From Digital to Physical: Computational Aspects of 3D Manufacturing

A DISSERTATION PRESENTED
BY
MORITZ NIKLAUS BÄCHER
TO
THE SCHOOL OF ENGINEERING AND APPLIED SCIENCES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN THE SUBJECT OF
COMPUTER SCIENCE

HARVARD UNIVERSITY
CAMBRIDGE, MASSACHUSETTS
SEPTEMBER 2013

Thesis advisor: Prof. Hanspeter Pfister Moritz Niklaus Bächer

From Digital to Physical: Computational Aspects of 3D Manufacturing

Abstract

The desktop publishing revolution of the 1980s is currently repeating itself in 3D, referred to as *desktop manufacturing*. Online services such as Shapeways have become available, making personalized manufacturing on cutting edge additive manufacturing (AM) technologies accessible to a broad audience. Affordable desktop printers will soon take over, enabling people to fabricate custom 3D models at home.

Contemporary AM technologies have advanced enough to enable 3D printing at high resolution, in full-color, and with mixtures of soft and hard materials. As opposed to subtractive manufacturing (SM) such as milling or drilling, they can fabricate highly complex assemblies without the need for a manual assembly of individual components. Yet, one of the major issues holding back widespread use of AM is the lack of efficient algorithms for the automated fabrication of digital CG, and the reproduction of physical content. Besides, we do not have tools at our disposal that aid us with the design of multi-material content or complex assembly structures.

For physical reproduction, we strive for methods to acquire properties such as, e.g., reflectance (*appearance*) or elasticity (*deformation behavior*) from real-world objects, representing them digitally, then automating their fabrication using AM. However, the vast majority of digital 3D content are directly designed on comput-

ers, hence, potentially exhibit a highly non-physical behavior. To fabricate such content, we seek methods for the automated estimation of physical models from these digital ones.

This dissertation examines computational aspects of 3D manufacturing. In particular, we investigate design tools and automated fabrication of an object's deformation behavior, articulation, and geometry. We present a complete process for measuring, representing, simulating, and physically fabricating an object's elastic deformation behavior. This process enables the reproduction of physical deformation behavior. Furthermore, we introduce a technique for the automated fabrication of articulated models, estimated from the most widely used format in character animation – so called skinned meshes. Our technique estimates assemblies, approximating this inherently non-physical input in a piecewise linear manner. Lastly, we propose a method for the scale-aware fabrication of static geometry, capable of abstracting, then engraving details that cannot be fabricated on a pre-specified 3D printer.

Contents

Listing of figures

List of Tables

To my parents, Esther and Martin, and my sister Franziska.

Acknowledgments

Foremost, I would like to express my deepest gratitude to my advisor Hanspeter Pfister for his support. He always had an open ear for all my ideas and encouraged me to work on those I was most excited about. His guidance played a key role in my academic and professional development.

Besides my advisor, I would like to thank my other committee members, Steven Gortler and Todd Zickler, for their feedback and encouragement. Shlomo always had an open door and his graphics course on image-based rendering was most enjoyable and inspiring. Todd deserves a special thank you for his invaluable advice in building our data acquisition systems.

During my Ph.D. studies, I was most fortunate to work with some of the most capable collaborators one can possibly have. Without their persistent help, this dissertation would not have been possible. I am in dept to Bernd Bickel, Nathan Carr, Markus Gross, Doug L. James, Wojciech Matusik, Radomir Měch, Miguel A.

Otaduy, and James Weaver. I would especially like to thank Doug for his unconditional support when I needed it the most. I also had the great pleasure to closely work with Gaurav Bharaj, Chirantan Ekbote, Miloš Hašan, Hyunho Richard Lee and wish to thank them for their hard work and fruitful discussions.

I also owe all my fellow labmates a special thank you for putting up with my unconscious habit of drumming with hands and feet and for making lab times most enjoyable. I am especially grateful to my lab neighbor and long-term companion Kalyan Sunkavalli for his constant help with all (Ph.D.) life matters, and Guillermo Diez-Cañas and Ioannis Gkioulekas for our discussions on C++ programming and their availability to bounce ideas.

Furthermore, I would like to thank the members of the Visual Computing Group for all their support and valuable feedback, and the computer graphics group at TU Berlin for their warm welcome and good Berlin times. Many of you became dear friends!

I am also in dept to Robert Wood, Radhika Nagpal, the Wyss Institute for Biologically Inspired Engineering, Joe Marks, and Disney Research Boston for letting me use their Objet printers during the past years.

Finally but most importantly, I would like to thank my parents, Esther and Martin Bächer, and my sister Franziska for encouraging me to follow my heart and dreams. Without their love and unconditional support, this thesis would not have been possible at the first place.

The purpose of computing is insight, not numbers.

Richard Hamming

1 Introduction

The desktop publishing revolution of the 1980s is currently repeating itself in 3D, referred to as *desktop manufacturing*. With contemporary additive manufacturing (AM) technologies, we can 3D print models with stunning detail using a wide range of materials including plaster, paper, elasto- and hard plastics, as well as several metals. Other AM technologies enable printing in full-color or with mix-

tures of soft and hard materials, within the same model and print job. With AM, we can fabricate highly complex assemblies that, unlike traditional manufacturing such as milling or drilling, do not require any manual assembly of individual components. We seem only steps away from what Neil Gershenfeld refers to as "personal fabricators" (PFs) [2005], devices envisioned to build objects at the atomic level. While PFs are still more fiction than reality, we will soon be able to print such assemblies, consisting of mechanical and deformable parts, with custom reflectance and scattering properties, and with embedded sensors and actuators – all on a single AM device and as single assembled pieces.

Moreover, online services such as Shapeways have become available, making personalized manufacturing on such cutting edge AM devices accessible to a broad audience, thereby taking on the role of print shops in the early stages of the revolution in 2D. Affordable desktop printers will soon be available, enabling people to fabricate personalized 3D content at home at the press of a button.

Despite these technical advances, AM is still of limited use due to the lack of efficient tools and algorithms for the automated fabrication of digital, and the reproduction of physical content. And, we do not have tools at our disposal that aid us with the design of multi-material content or complex assembly structures.

Both digital CG and physical objects are complex functions of space and time. Their appearance varies with position and orientation, and their elasticity, plasticity, and viscosity properties capture their behavior when undergoing deformations. For physical reproduction, we strive after methods for acquiring these properties from real-world objects, representing them digitally, then automating their

fabrication using AM. However, the vast majority of digital 3D content are directly designed on computers, hence, potentially exhibit a highly non-physical behavior. To fabricate such content, we seek methods for the automated estimation of physical models from these digital ones.

This dissertation examines computational aspects of 3D manufacturing an object's deformation behavior, articulation, and geometry. In particular, we present a complete process for measuring, representing, simulating, and physically fabricating an object's elastic deformation behavior. While our process allows to physically reproduce deformation properties, it is by no means restricted to inputs sampled from real-world objects. Probes can also be taken from simulations of deformable models, enabling the fabrication of digital content.

Furthermore, we introduce a technique for the automated fabrication of articulated models, estimated from the most widely used format in character animation – so called skinned meshes. In contrast to the input to our reproduction process, these skinned representations are inherently non-physical. Moreover, animated characters for feature film or computer games are often purposefully over-exaggerated, exhibiting a toon-like articulation. Our technique estimates assemblies, approximating these non-physical deformable input models in a piecewise linear manner.

