{\partial}{\partial \mathbf{x}}(\mathbf{N}_1^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1) && \text{(B.10)}\\
&= \frac{\partial \mathbf{N}_1^t}{\partial \mathbf{x}} \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1 + \mathbf{N}_1^t \frac{\partial^2 \mathbf{C}}{\partial \mathbf{x}^2} \mathbf{N}_1 + \mathbf{N}_1^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \frac{\partial \mathbf{N}_1^t}{\partial \mathbf{x}} && \text{(B.11)}\\
&= \left(\frac{\partial \mathbf{N}_1}{\partial \alpha} \frac{\partial \alpha}{\partial \mathbf{x}} \right)^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1 + \mathbf{N}_1^t \frac{\partial^2 \mathbf{C}}{\partial \mathbf{x}^2} \mathbf{N}_1 + \mathbf{N}_1^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \left(\frac{\partial \mathbf{N}_1}{\partial \alpha} \frac{\partial \alpha}{\partial \mathbf{x}} \right) && \text{(B.12)}\\
&= \frac{\partial \alpha}{\partial \mathbf{x}}^t \mathbf{N}_2 \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1 + \mathbf{N}_1^t \frac{\partial^2 \mathbf{C}}{\partial \mathbf{x}^2} \mathbf{N}_1 + \mathbf{N}_1^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_2 \frac{\partial \alpha}{\partial \mathbf{x}} \,. && \text{(B.13)}
\end{aligned}$$

The first and last term of (B.13) can be further simplified. Using the fact that $\mathbf{N}_1^t \mathbf{C} \mathbf{N}_2 = 0$ we have

$$\frac{\partial}{\partial \mathbf{x}}(\mathbf{N}_2^t \mathbf{C} \mathbf{N}_1) = \mathbf{N}_2^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1 + \frac{\partial \mathbf{N}_2}{\partial \mathbf{x}}^t \mathbf{C} \mathbf{N}_1 + \mathbf{N}_2^t \mathbf{C} \frac{\partial \mathbf{N}_1}{\partial \mathbf{x}} = 0 \,, \qquad \text{(B.14)}$$

and therefore

$$\begin{aligned}
\mathbf{N}_2^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}} \mathbf{N}_1 &= -\left(\frac{\partial \mathbf{N}_2}{\partial \alpha} \frac{\partial \alpha}{\partial \mathbf{x}} \right)^t \mathbf{C} \mathbf{N}_1 - \mathbf{N}_2^t \mathbf{C} \left(\frac{\partial \mathbf{N}_1}{\partial \alpha} \frac{\partial \alpha}{\partial \mathbf{x}} \right) && \text{(B.15)}\\
&= \frac{\partial \alpha}{\partial \mathbf{x}}^t \mathbf{N}_1 \mathbf{C} \mathbf{N}_1 - \mathbf{N}_2^t \mathbf{C} \mathbf{N}_2 \frac{\partial \alpha}{\partial \mathbf{x}} && \text{(B.16)}\\
&= (\lambda_1 - \lambda_2) \frac{\partial \alpha}{\partial \mathbf{x}}^t \,. && \text{(B.17)}
\end{aligned}$$

[2] Using Constant Strain Triangles for discretization, the deformation gradient $\mathbf{F}$ is a linear function of $\mathbf{x}$ such that $\mathbf{C}$ is quadratic in positions.

Using (B.17) in (B.13), we have

$$\frac{\partial^2 \lambda_1}{\partial \mathbf{x}^2} = 2(\lambda_1 - \lambda_2)\frac{\partial \alpha}{\partial \mathbf{x}}\frac{\partial \alpha}{\partial \mathbf{x}}^t + \mathbf{N}_1^t \frac{\partial^2 \mathbf{C}}{\partial \mathbf{x}^2}\mathbf{N}_1 \,. \tag{B.18}$$

Finally, by using (B.17) in (B.15) we obtain

$$\frac{\partial \alpha}{\partial \mathbf{x}}^t = \frac{1}{\lambda_1 - \lambda_2}\mathbf{N}_2^t \frac{\partial \mathbf{C}}{\partial \mathbf{x}}\mathbf{N}_1 \tag{B.19}$$

and can thus compute the second derivative of λ_1.