In addition to the above, we propose a method for the scale-aware fabrication of static geometry. Although an object's detailed geometry may render correctly at any screen resolution and far camera views, features too fine and thin may be fused or break during printing. Prior work has addressed the automated detection and

thickening of such critical features [Stava et al. 2012]. However, while thickening leads to pleasing results when printing models at mid- to large scales, and with strong and detailed materials, it results in blobby models when we aim for small scales or print at lower resolutions and with weaker materials. Our method is capable of abstracting, then engraving such surface features. While we avoid weak output models using our abstraction, we keep as much of the perceived detail as possible by engraving.

1.1 Outline

After reviewing related work in computer graphics and relevant fields in the next chapter, we give a short primer on contemporary additive manufacturing, and their advantages over traditional manufacturing techniques (Chapter 3).

As indicated in Figure 1.1.1 in brown, we will introduce two measurement systems – one simple-to-build and hand-held (Section 4.4), one fully automated (Section 5.6) – that allow acquiring force-displacement samples of a wide range of deformable objects, including human soft tissue.

In Chapter 4, we then use sets of such example deformations to construct convincing deformable "clones" of physical objects (Figure 1.1.1, red). By doing so, we avoid complex selection and tuning of physical material parameters, yet retain the richness of non-linear heterogeneous elastic behavior. We represent each force-displacement sample as a spatially-varying stress-strain relationship in a finite-element model. We then model the material by a non-linear interpolation of these

stress-strain relationships in strain-space. Our run-time simulation algorithm is based on incremental loading, making it suitable for interactive computer graphics applications. We present the results of our approach for several non-linear materials and biological soft tissue, with accurate agreement of our model to the measured data.

Next, we introduce our reproduction process (Figure 1.1.1, blue), enabling the

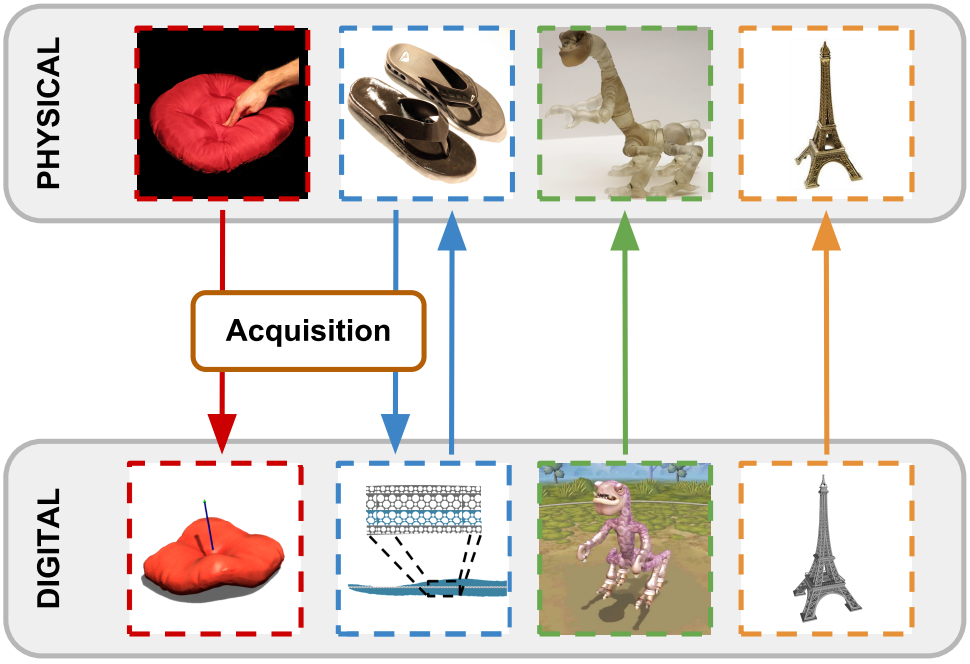


Figure 1.1.1: Computational Aspects of 3D Manufacturing: To probe an object's elastic deformation behavior, we built two acquisition systems (brown, Sections 4.4 and 5.6). From acquired deformation probes, we first estimate digital, deformable "clones" (red, Chapter 5), then physically reproduce them using AM (blue, Chapter 5). Next, we estimate fabricatable articulated characters with internal joints from skinned input meshes (green, Chapter 6). Final 3D printed characters have durable joints with a frictional design for character posing. Lastly, we estimate abstracted models from a detailed input geometry, then engrave fine detail to ensure it is still perceived in the final printouts (orange, Chapter 7).

automated fabrication of digital deformation behavior on multi-material AM devices. Our process starts with measuring deformation properties of a set of printable base materials using our automated measurement system (Section 5.6), followed by representing each of them as a non-linear stress-strain relationship using the data-driven technique described in Chapter 4. We then introduce an optimization process that finds the best combination of stacked layers of base materials, given a user-specified or measured deformation behavior in form of example deformations. We demonstrate a complete "physical cloning" process by acquiring and fabricating the deformation behavior of several objects with complex non-linear and heterogeneous material properties.

In Chapter 6 and as illustrated in Figure 1.1.1 in green, we introduce our technique to fabricate articulated characters from skinned meshes. We first extract a set of potential joint locations. From this set, together with optional, user-specified range constraints, we then estimate mechanical friction joints that satisfy inter-joint non-penetration and other fabrication constraints. To avoid brittle joint designs, we place joint centers on an approximate medial axis representation of the input geometry, and maximize each joint's minimal cross-sectional area. We provide several demonstrations, manufactured as single, assembled pieces using 3D printers.

Before we conclude with a summary and outlook in Chapter 8, we discuss our scale-aware fabrication of detailed geometry in Chapter 7 (Figure 1.1.1, orange): we first estimate medial ball representations of our input geometry (union of interior balls) and its embedding (union of exterior balls). Next, we abstract non-

fabricatable detail by analyzing radii along edges of a medial graph connecting all exterior balls, then marking a subset as interior. Thereafter, we extract the surface separating interior from exterior balls, resulting in a watertight mesh, free of self-intersecting faces. We engrave non-fabricatable detail by unifying the union of ball representations of our original input with offset ball representations of our abstraction.

1.2 Contributions

All our contributions summarized herein are bridging the gab between the physical and digital in one way or the other:

Sections 4.4 and 5.6 We contribute two measurement systems tailored for the non-invasive acquisition of elastic deformation behavior: one with a hand-held probe, one with a probe attached to a 4 DOF robotic arm. We use stereo-vision subsystems to track painted markers during interactions, making the acquisition of surface displacements independent of the object's appearance properties and robustness w.r.t. occlusions during active probing. Our hand-held system is well-suited for the sampling of physical objects and human tissue at arbitrary locations, varying angles, and with custom contact shapes. Our automated system is tailored for repeatable high precision acquisition of elastic materials. The resulting force-displacement samples serve us as input to our data-driven digital cloning and physical reproduction.

Chapter 4 Given a set of elastic deformation samples, we fit a static co-rotational FEM formulation to each of these force-displacement pairs. To this end, we discretize a solid model into homogeneous, isotropic tetrahedra, then estimate the per-element Poisson ratios and Young's moduli from the measured displacement constraints and applied forces. To tackle this underconstrained, inverse problem, we regularize with a Laplacian matrix, enforcing smoothness of parameters between neighboring elements. For simulations of a captured elastic behavior, we use incremental loading.

Chapter 5 We propose a complete process for the physical reproduction and design of elastic deformable materials using AM.

Chapter 6 We introduce a technique for the automated fabrication of an articulated deformable character on an AM device. Our method takes a skinned mesh as input, then estimates a fabricatable single-material model that approximates the 3D kinematics of the corresponding virtual articulated character in a piece-wise linear manner.

Section 6.4.1 We show that an analysis of skinning weights leads to a plausible segmentation of the character's geometry into rigid body parts. Transitions between neighboring segments mark potential joint locations.

Section 6.4.2 We propose novel geometric approximate models of joint strength that enable the estimation of strong mechanical friction joints by maximizing their minimal cross-sectional areas.

Section 6.4.4 To ensure strong and functional joints in our output model, we introduce a collision resolution that allows to keep as much of the "fabricatable" input articulation as possible, while avoiding inter-joint penetration.

Chapter 7 We propose an automated processing, enabling the small-scale fabrication of detailed static geometry. Inspired by souvenir manufacturing, we abstract features too fine and thin, then engrave them so that they are still perceived in printouts.

Sections 7.4 and 7.5 We extend a representation of unions of interior and exterior medial balls [Amenta et al. 2001a;b] with abstraction and set boolean operations that enable the selective closing of concavities, and the unification, intersection, and difference of volumes represented by these balls. For closing, we propose analyzing the change of rate of medial ball radii along branch ends of the exterior medial axis. Abstractly speaking, these operations enable the editing and merging of medial axis transforms.

1.3 Publications

Chapters 4, 5, and 6 present material published in the following peer-reviewed publications.

Chapter 4 B. Bickel, M. Bächer, M. A. Otaduy, W. Matusik, H. Pfister, M. Gross. *Capture and Modeling of Non-Linear Heterogeneous Soft Tissue*. In Proceedings of ACM SIGGRAPH (New Orleans, USA, August 3-7, 2009), ACM Transactions on Graphics, vol. 28, no. 3.

Chapter 5 B. Bickel, M. Bächer, M. A. Otaduy, H. R. Lee, H. Pfister, M. Gross, W. Matusik. *Design and Fabrication of Materials with Desired Deformation Behavior*. In Proceedings of ACM SIGGRAPH (Los Angeles, USA, July 25-29, 2010), ACM Transactions on Graphics, vol. 29, no. 3.

Chapter 6 M. Bächer, B. Bickel, D. L. James, H. Pfister. *Fabricating Articulated Characters from Skinned Meshes*. In Proceedings of ACM SIGGRAPH (Los Angeles, USA, August 5-9, 2012), ACM Transactions on Graphics, vol. 31, no. 4.

During the time period of this thesis (but not directly related) following workshop papers were published:

1. A. Peters Randles, M. Bächer, H. Pfister, E. Kaxiras. *A Lattice Boltzmann Simulation of Hemodynamics in a Patient-Specific Aortic Coarctation Model.* STACOM 2012 Workshop, Held in Conjunction with MICCAI 2012, LNCS vol. 7746, Springer, 2012.

2. H. Zhang, J. K. Lai, M. Bächer. *Hallucination: A Mixed-Initiative Approach for Efficient Document Reconstruction*. The 4th Human Computation Workshop (HCOMP), 2012.

Either write something worth reading or do something worth writing.

Benjamin Franklin

2

Related Work

Before diving into the specifics of capturing and modeling deformable objects, physically cloning such "deformables", and automating the fabrication of skinned characters and simplified geometry, we discuss work related to these topics in and beyond computer graphics.

We start with an overview of manufacturing work in graphics, then discuss prior

work on measuring deformations of physical objects in Section 2.2. Thereafter, we review the large body of prior art on the simulation of deformable models in mechanical engineering and graphics in Section 2.3. We extend our review to physical reproduction in Section 2.4. Next, we discuss work on animation and toy manufacturing (Section 2.5) as they are most closely related to our fabrication work on skinned characters. For our fabrication-related geometric processing, we review prior work in computational geometry and graphics in Section 2.6.

2.1 Fabrication in Computer Graphics

Triggered by the recent advances and popularity of 3D manufacturing technologies, the computer graphics community has intensified their efforts in bridging the gab between the digital and the physical. While capture and data-driven techniques have a long tradition, the reverse process of fabricating digital content has only recently gotten the attention of the broader community.

To automate this process, we have to successfully map three components of a given virtual model to reality: its two static properties, namely geometry and appearance, and its dynamic properties such as, e.g., its articulation or deformation behavior.

For the 3D manufacturing of *geometry*, three key problems have been addressed by prior work: while "stress relief" [Stava et al. 2012] and Zhou et al.'s work [2013] detect and correct structurally unsound geometry, "chopper" [Luo et al. 2012] and "make it stand" [Prévost et al. 2013] allow to partition a model into 3D-printable parts (scalability) and make it stand as initially intended (balance). We comple-

ment these three techniques with a geometric processing framework that allows to simplify geometry in a scale-aware manner. We defer a detailed review of closely related work to Section 2.6.

Our community has also devoted efforts to the fabrication of *appearance* properties such as a model's reflectance and subsurface scattering: Weyrich and colleagues [2009] use computer-controlled milling to manufacture custom surface reflectance and Matusik et al. [2009] 2D ink printing to fabricate spatially-varying isotropic reflectance. Anisotropy has been addressed using opaque ink on a reflective substrate [Malzbender et al. 2012], a combination of 3D and UV printing [Lan et al. 2013], or wave optics [Levin et al. 2013]. To approximate a model's homogeneous and inhomogeneous subsurface scattering, Dong et al. [2010] and Hasan et al. [2010] fabricate layers of varying thickness of translucent materials using milling and 3D printing. More recently, Papas et al. [2013] use continuous pigment mixtures to avoid discretization artifacts for homogeneous scattering of the previous two techniques.

In our work, we address two key aspects of a model's *behavior under motion*. Firstly, we reproduce a model's deformation behavior using multi-material printing (see Chapter 5). Secondly and as discussed in Chapter 6, we approximate the articulation of skinned characters – the most widely used format in animation – in a piecewise linear manner. Later (but considered concurrent), Calì et al. [2012] propose an approach aiding the design of articulated models. However, unlike ours, their technique starts off with static geometry, while we automatically estimate printable, jointed toy models from a format that encodes articulation.

While our articulated characters are manually posed, Zhu et al. [2012] and Coros, Thomaszewski, and colleagues [2013] propose systems for the automated design of mechanically actuated, animated characters. The design of actuated, deformable characters from a set of target poses has been studied by Skouras et al. [2013]. Moreover, Bickel et al. [2012] propose a method to physically clone faces. Most recently, "OpenFab" [2013] and "Spec2Fab" [2013a] were introduced, facilitating the design and fabrication of multi-material content.

Besides the above, our community has contributed tools and techniques to design and fabricate cloth [Okabe et al. 1992], paper craft [Mitani and Suzuki 2004; Kilian et al. 2008; Chen et al. 2013b], pop-ups [Hoiem et al. 2005; Li et al. 2010; 2011], plush toys [Mori and Igarashi 2007], reliefs [Weyrich et al. 2007; Alexa and Matusik 2010], 3D puzzles [Lo et al. 2009; Xin et al. 2011; Song et al. 2012], custom-made metallophones [Umetani et al. 2010], holography [Regg et al. 2010], multilayer models [Holroyd et al. 2011], furniture [Lau et al. 2011], baloons [Skouras et al. 2012], caustics [Papas et al. 2011], and masonry models [Whiting et al. 2012; Panozzo et al. 2013]. Others use shadow imagery [Mitra and Pauly 2009; Baran et al. 2012; Bermano et al. 2012] and planar slices to approximate geometry [McCrae et al. 2011; Hildebrand et al. 2012; Schwartzburg and Pauly 2013], and propose a 2D cutting tool [Rivers et al. 2012b] and 3D sculpting aid [Rivers et al. 2012a] to support humans during manual manufacturing.