It should be noted that (B.17) and (B.19) are not well-defined when $\lambda_1 = \lambda_2$, in which case the eigendecomposition of $\mathbf{C}$ becomes non-unique anyways. For the application described in Chapter 6, however, this problem does not arise since the case of $\lambda_1 = \lambda_2$ is handled by the original energy density

$$\psi(\lambda_1, \lambda_2) = \kappa\left(\lambda_1 + \lambda_2 + \frac{1}{\lambda_1 \lambda_2} - 3\right) = \kappa\left(\mathrm{tr}(\mathbf{C}) + \frac{1}{\det \mathbf{C}} - 3\right) \,, \tag{B.20}$$

which does not require eigenvalues.

Bibliography

[Bächer et al., 2012] Moritz Bächer, Bernd Bickel, Doug L. James, and Hanspeter Pfister. Fabricating articulated characters from skinned meshes. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 31(4):47:1–47:9, 2012.

[Baginski et al., 2008] Frank Baginski, Michael Barg, and William Collier. Existence theorems for tendon-reinforced thin wrinkled membranes subjected to a hydrostatic pressure load. *Mathematics and Mechanics of Solids*, 13(6):532–570, 2008.

[Barbič et al., 2009] Jernej Barbič, Marco da Silva, and Jovan Popović. Deformable object animation using reduced optimal control. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 28(3):53:1–53:9, 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 Transactions on Graphics (Proceedings of SIGGRAPH)*, 26(3):16:1–16:8, 2007.

[Bathe, 1995] K.-J. Bathe. *Finite Element Procedures*. Prentice Hall, 1995.

[Becker and Teschner, 2007] M. Becker and M. Teschner. Robust and efficient estimation of elasticity parameters using the linear finite element method. In *SimVis*, pages 15–28, 2007.

[Bendsoe and Sigmund, 2004] Martin Philip Bendsoe and Ole Sigmund. *Topology Optimization*. Springer, 2004.

[Bergou et al., 2007] Miklós Bergou, Saurabh Mathur, Max Wardetzky, and Eitan Grinspun. Tracks: toward directable thin shells. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 26(3):50:1–50:10, 2007.

[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. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 28(3):89:1–89:9, 2009.

[Bickel et al., 2010] Bernd Bickel, Moritz Bächer, Miguel A. Otaduy, Hyunho Richard Lee, Hanspeter Pfister, Markus Gross, and Wojciech Matusik. Design and fabrication of materials with desired deformation behavior. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 29(4):63:1–63:10, 2010.

[Bickel et al., 2012] Bernd Bickel, Peter Kaufmann, Mélina Skouras, Bernhard Thomaszewski, Derek Bradley, Thabo Beeler, Phil Jackson, Steve Marschner, Wojciech Matusik, and Markus Gross. Physical face cloning. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 31(4):118:1–118:10, 2012.

[Bonet and Wood, 1997] J. Bonet and R. D. Wood. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press, 1997.

[Bonet et al., 2000] J. Bonet, R.D. Wood, J. Mahaney, and P. Heywood. Finite element analysis of air supported membrane structures. *Computer Methods in Applied Mechanics and Engineering*, 190(5):579–595, 2000.

[Bridson, 2008] Robert Bridson. *Fluid Simulation for Computer Graphics*. A K Peters, 2008.

[Byrd et al., 2010] Richard H. Byrd, Frank E. Curtis, and Jorge Nocedal. An inexact newton method for nonconvex equality constrained optimization. *Mathematical Programming*, 122(2):273–299, 2010.

[Calì et al., 2012] Jacques Calì, Dan A. Calian, Cristina Amati, Rebecca Kleinberger, Anthony Steed, Jan Kautz, and Tim Weyrich. 3d-printing of non-assembly, articulated models. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 31(6):130:1–130:8, 2012.

[Carlson et al., 2002] Mark Carlson, Peter J. Mucha, R. Brooks Van Horn, III, and Greg Turk. Melting and flowing. In *Proceedings of Symposium on Computer Animation*, pages 167–174, 2002.

[Carr et al., 2001] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of SIGGRAPH 01*, Annual Conference Series, pages 67–76, 2001.

[Ceylan et al., 2013] Duygu Ceylan, Wilmot Li, Niloy J Mitra, Maneesh Agrawala, and Mark Pauly. Designing and fabricating mechanical automata from mocap sequences. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 32(6):186:1–186:11, 2013.