2.2 Deformation Capture

Even though variants of tensile testing [Hart 1967] allow to identify characteristics of composite materials [Smits et al. 2007] and soft tissue [Bursa and Zemanek 2008], specimen dimensions are commonly prescribed. In contrast, we aim for systems that are *easy-to-use* for users unfamiliar with the mechanics of materials and allow to sample the elastic deformation behavior of objects without the need to alternate their rest pose geometry (*preservation* of an object's *static* and *dynamic properties*).

To measure force-displacement samples of deformable materials and human tissue, we use a combination of stereo-vision acquisition systems and force sensors similar to earlier approaches [Pai et al. 2001].

2.3 Modeling Deformation Behavior

Researchers in many fields, ranging from mechanical engineering to biology, have long studied the problem of modeling complex elasticity properties. For a recent survey of deformation models in computer graphics, we refer the interested reader to [Nealen et al. 2006].

Mechanical Models A common approach to model the non-linear stress-strain behavior of complex materials and human tissue is to devise a constitutive model, then tune its parameters until they best fit empirical data. However, while hyperelastic models such as, e.g., the Ogden model [Ogden 1997] capture various behavior regimes of materials and tissue well, this parameter tuning approach is tedious

and utterly complex as it relies on accurate modeling of the layered geometry (e.g., the bones, fat, and muscles for facial tissue), rich excitation of material regimes, and accurate measurement of forces and deformations (even in typically inaccessible regions). Despite the complexity of the approach, it has seen large applications in graphics since the pioneering work by Terzopoulos et al. [Terzopoulos et al. 1987], as it can lead to stunning results with the appropriate amount of effort. Some examples of complex bio-mechanical models in computer graphics include the neck [Lee and Terzopoulos 2006], the torso [Zordan et al. 2004; Teran et al. 2005; DiLorenzo et al. 2008], the face [Koch et al. 1996; Magnenat-Thalmann et al. 2002; Terzopoulus and Waters 1993; Sifakis et al. 2005], and the hand [Sueda et al. 2008].

Measurement-Based Model Fitting To circumvent the complexity of parameter tuning, several authors have proposed measurement-based model fitting approaches. The seminal work of Pai et al. [2001] presents a capture and modeling system for a deformable object's shape, elasticity, and surface roughness. Their deformable model is based on a matrix representation of Green's function [James and Pai 1999], and was later extended to increase fitting robustness by Lang and colleagues [2002], and to handle viscoelasticity by Schoner et al. [2004]. Our approach shares their strategy for measuring surface displacements as the result of applied forces, but, unlike theirs, is not limited to linear material behavior and does not rely on global response functions. Sifakis et al. [2005] give a different spin to measurement-based modeling approaches, as they learn the relationship between facial muscle activation and skin positions. Others, particularly in biome-

chanics, have explored measurement-based fitting of parameters of various constitutive models: Schur and Zabaras [1992] use non-liner least squares to estimate Young's modulus, while Becker and Teschner [2007] employ a linear least squares formulation to estimate both, Young's modulus and Poisson's ratio. The estimation of non-linear viscoelastic materials and plasticity have been addressed by Kauer [2002] and Kajberg and Lindkvist [2004], respectively. Our work borrows from these approaches for the estimation of each individual sample of the stress-strain relationship. However, this alone is not sufficient for capturing the rich non-linear behavior of soft tissue. In contrast to previous work, the realism of our material model is greatly enhanced with spatially-varying non-linear interpolation in strain space.

Data-driven Methods Purely data-driven techniques have gained large popularity in computer graphics, as they produce highly realistic results for phenomena that are otherwise extremely complex to model. The interpolation of light-field samples [Buehler et al. 2001] allows simulating the illumination of complex scenes, while data-driven reflection models [Matusik et al. 2003] represent each bidirectional reflectance distribution function (BRDF) through a dense set of measurements. Data-driven methods have also been applied to several other aspects of deformation modeling in computer graphics, such as facial wrinkle formation from local skin deformations [Ma et al. 2008; Bickel et al. 2008], grasping of objects [Kry and Pai 2006], skeleton-driven cloth wrinkles [Kim and Vendrovsky 2008], body-skin deformation [Park and Hodgins 2006], or learning of skeleton-driven skin dynamics [Park and Hodgins 2008]. Our method is a mixture of model

fitting techniques (i.e., estimating stress-strain parameters from local samples) and data-driven methods (i.e., using tabulated stress-strain parameters and non-linear interpolation during runtime).

Shape Modeling Another common approach to model deformations is shape modeling [Botsch and Sorkine 2008]. Some of the existing approaches rely on predefined examples [Sloan et al. 2001; Allen et al. 2002; Sumner et al. 2005], or even exploit interpolation [Bergeron and Lachapelle 1985; Lewis et al. 2000; Blanz et al. 2003]. However, these techniques cannot model deformations as a reaction to contact in the way our technique does. Some recent approaches connect shape modeling with physically-based reactive models by rigging using templates of forces [Capell et al. 2005] or skeletal interpolation of elastic forces [Galoppo et al. 2009]. Yet, unlike ours, these approaches cannot model a general non-linear, heterogeneous deformation behavior.

2.4 Physically Reproducing Deformation Behavior

For our physical reproduction of deformation behavior, we introduce a complete pipeline to acquire, model, design, and fabricate desired deformation properties using multi-material 3D printing. Similar reproduction pipelines have also been proposed for subsurface scattering [Dong et al. 2010; Hašan et al. 2010; Papas et al. 2013].

While modern multi-material printers [Stratasys 2013] allow to print detailed structures with spatially-varying mixtures of soft, rubber-like and hard, plastic-like materials, we lack tools to design and reproduce such deformable materials. For

physically fabricating a desired deformation behavior, we estimate a layered approximate model consisting of printable base materials, tailored for manufacturing on such multi-material devices.

To represent our base materials, we build on our data-driven technique that we discuss in detail in Chapter 4 with some notable adjustments. First, by restricting the types of materials to homogeneous ones, our model requires far fewer degrees of freedom. Homogeneity of the base materials is not a limitation in our case, since we achieve inhomogeneity in the final output materials by combining various homogeneous materials. Second, we increase the robustness of the fitting process by fitting one single non-linear model to all input examples simultaneously.

Recent work in graphics aims at modeling high-resolution heterogeneities even when the resolution of the discretization is considerably coarser [Kharevych et al. 2009; Nesme et al. 2009]. This process, known as *homogenization*, tries to find parameter values of a constitutive model sampled at low resolution such that the behavior of the object best matches the heterogeneous material. The first step of our reproduction process can be considered as a variant of homogenization, where the fine-scale inhomogeneous material is an actual physical one. In the second step, however, we take the opposite approach to homogenization, generating a heterogeneous object that fits coarse force-deformation data from small-scale materials with known behavior.