[Chen et al., 2013] Desai Chen, David I. W. Levin, Piotr Didyk, Pitchaya Sitthi-Amorn, and Wojciech Matusik. Spec2fab: A reducer-tuner model for translating specifications to 3d prints. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):135:1–135:10, 2013.

[Choi and Ko, 2002] Kwang-Jin Choi and Hyeong-Seok Ko. Stable but responsive cloth. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 21(3):604–611, 2002.

[Choi and Ko, 2003] K.-J. Choi and H.-S. Ko. Extending the immediate buckling model to triangular meshes for simulating complex clothes. In *Eurographics 2003 Short Presentations*, pages 187–191, 2003.

[Cohen-Steiner et al., 2004] David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. Variational shape approximation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 23(3):905–914, 2004.

[Cohen, 1991] Laurent D. Cohen. On active contour models and balloons. *CVGIP: Image Understanding*, 53(2):211 – 218, 1991.

[Coros et al., 2010] Stelian Coros, Philippe Beaudoin, and Michiel van de Panne. Generalized biped walking control. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 29(4):130:1–130:9, 2010.

[Coros et al., 2012] Stelian Coros, Sebastian Martin, Bernhard Thomaszewski, Christian Schumacher, Robert Sumner, and Markus Gross. Deformable objects alive! *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 31(4):69:1–69:9, 2012.

[Coros et al., 2013] Stelian Coros, Bernhard Thomaszewski, Gioacchino Noris, Shinjiro Sueda, Moira Forberg, Robert W. Sumner, Wojciech Matusik, and Bernd Bickel. Computational design of mechanical characters. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):83:1–83:12, 2013.

[Derouet-Jourdan et al., 2010] Alexandre Derouet-Jourdan, Florence Bertails-Descoubes, and Joëlle Thollot. Stable inverse dynamic curves. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 29(6):137:1–137:10, 2010.

[Desbrun et al., 2002] Mathieu Desbrun, Mark Meyer, and Pierre Alliez. Intrinsic parameterizations of surface meshes. *Computer Graphics Forum*, 21(3):209–218, 2002.

[do Carmo, 1976] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976.

[Dong et al., 2010] Yue Dong, Jiaping Wang, Fabio Pellacini, Xin Tong, and Baining Guo. Fabricating spatially-varying subsurface scattering. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 29(4):62:1–62:10, 2010.

[Frey, 2004] William H. Frey. Modeling buckled developable surfaces by triangulation. *Computer-Aided Design*, 36(4):299 – 313, 2004.

[Furuta et al., 2010] Yohsuke Furuta, Nobuyuki Umetani, Jun Mitani, Takeo Igarashi, and Yukio Fukui. A film balloon design system integrated with shell element simulation. In *EUROGRAPHICS 2010 short paper*, 2010.

[Gibson and Mirtich, 1997] Sarah F. F. Gibson and Brian Mirtich. A survey of deformable modeling in computer graphics. Technical Report TR97-19, MERL - Mitsubishi Electric Research Laboratories, November 1997.

[Girard and Maciejewski, 1985] Michael Girard and A. A. Maciejewski. Computational modeling for the computer animation of legged figures. *SIGGRAPH Computer Graphics*, 19(3):263–270, 1985.

[Grinspun et al., 2003] Eitan Grinspun, Anil N. Hirani, Mathieu Desbrun, and Peter Schröder. Discrete shells. In *Proceedings of Symposium on Computer Animation*, pages 62–67, 2003.

[Hart et al., 2003] John C. Hart, Brent Baker, and Jeyprakash Michaelraj. Structural simulation of tree growth and response. *The Visual Computer*, 19(2-3):151–163, 2003.

[Hart-Smith, 1966] L. J. Hart-Smith. Elasticity parameters for finite deformations of rubber-like materials. *Zeitschrift für angewandte Mathematik und Physik*, 17, 1966.

[Hasan et al., 2010] Milos Hasan, Martin Fuchs, Wojciech Matusik, Hanspeter Pfister, and Szymon Rusinkiewicz. Physical reproduction of materials with specified subsurface scattering. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 29(4):61:1–61:10, 2010.

[Haslinger and Mäkinen, 2003] J. Haslinger and R. A. E. Mäkinen. *Introduction to Shape Optimization*. SIAM, 2003.

[Hormann et al., 2007] Kai Hormann, Bruno Lévy, and Alla Sheffer. *Mesh Parameterization: Theory and Practice*. SIGGRAPH 2007 Course Notes. ACM Press, August 2007.

[Hughes, 2000] Thomas J. R. Hughes. *The Finite Element Method. Linear Static and Dynamic Finite Element Analysis*. Dover Publications, 2000.

[Hullin et al., 2013] Matthias B. Hullin, Ivo Ihrke, Wolfgang Heidrich, Tim Weyrich, Gerwin Damberg, and Martin Fuchs. Computational fabrication and display of material appearance. In *Eurographics State-of-the-Art Reports (STAR)*, pages 137–153. Eurographics Association, 2013.

[Igarashi and Igarashi, 2008] Yuki Igarashi and Takeo Igarashi. Pillow: Interactive flattening of a 3d model for plush toy design. In *Smart Graphics*, volume 5166 of *Lecture Notes in Computer Science*, pages 1–7. Springer Berlin Heidelberg, 2008.

[Irving et al., 2004] G. Irving, J. Teran, and R. Fedkiw. Invertible finite elements for robust simulation of large deformation. In *Proceedings of Symposium on Computer Animation*, pages 131–140, 2004.

[Julius et al., 2005] Dan Julius, Vladislav Kraevoy, and Alla Sheffer. D-charts: Quasi-developablemesh segmentation. *Computer Graphics Forum*, 24(3):581–590, 2005.

[Kajberg and Lindkvist, 2004] J. Kajberg and G. Lindkvist. Characterisation of materials subjected to large strains by inverse modelling based on in-plane displacement fields. *International Journal of Solids and Structures*, 41(13):3439–3459, 2004.

[Kass et al., 1988] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[Kauer et al., 2001] M. Kauer, V. Vuskovic, J. Dual, G. Székely, and M. Bajka. Inverse Finite Element Characterization of Soft Tissues. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2001*, pages 128–136, 2001.

[Kilian et al., 2008] Martin Kilian, Simon Flöry, Zhonggui Chen, Niloy J. Mitra, Alla Sheffer, and Helmut Pottmann. Curved folding. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 27(3):75:1–75:9, 2008.

[Kondo et al., 2005] Ryo Kondo, Takashi Kanai, and Ken-ichi Anjyo. Directable animation of elastic objects. In *Proceedings of Symposium on Computer Animation*, pages 127–134, 2005.

[Landau et al., 1986] L.D. Landau, E.M. Lifshitz, A.M. Kosevitch, and L.P. Pitaevskiĭ. *Theory of Elasticity*. Course of theoretical physics. Elsevier, 1986.

[Lau et al., 2011] Manfred Lau, Akira Ohgawara, Jun Mitani, and Takeo Igarashi. Converting 3d furniture models to fabricatable parts and connectors. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):85:1–85:6, 2011.

[Lee and Terzopoulos, 2006] Sung-Hee Lee and Demetri Terzopoulos. Heads up!: Biomechanical modeling and neuromuscular control of the neck. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 25(3):1188–1198, 2006.

[Levin et al., 2013] Anat Levin, Daniel Glasner, Ying Xiong, Frédo Durand, William Freeman, Wojciech Matusik, and Todd Zickler. Fabricating BRDFs at high spatial resolution using wave optics. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):144:1–144:14, 2013.

[Li et al., 2011] Xian-Ying Li, Tao Ju, Yan Gu, and Shi-Min Hu. A geometric study of v-style pop-ups: Theories and algorithms. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):98:1–98:10, 2011.

[Martin et al., 2010] Sebastian Martin, Peter Kaufmann, Mario Botsch, Eitan Grinspun, and Markus Gross. Unified simulation of elastic rods, shells, and solids. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 29(4):39:1–39:10, 2010.

[Martin et al., 2011] Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus Gross. Example-based elastic materials. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):72:1–72:8, 2011.

[Massarwi et al., 2007] Fady Massarwi, Craig Gotsman, and Gershon Elber. Papercraft models using generalized cylinders. In *Proceedings of Pacific Graphics*, pages 148–157. IEEE, 2007.