Digital materials, composed of a set of discrete voxels, can exhibit widely varying material properties [Hiller and Lipson 2009]. A general introduction to the optimization of spatial material distributions can be found in [Bendsoe and Sig-

mund 2003].

As the design space increases exponentially with the number of possible combinations of base materials, evolutionary algorithms [Kicinger et al. 2005] are a popular non-linear optimization strategy. In contrast, we apply a branch-and-bound search strategy in combination with clustering.

2.5 Fabricating Articulated Characters

In our 3D manufacturing work on articulated characters, we estimate piecewise-rigid, jointed volume models from input characters whose articulation is encoded in its skin.

Articulated characters are widespread in computer animation, with linear blend skinning (LBS) and example-based approaches common [Lewis et al. 2000; Mohr and Gleicher 2003; Kavan et al. 2008]. Most character rigging methods either estimate a skeleton or LBS from a mesh [Baran and Popović 2007] or estimate a skinned character model from example poses [Kry et al. 2002; Mohr and Gleicher 2003; Wang et al. 2007] or input animations [James and Twigg 2005]. We focus on articulation specified as a linear blend skin as it is the most widely used format. However, current AM techniques do not support printing of skinned meshes. Existing tools only convert their appearance and shape properties and ignore their articulation.

Because our targeted output models share strong similarities with articulated toys such as dolls or puppets, and action figures, we draw inspiration from the large body of patents filed on this topic. They describe many mechanical joints ranging

from basic swivel to elaborate, multi-part designs [Abbat 1993; Ferre 2000] that overcome common structural and range shortcomings. However, none of them is based on a geometric model of joint strength that complies with range constraints like our hinge and ball-and-socket designs. To make our joints posable, we fabricate small protrusions similar to [Grey 1999; Wai 2006] that cause friction under joint motion but extent their ideas to prevent fusion during manufacturing.

When recasting our joint optimizations as pure geometric problems, we draw inspiration from structural engineering [Beer et al. 2011]: to increase the strength of a simple structure, civil engineers identify and maximize its critical cross-sectional area. In graphics, similar ideas have been used to automate the generation of truss structures [Smith et al. 2002] and procedural models of buildings [Whiting et al. 2009].

2.6 Scale-Aware Fabrication

Our scale-aware simplification for manufacturing is most closely related to mesh simplification. However, while we aim for reducing the triangle count in typical simplification, we are most concerned with features too thin and fine in a manufacturing context.

Mesh simplification addresses the problem of reducing the complexity of small or distant models and got a tremendous amount of attention after Clark's early work [Clark 1976]. Prominent techniques including vertex clustering [Rossignac and Borrel 1993], vertex merging [Garland and Heckbert 1997] based on quadric error metrics, besides voxel-based [Cohen et al. 1996], envelope-based [Cohen

et al. 1996], and progressive [Hoppe 1996] approaches. However, while our technique shares the capabilities of alternating a model's global topology properties with [Rossignac and Borrel 1993; Cohen et al. 1996], our overall goal is to reduce its complexity w.r.t. minimal feature sizes rather than polygon count. While we draw inspiration from the perceptual-driven abstraction work by Mehra and colleagues [Mehra et al. 2009], we focus on manufacturing constraints not addressed in their work.

Mesh repair While we assume our input to be manifold, closed, and intersection free, our method directly applies to models consisting of several connected components, containing self-intersections, non-manifold faces, and open boundaries by using Jacobson et al.'s generalized winding numbers [Jacobson et al. 2013] as a pre-processor. For an exhaustive review of work prior to [Jacobson et al. 2013], we refer the interested reader to a recent survey [Attene et al. 2013].

Fabricating Geometry Closely related to our work is stress relief [Stava et al. 2012] where the authors propose the use of local thickening, hollowing, and strut insertion to reduce the high stresses in 3D models prior to printing. Our method complements this work in that it uses global abstraction and local engraving to gradually simplify models while avoiding weak links and non-fabricatable features.

Medial Axis Transform (MAT) The extraction of approximate unions of medial ball representations has gotten a tremendous amount of attention since Blum's pioneering work [1967]. Blum observed that a subset of the Voronoi diagram of a dense enough sampling of a given curve or surface, approximates their medial axes. In 2D, all Voronoi vertices lie close to the true axes for such samplings. In 3D,

however, only a subset of these vertices are close, even for arbitrarily dense samplings [Amenta et al. 1998]. This observation led to the definition of poles [Amenta et al. 1998; Amenta and Bern 1998]: Voronoi vertices that are furthest away from samples, one on either side of the surface. However, we found that this subset of 3D Voronoi vertices is rather conservative leading to noisy reconstructions when converting unions of filtered balls back to surface representations. For an exhaustive review of methods to extract and process the MAT, we refer the interested reader to a recent book [Siddiqi and Pizer 2008] and survey [Attali et al. 2009].

Power Crust and Alpha Shapes We base our work on the rigorous power crust algorithm [Amenta et al. 2001a;b]: given a set of points, their algorithm computes an approximate medial axis and surface mesh, referred to as power shape and crust. Applied to the point cloud reconstruction problem, their remarkable technique guarantees the resulting mesh to be "water-tight", and self-intersection free. Moreover, sharp corners and edges are reconstructed with high fidelity. We extend their framework with an adaptive Poisson-disk sampling [Corsini et al. 2012], guaranteeing that corners and edges are well-preserved when processing a given input mesh. In earlier work, Amenta and colleagues [Amenta et al. 1998; Amenta and Bern 1998] formulated a sampling requirement for their algorithms. Our sampling is *minimal* w.r.t. this requirement, keeping sampling complexity low in flat regions far from the medial axis. Furthermore, we introduce several geometric processing operators, acting directly on the unions of medial balls representations: set union and other boolean operators, abstraction, and engraving. Our abstraction is similar to alpha shapes [Edelsbrunner and Mücke 1994] in that it produces a "tighter"

convex hull. However, unlike theirs, our method allows to control which concave corners to round off and produces a manifold output.

... "Stereolithography" is a method and apparatus for making solid objects by successively "printing" thin layers of a curable material ... one on top of the other.

Charles W. Hull, Inventor of 3D Printing

3

Additive Manufacturing: A Primer

Throughout this thesis, we exclusively use additive manufacturing (AM) technologies for the 3D fabrication of our models. While the above quote, taken from the seminal patent filed on this topic [Hull 1986], explains the essence of additive manufacturing well, we hereafter give the unfamiliar reader a brief introduction. We start with a comparison of additive manufacturing to traditional, subtractive

manufacturing (SM) in Section 3.1, followed by an overview of contemporary AM technologies in Section 3.2. Subsequently, we review multi-material printing in Section 3.3 and conclude with a discussion on printing assemblies in Section 3.4.

3.1 Additive vs. Subtractive Manufacturing

Similar to the way a 2D printer prints a document line-by-line, a 3D printer builds a given model layer-by-layer. As opposed to *adding materials*, traditional subtractive manufacturing such as milling, cutting, or drilling *remove materials*. AM has several advantages over traditional manufacturing techniques. While subtractive processes offer higher flexibility in the selection of end-use materials, they place severe limitations on the input geometry. Models with undercuts (areas where one part of the model overhangs another) cannot be fabricated using traditional manufacturing techniques. Almost all AM technologies overcome these geometric limitations by using a supporting structure that can be removed after printing. Such supporting structures can either be made of less densely printed *build material* – the material the final part is made from – or an additional *support material*.