[Matusik et al., 2009] Wojciech Matusik, Boris Ajdin, Jinwei Gu, Jason Lawrence, Hendrik P. A. Lensch, Fabio Pellacini, and Szymon Rusinkiewicz. Printing spatially-varying reflectance. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 28(5):128:1–128:9, 2009.

[McInerney and Terzopoulos, 1993] Tim McInerney and Demetri Terzopoulos. A finite element model for 3d shape reconstruction and nonrigid motion tracking. In *Proc. 4th Int. Conf. Computer Vision*, pages 518–523, 1993.

[McNamara et al., 2004] Antoine McNamara, Adrien Treuille, Zoran Popović, and Jos Stam. Fluid control using the adjoint method. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 23(3):449–456, 2004.

[Miguel et al., 2012] E. Miguel, D. Bradley, B. Thomaszewski, B. Bickel, W. Matusik, M. A. Otaduy, and S. Marschner. Data-driven estimation of cloth simulation models. *Computer Graphics Forum*, 31(2pt2):519–528, 2012.

[Mitani and Suzuki, 2004] Jun Mitani and Hiromasa Suzuki. Making papercraft toys from meshes using strip-based approximate unfolding. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 23(3):259–263, 2004.

[Mordatch et al., 2012] Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 31(4):43:1–43:8, 2012.

[Mori and Igarashi, 2007] Yuki Mori and Takeo Igarashi. Plushie: An interactive design system for plush toys. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 26(3):45:1–45:8, 2007.

[Müller et al., 2004] Matthias Müller, Richard Keiser, Andrew Nealen, Mark Pauly, Markus Gross, and Marc Alexa. Point-based animation of elastic, plastic and melting objects. In *Proceedings of Symposium on Computer Animation*, pages 141–151, 2004.

[Narain et al., 2012] Rahul Narain, Armin Samii, and James F. O'Brien. Adaptive anisotropic remeshing for cloth simulation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 31(6):152:1–152:10, 2012.

[Nealen et al., 2006] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically based deformable models in computer graphics. *Computer Graphics Forum*, 25(4):809–836, 2006.

[Nocedal and Wright, 2000] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 2000.

[O'Brien and Hodgins, 1999] James F. O'Brien and Jessica K. Hodgins. Graphical modeling and animation of brittle fracture. In *Proceedings of SIGGRAPH 99*, Annual Conference Series, pages 137–146, 1999.

[Ogden, 1972] R. W. Ogden. Large deformation isotropic elasticity: On the correlation of theory and experiment for compressible rubberlike solids. *Proceedings of the Royal Society of London. Series A*, 328(1575), 1972.

[Okabe et al., 1992] Hidehiko Okabe, Haruki Imaoka, Takako Tomiha, and Haruo Niwaya. Three dimensional apparel cad system. *SIGGRAPH Computer Graphics*, 26(2):105–110, July 1992.

[Öztireli et al., 2009] Cengiz Öztireli, Gaël Guennebaud, and Markus Gross. Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum (Proceedings Eurographics)*, 28(2), 2009.

[Pai et al., 2001] Dinesh K. Pai, Kees van den Doel, Doug L. James, Jochen Lang, John E. Lloyd, Joshua L. Richmond, and Som H. Yau. Scanning physical interaction behavior of 3d objects. In *Proceedings of SIGGRAPH 99*, Annual Conference Series, pages 87–96, 2001.

[Papas et al., 2013] Marios Papas, Christian Regg, Wojciech Jarosz, Bernd Bickel, Philip Jackson, Wojciech Matusik, Steve Marschner, and Markus Gross. Fabricating translucent materials using continuous pigment mixtures. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):146:1–146:12, 2013.

[Pathmanathan et al., 2009] P. Pathmanathan, SJ Chapman, and DJ Gavaghan. Inverse membrane problems in elasticity. *Quaterly Journal of Mechanics and Applied Mathematics*, 62(1):67–87, Feb 2009.

[Pipkin, 1986] Allen C. Pipkin. The relaxed energy density for isotropic elastic membranes. *IMA Journal of Applied Mathematics*, 36(1):85–99, 1986.

[Popović et al., 2000] Jovan Popović, Steven M. Seitz, Michael Erdmann, Zoran Popović, and Andrew Witkin. Interactive manipulation of rigid body simula-

tions. In *Proceedings of SIGGRAPH 00*, Annual Conference Series, pages 209–217, 2000.

[Pottmann et al., 2007] Helmut Pottmann, Andreas Asperl, Michael Hofer, and Axel Kilian. *Architectural Geometry*. Bentley Institute Press, 2007.

[Prévost et al., 2013] Romain Prévost, Emily Whiting, Sylvain Lefebvre, and Olga Sorkine-Hornung. Make it stand: Balancing shapes for 3d fabrication. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):81:1–81:10, 2013.

[Raibert and Hodgins, 1991] Marc H. Raibert and Jessica K. Hodgins. Animation of dynamic legged locomotion. In *Proceedings of SIGGRAPH 91*, Annual Conference Series, pages 349–358, 1991.

[Rohmer et al., 2010] Damien Rohmer, Tiberiu Popa, Marie-Paule Cani, Stefanie Hahmann, and Alla Sheffer. Animation wrinkling: Augmenting coarse cloth simulations with realistic-looking wrinkles. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 29(6):157:1–157:8, 2010.

[Rose et al., 2007] Kenneth Rose, Alla Sheffer, Jamie Wither, Marie-Paule Cani, and Boris Thibert. Developable surfaces from arbitrary sketched boundaries. In *Proceedings of Symposium on Geometry Processing*, pages 163–172. Eurographics Association, 2007.

[Rozvany, 2001] G.I.N. Rozvany. Aims, scope, methods, history and unified terminology of computer-aided topology optimization in structural mechanics. *Structural and Multidisciplinary Optimization*, 21(2):90–108, 2001.

[Sander et al., 2003] P. V. Sander, Z. J. Wood, S. J. Gortler, J. Snyder, and H. Hoppe. Multi-chart geometry images. In *Proceedings of Symposium on Geometry Processing*, pages 146–155. Eurographics Association, 2003.

[Schenk and Gärtner, 2006] Olaf Schenk and K. Gärtner. On fast factorization pivoting methods for symmetric indefinite systems. *Electronic Transactions on Numerical Analysis*, 23(1):158–179, 2006.

[Shatz et al., 2006] Idan Shatz, Ayellet Tal, and George Leifman. Paper craft models from meshes. *The Visual Computer*, 22(9-11):825–834, 2006.

[Sheffer et al., 2005] Alla Sheffer, Bruno Lévy, Maxim Mogilnitsky, and Alexander Bogomyakov. Abf++: Fast and robust angle based flattening. *ACM Transactions on Graphics*, 24(2):311–330, 2005.

[Shewchuk, 1996] Jonathan Richard Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148, pages 203–222. Springer-Verlag, 1996.

[Sifakis et al., 2005] Eftychios Sifakis, Igor Neverov, and Ronald Fedkiw. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 24(3):417–425, 2005.

[Smith et al., 2002] Jeffrey Smith, Jessica K. Hodgins, Irving Oppenheim, and Andrew Witkin. Creating models of truss structures with optimization. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 21(3):295–301, 2002.

[Solomon et al., 2012] Justin Solomon, Etienne Vouga, Max Wardetzky, and Eitan Grinspun. Flexible developable surfaces. *Computer Graphics Forum*, 31(5):1567–1576, 2012.

[Sorkine et al., 2002] Olga Sorkine, Daniel Cohen-Or, Rony Goldenthal, and Dani Lischinski. Bounded-distortion piecewise mesh parameterization. In *Proceedings of the Conference on Visualization '02*, pages 355–362. IEEE Computer Society, 2002.

[Stava et al., 2012] Ondrej Stava, Juraj Vanek, Bedrich Benes, Nathan Carr, and Radomír Měch. Stress relief: improving structural strength of 3d printable objects. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 31(4):48:1–48:11, 2012.

[Steigmann, 1990] D. J. Steigmann. Tension-field theory. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 429(1876), 1990.

[Sueda et al., 2008] Shinjiro Sueda, Andrew Kaufman, and Dinesh K. Pai. Musculotendon simulation for hand animation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 27(3):83:1–83:8, 2008.

[Tan et al., 2011] Jie Tan, Yuting Gu, Greg Turk, and C. Karen Liu. Articulated swimming creatures. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):58:1–58:12, 2011.

[Teran et al., 2005] Joseph Teran, Eftychios Sifakis, Geoffrey Irving, and Ronald Fedkiw. Robust quasistatic finite elements and flesh simulation. In *Proceedings of Symposium on Computer Animation*, pages 181–190. ACM, 2005.