3.2 Contemporary AM Technologies

We can categories current AM technologies according to type (how layers are deposited) and materials. Among the first and most prominent are extrusion-based techniques such as fused deposition modeling (FDM). FDM devices un-

wind filaments (mostly thermoplastics) from a coil to a heated extrusion nozzle which melts and drops the material onto the printing tray. They typically use less densely printed build material for the supporting structure.

A second category are formed by technologies that use granulated materials (e.g., plastic, metal powders, or plaster). These techniques build up a granular bed layer-by-layer, fusing the material powder at cross-sections of the rasterized 3D model. While lasers (selective laser sintering, SLS) or droplets of binding materials (inkjet 3D printing) are commonly used for the fusing, they all use the unfused material to support overhanging parts. A closely related category is laminated object manufacturing (LOM) where laser cutting is used to trace along contours of cross-sections at the top sheet of a stack of glued paper, plastic or metal laminates.

Currently highest resolution systems rely on photopolymerization and produce solid parts by curing a liquid resin using light activation. While stereolithography (SLA) – the oldest technology in this category – is similar to laser sintering in that it uses lasers to harden the resin in a layer-by-layer manner, the material bed is a fluid rather than a granulate.

3.3 Multi-Material Printing

While true multi-material printing is yet in its early stages [Lipson 2005; Alonso 2009], commercially available systems such as the Objet Connex series [Stratasys 2013] use blends of materials that are all based on such photopolymer resigns. Material is deposited from predefined mixtures of currently two liquids and solidified after deposition of each layer using UV light. The two sealed cartridges, holding

the liquids, can be replaced with other pairs, leading to materials with different properties. Objet's support material is gel-like and can be removed with a water-jet and their Connex 500 printer has a resolution of 600 DPI on the horizontal x and y axis, and 1600 DPI on the vertical z axis.

Despite the shared material base, we can currently print with subsets of over a 100 different materials with characteristics ranging from rubber- to plastic-like, within the same model and print job. While we cannot reproduce materials exactly, we can approximate, e.g., a given deformation behavior reasonably well as we will see in Chapter 5.

3.4 Printing Assemblies

While traditional manufacturing of models with movable parts commonly involves a manual assembly step, we can fabricate such assemblies in a single print job when using AM as we illustrate in Figure 3.4.1 with a hinge joint example. Although the tolerance between movable parts needs to be calibrated for each material and AM device, the design and manufacturing of highly complex assemblies such as our articulated characters in Chapter 6, is greatly facilitated.

Unlike with AM, such a hinge joint would need to be split into at least three parts for subtractive processes as we cannot directly assemble the final hinge from the upper and lower components.

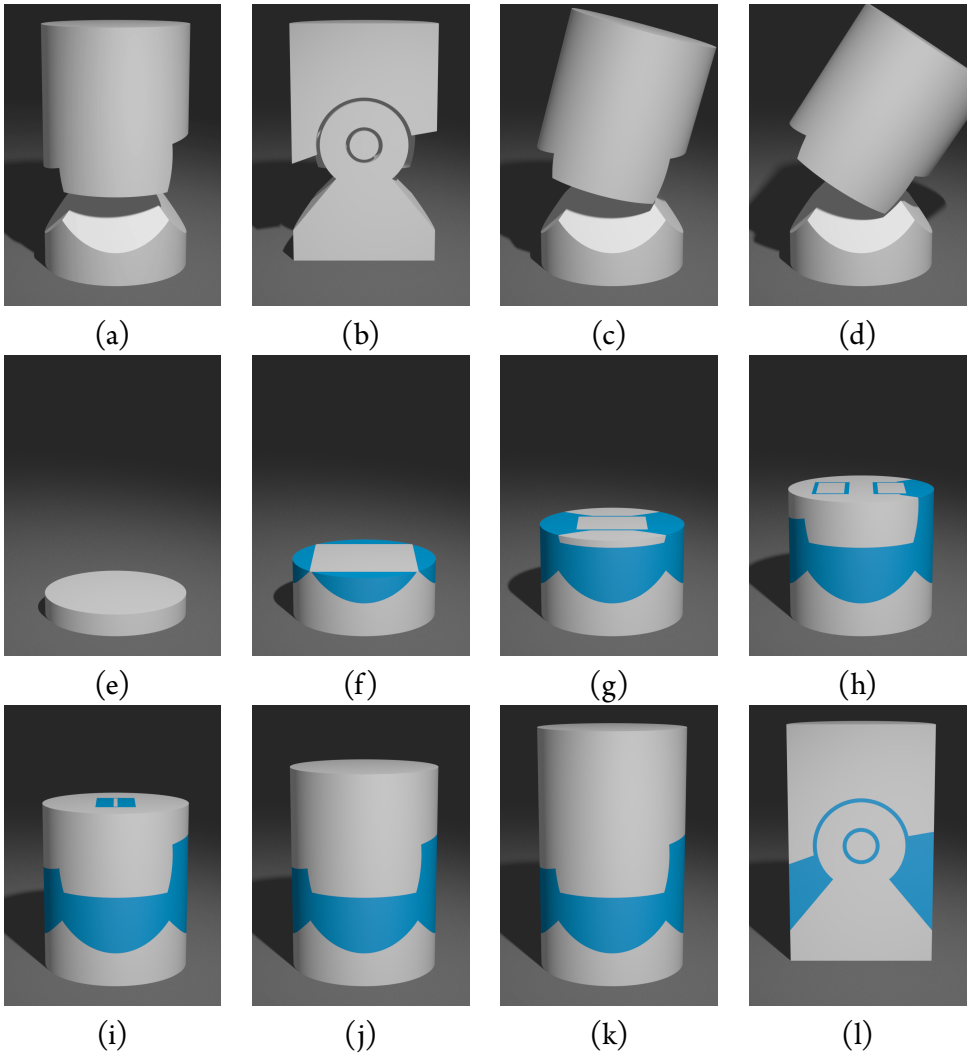


Figure 3.4.1: Printing Assemblies With AM, we can manufacture a hinge joint (a), consisting of two movable parts (b-d), without the need for manual assembly. We build the assembled model layer-by-layer (e-k), filling the model volume with build (in gray) and voids with support material (in blue). As long as we keep a minimal tolerance between movable parts, the support material can be removed, resulting in a fully functional mechanical hinge.

An algorithm must be seen to be believed.

Donald Knuth

4

Capturing Deformation Behavior

In this chapter, we introduce our data-driven representation and modeling technique for simulating non-linear, heterogeneous materials and soft tissue. Our approach simplifies the construction of convincing deformable models by avoiding complex selection and tuning of physical material parameters, yet retaining the richness of non-linear heterogeneous behavior (compare with Figure 4.0.1).

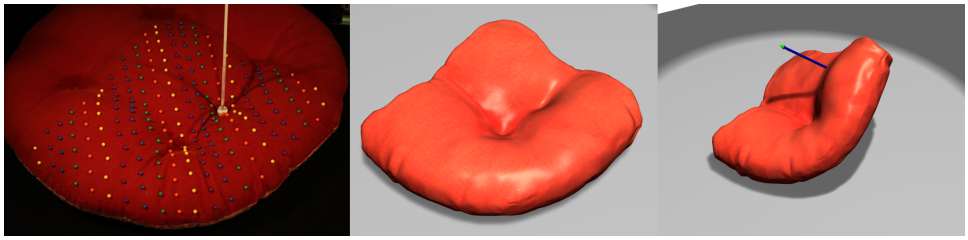


Figure 4.0.1: Modeling Deformation Behavior (from left to right): Force-and-deformation capture of a non-linear heterogeneous pillow; synthesized deformation with fitted material parameters; and interactive deformation synthesized with our data-driven modeling technique.