[Terzopoulos et al., 1987] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. *SIGGRAPH Computer Graphics*, 21(4):205–214, 1987.

[Terzopoulos et al., 1989] Demetri Terzopoulos, John Platt, and Kurt Fleischer. Heating and melting deformable models (from goop to glop). *Graphics Interface '89*, pages 219—-226, 1989.

[Thürey et al., 2006] N. Thürey, R. Keiser, M. Pauly, and U. Rüde. Detail-preserving fluid control. In *Proceedings of Symposium on Computer Animation*, pages 7–12, 2006.

[Treloar, 1944] L. R. G. Treloar. Stress-strain data for vulcanised rubber under various types of deformation. *Transactions of the Faraday Society*, 40(59), 1944.

[Tutte, 1963] W. T Tutte. How to draw a graph. *Proc. London Math. Soc.*, 3(13):743–767, 1963.

[Twigg and Kačić-Alesić, 2011] Christopher D. Twigg and Zoran Kačić-Alesić. Optimization for sag-free simulations. In *Proceedings of Symposium on Computer Animation*, pages 225–236, 2011.

[Umetani and Schmidt, 2013] Nobuyuki Umetani and Ryan Schmidt. Cross-sectional structural analysis for 3d printing optimization. In *SIGGRAPH Asia 2013 Technical Briefs*, pages 5:1–5:4, 2013.

[Umetani et al., 2011] Nobuyuki Umetani, Danny M. Kaufman, Takeo Igarashi, and Eitan Grinspun. Sensitive couture for interactive garment modeling and editing. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):90:1–90:12, 2011.

[Umetani et al., 2012] Nobuyuki Umetani, Takeo Igarashi, and Niloy J. Mitra. Guided exploration of physically valid shapes for furniture design. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 31(4):86:1–86:11, 2012.

[Vidimče et al., 2013] Kiril Vidimče, Szu-Po Wang, Jonathan Ragan-Kelley, and Wojciech Matusik. Openfab: A programmable pipeline for multi-material fabrication. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):136:1–136:12, 2013.

[Volino et al., 2009] Pascal Volino, Nadia Magnenat-Thalmann, and Francois Faure. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Transactions on Graphics*, 28(4):105:1–105:16, 2009.

[Wang and Tang, 2007] Charlie C.L. Wang and Kai Tang. Woven model based geometric design of elastic medical braces. *Computer-Aided Design*, 39(1):69 – 79, 2007.

[Wang and Tang, 2010] Charlie C. L. Wang and Kai Tang. Pattern computation for compression garment by a physical/geometric approach. *Computer-Aided Design*, 42(2):78–86, 2010.

[Wang et al., 2011] Huamin Wang, James F. O'Brien, and Ravi Ramamoorthi. Data-driven elastic models for cloth: Modeling and measurement. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):71:1–71:12, 2011.

[Wang, 2008] Charlie Wang. Computing length-preserved free boundary for quasi-developable mesh segmentation. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):25–36, 2008.

[Weyrich et al., 2009] Tim Weyrich, Pieter Peers, Wojciech Matusik, and Szymon Rusinkiewicz. Fabricating microgeometry for custom surface reflectance. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 28(3):32:1–32:6, 2009.

[Whiting et al., 2009] Emily Whiting, John Ochsendorf, and Frédo Durand. Procedural modeling of structurally-sound masonry buildings. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 28(5):112:1–112:9, 2009.

[Whiting et al., 2012] Emily Whiting, Hijung Shin, Robert Wang, John Ochsendorf, and Frédo Durand. Structural optimization of 3d masonry buildings. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 31(6):159:1–159:11, 2012.

[Wicke et al., 2010] Martin Wicke, Daniel Ritchie, Bryan M. Klingner, Sebastian Burke, Jonathan R. Shewchuk, and James F. O'Brien. Dynamic local remeshing for elastoplastic simulation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 29(4):49:1–49:11, 2010.

[Witkin and Kass, 1988] Andrew Witkin and Michael Kass. Spacetime constraints. *SIGGRAPH Computer Graphics*, 22(4):159–168, 1988.

[Wojtan and Turk, 2008] Chris Wojtan and Greg Turk. Fast viscoelastic behavior with thin features. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 27(3):47:1–47:8, 2008.

[Wu and Popović, 2003] Jia-chi Wu and Zoran Popović. Realistic modeling of bird flight animations. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 22(3):888–895, 2003.

[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. *Computer Graphics Forum (Proceedings Eurographics)*, 20(3):349–358, 2001.

[Xin et al., 2011] Shiqing Xin, Chi-Fu Lai, Chi-Wing Fu, Tien-Tsin Wong, Ying He, and Daniel Cohen-Or. Making burr puzzles from 3d models. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):97:1–97:8, 2011.

[Yin et al., 2007] KangKang Yin, Kevin Loken, and Michiel van de Panne. SIMBICON: Simple biped locomotion control. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 26(3), 2007.

[Yu et al., 2011] Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher. Make it home: Automatic optimization of

furniture arrangement. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):86:1–86:12, 2011.

[Zhu et al., 2012] Lifeng Zhu, Weiwei Xu, John Snyder, Yang Liu, Guoping Wang, and Baining Guo. Motion-guided mechanical toy modeling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 31(6):127:1–127:10, 2012.

Curriculum Vitae

Mélina Skouras

Personal Data

Nov. 27, 1981	Born in Paris, France
Nationality	French

Education

Sept. 24, 2014	Ph.D. defense.
Jan. 2010 – Sept. 2014	Ph.D. studies at the Computer Graphics Laboratory of ETH Zurich, Switzerland. Prof. Markus Gross.
Sept. 2004	Diplôme d'Ingénieur (Master's Degree), ENSIMAG, Grenoble, France. Option: Images & Virtual Reality.
Sept. 2003 – Apr. 2004	Visiting student at the University of Waterloo, Ontario, Canada. Courses in Software Engineering, Computer Graphics, Image Processing and Computer Vision.
Sept. 2001 – Apr. 2004	Studies at ENSIMAG (National Graduate Engineering School of Computer Science and Applied Mathematics), INPG, Grenoble, France. Options in Geometric Modeling, Image Processing, User Interface.
Sept. 1999 – Apr. 2001	CPGE (Preparatory cycle to graduate engineering schools), MPSI-PSI*, Lycée Saint Louis, Paris, France.
June 1999	Baccalauréat, major in Mathematics, graduated with first class honours, Lycée Michelet, Vanves, France.

Employment

Since Jan. 2010	Research assistant at ETH Zurich, Zurich, Switzerland.
Jan. 2005 – Dec. 2009	Software developer at Dassault Systèmes, CATIA Geometric Modeler Team, Vélizy, France.
Apr. 2004 – Sept. 2004	Internship at THALES Training & Simulation, Visual Technologies Team, Cergy-Pontoise, France.
July 2003 – Aug. 2003	Internship at LMC-Imag (Laboratoire de Modélisation et Calcul, now part of Laboratoire Jean Kuntzmann) Grenoble, France.

Scientific Publications

M. Skouras, B. Thomaszewski, B. Bickel, P. Kaufmann, A. Garg, E. Grinspun and M. Gross. Designing Inflatable Structures, *Proceedings of ACM SIGGRAPH (Vancouver, Canada, August 10-14, 2014), ACM Transactions on Graphics*, vol. 34, no. 4, 2014.

M. Skouras, B. Thomaszewski, S. Coros, B. Bickel and M. Gross. Computational Design of Actuated Deformable Characters, *Proceedings of ACM SIGGRAPH (Anaheim, USA, July 21-25, 2013), ACM Transactions on Graphics*, vol. 32, no. 4, pp. 82:1–82:10, 2013.

B. Bickel, P. Kaufmann, M. Skouras, B. Thomaszewski, D. Bradley, T. Beeler, P. Jackson, S. Marschner, W. Matusik and M. Gross. Physical Face Cloning, *Proceedings of ACM SIGGRAPH (Los Angeles, USA, August 5-9, 2012), ACM Transactions on Graphics*, vol. 31, no. 4, pp. 118:1–118:10, 2012.

M. Skouras, B. Thomaszewski, B. Bickel, and M. Gross. Computational Design of Rubber Balloons, *Proceedings of Eurographics (Cagliari, Italy, May 13-18, 2012), Computer Graphics Forum*, vol. 31, no. 2, pp. 835–844, 2012.