After further motivating the need for a data-driven material representation in Section 4.1, we formally introduce our deformable model in Section 4.2. Thereupon, we discuss how we estimate model parameters from a set of example deformations (Section 4.3), acquired using a hand-held measurement system (Section 4.4). We present the results of our approach for several non-linear materials and biological soft tissue, with accurate agreement of our model to the measured data in Section 4.5, and conclude with a discussion and summary (Sections 4.6 and 4.7). We will build upon the here presented data-driven material model in our physical reproduction and fabrication work, discussed in the next chapter.

4.1 Introduction

Recent years have witnessed significant progress of physically-based deformation models. Numerous researchers have combined Newtonian mechanics, continuum mechanics, numerical computation and computer graphics, providing a powerful toolkit for physically-based deformations and stunning simulations, with application in feature films, video games, and virtual surgery, among others.

However, achieving realistic deformations of a complex behavior requires careful choices for material models and their parameters. Many real-world objects consist of *heterogeneous materials*, requiring spatially-varying material parameters such as, e.g., Young's modulus and Poisson's ratio. Setting them is a difficult and time-consuming process. Even more challenging is the problem of *material non-linearities*. Most materials, for example rubber or biological soft tissue, show non-linear constitutive behavior, i.e., a non-linear relationship between stress and strain. Despite the wide variety of non-linear constitutive models in the literature, such as the popular hyperelastic Neo-Hookean and Mooney-Rivlin models [Ogden 1997], material modeling is still an active research area in material science. Nonetheless, non-linear physics equations are often simplified approximations to real material behavior, and choosing the appropriate model as well as tuning its parameters are extremely complex tasks.

Our technique employs finite element methods and exploits a set of measured example deformations of real-world objects, thereby avoiding complex selection of material parameters. Refer to Figure 4.1.1: we transfer every measured example deformation into a local element-wise strain space, and represent this example deformation as a locally linear sample of the material's stress-strain relation. We then model the full non-linear behavior by interpolating the material samples in strain space using radial basis functions (RBFs). Finally, a simple elastostatic finite-element simulation of the non-linearly interpolated material samples based on incremental loading allows for efficient computation of rich non-linear material simulations.

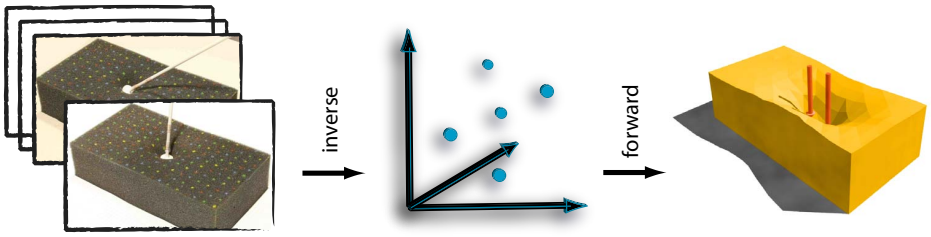


Figure 4.1.1: Acquiring and Modeling Non-Linear Quasi-Static Deformation Behavior (from left to right): An object is probed with a force sensor to acquire several example deformations, the applied force direction, and the force magnitude. For every measurement, we estimate its stress-strain relationship and represent it as a sample in strain space. During runtime, we interpolate these samples in strain space using radial basis functions (RBFs) to synthesize deformations for novel force inputs.

Earlier work in graphics and robotics also proposed acquisition-based model fitting as a means for obtaining deformable object representations [Pai et al. 2001; Lang et al. 2002; Schoner et al. 2004], but was limited to linear material models with global support. In contrast, our work is the first to represent *complex non-linear heterogeneous materials* through spatially-varying non-linear interpolation of local material properties. Together with our hand-held system for deformation capture from Section 4.4, our modeling pipeline is also distinct for its simplicity.

We present an efficient and robust algorithm for fitting the local strain-space material samples and demonstrate the effectiveness of our data-driven modeling method for several non-linear materials and biological soft tissue. The combination of simplicity and efficiency, both in acquisition and computation, and the high-expressiveness of the results make our technique applicable for interactive applications in computer graphics and other fields.

4.2 Modeling of Non-Linear Materials

In this section, we describe our representation of non-linear heterogeneous elastic materials, and how this representation is used for modeling an object's deformation behavior. We first give an overview of the representation, and then describe how we parameterize the materials and how this parameterization extends from the continuum setting to a finite element discretization. We also explain how we support material non-linearities through interpolation of local linear models, and finally we describe our algorithm for computing non-linear elastostatic deformations based on incremental loading.

4.2.1 Overview of our Approach

In materials science, (one-dimensional) elasticity properties have long been described through stress-strain curves. Inspired by this popular representation, we opt for modeling three-dimensional elastic properties by sampling the stress-strain function at various operating regimes and interpolating these samples in strain-space (see Figure 4.1.1).

More specifically, we characterize each sample of the stress-strain function using a (local) linear constitutive model. Then, in order to capture material non-linearity, we define the parameter values of the constitutive model at an arbitrary operating point through scattered-data interpolation in strain-space. Moreover, in order to capture material heterogeneity, we compute both the stress-strain samples and the scattered-data interpolation in a spatially-varying manner. Figure 4.2.1

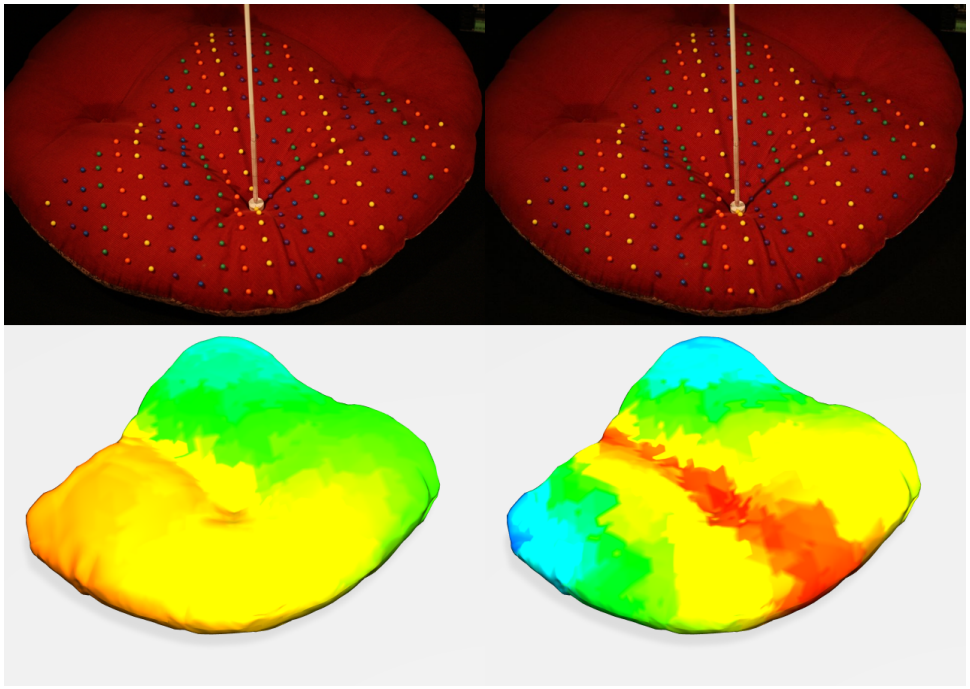


Figure 4.2.1: Spatially-Varying, Strain-Dependent Young's Modulus Two examples of a deformed pillow with color-coded Young's modulus ('blue' is low, 'red' is high), which varies both as a function of location and the local strain. Probe pressure was higher on the right.

shows example deformations with color-coded Young's modulus, which varies both as a function of the location and the local strain.

It is worth noting that our model can capture elasticity properties, but not plasticity or viscosity, among others. Our model builds on FEM and linear elasticity theory, and we refer the interested reader to books on the topic [Bathe 1995; Hughes 2000].

4.2.2 Discretization and Parameterization

We use linear co-rotational FEM to locally represent a deformable object's elastic properties. In other words, given an object's deformed configuration, we model the stress-strain relationship with linear FEM. We capture non-linearity by varying the parameters of the stress-strain relationship as a function of the strain itself. Given a displacement field $\mathbf{u}$, the linear co-rotational FEM employs Cauchy's linear strain tensor $\varepsilon(\mathbf{u}) = \frac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right)$. Invariance of the strain under rotations is obtained by extracting the rotational part of the deformation gradient through polar decomposition, and then warping the stiffness matrix [Müller and Gross 2004].

Because both, the strain and stress tensors, are symmetric, we can represent both as 6-vectors. Given the strain tensor, we construct the 6-vector as

$$\varepsilon = (\varepsilon_{xx}\ \varepsilon_{yy}\ \varepsilon_{zz}\ \varepsilon_{xy}\ \varepsilon_{xz}\ \varepsilon_{yz})^T, \tag{4.1}$$

and similarly for the stress. The local linear material yields then a relationship

$$\sigma(\mathbf{u}) = \mathbf{E}\varepsilon(\mathbf{u}) \tag{4.2}$$

between strain and stress. For each element (in our case, a tetrahedron), assuming locally linear isotropic material, the 6×6 stress-strain relationship matrix $\mathbf{E}$ can be represented by Young's modulus E and Possion's ratio ν

$$\mathbf{E} = \frac{E}{(1+\nu)(1-2\nu)}\left(\mathbf{G} + \nu\mathbf{H}\right), \tag{4.3}$$

with the two constant matrices

$$\mathbf{G} = \operatorname{diag}(1, 1, 1, 0.5, 0.5, 0.5) \tag{4.4}$$

and

$$\mathbf{H} = \begin{bmatrix} -1 & 1 & 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}. \tag{4.5}$$

This parametrization is intuitive, where Poisson's ratio ν is unit-less and describes material compressibility, while Young's modulus E defines material elasticity. However, we employ an alternative parameterization (λ, α) that allows us to describe the stress-strain relationship as a linear function of the parameters [Becker and Teschner 2007]:

$$\mathbf{E} = \lambda \mathbf{G} + \alpha \mathbf{H}, \tag{4.6}$$

with

$$\lambda = \frac{E}{(1+\nu)(1-2\nu)} \quad \text{and} \quad \alpha = \lambda \nu. \tag{4.7}$$

The parameter α is also known as Lamé's first parameter in elasticity theory, whereas λ is not directly related to any elasticity constant. With the (λ, α) parameterization, the stiffness matrix and the elastic forces become linear in the parameters. We exploit this property in our parameter fitting algorithm in Section 4.3.1.

The per-element stiffness matrix can be written as

$$\mathbf{K}_e = \lambda_e V_e \mathbf{B}_e^T \mathbf{G} \mathbf{B}_e + a_e V_e \mathbf{B}_e^T \mathbf{H} \mathbf{B}_e, \tag{4.8}$$

where V_e is the volume of the element (i.e., tetrahedron), and $\mathbf{B}_e$ is a matrix dependent on the initial position of the element's nodes. The complete stiffness matrix is obtained by assembling the warped per-element stiffness matrices $\mathbf{R}_e \mathbf{K}_e \mathbf{R}_e^T$, where $\mathbf{R}_e$ is the element's rotation. By grouping all material parameters $\{\lambda_e, a_e\}$ in one vector $\mathbf{p}$, the stiffness matrix is parameterized as $\mathbf{K}(\mathbf{p})$.

4.2.3 Strain-Space Interpolation

We describe the non-linear material properties through scattered-data interpolation of known local linear parameters in an element-wise manner. We obtain these known local parameters from a set of example deformations, largely simplifying an artist's job of tuning material parameters for complex non-linear constitutive models.

Let us assume a set of M known example measurements, each with a corresponding element-wise strain vector $\varepsilon_i \in \mathbb{R}^6$ and a parameter vector $\mathbf{p}_i = (\lambda_i, a_i)^T$. Recall that we use a rotationally-invariant strain by extracting the rotation of the deformation gradient through polar decomposition [Müller and Gross 2004]. Our non-linear strain-dependent material $\mathbf{p}(\varepsilon)$ is formed by interpolating linear material samples $\mathbf{p}_i(\varepsilon_i)$. At a given deformed configuration, the non-linear material is represented by the corresponding linear material that achieves the same force-displacement relationship. Note that we do not exploit linearization in the more

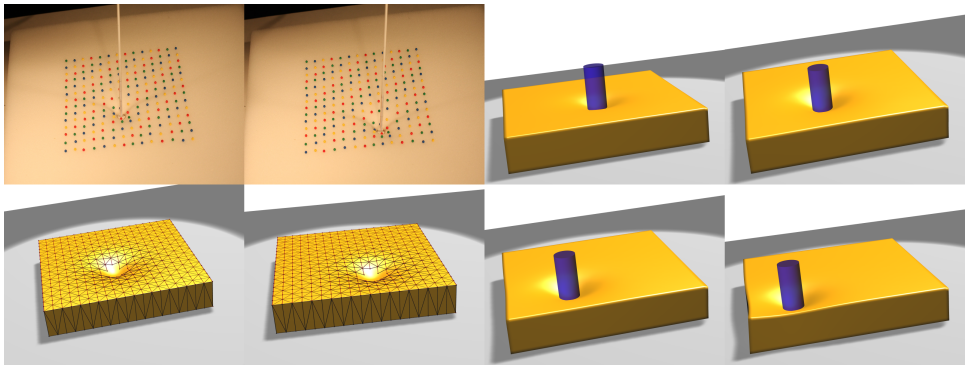


Figure 4.2.2: Captured and Synthesized Deformations (Foam) (two left-most columns): comparison of captured and synthesized deformations for a foam block. (two right-most columns): examples of interactive deformations produced by sliding a cylinder on top of the model.

traditional way of capturing the local slope of a non-linear function.

For each element, we define the stress-strain relationship through scattered-data interpolation in the strain-space $\mathbb{R}^6$ using radial basis functions (RBFs). The element-wise function describing the material, $\mathbf{p}(\varepsilon) : \mathbb{R}^6 \rightarrow \mathbb{R}^2$, has the form

$$\mathbf{p}(\varepsilon) = \sum_{i=1}^{M} \mathbf{w}_i \cdot \phi(||\varepsilon - \varepsilon_i||) , \tag{4.9}$$

where ϕ is a scalar basis function, and $\mathbf{w}_i \in \mathbb{R}^2$ and ε_i are the weight and feature vector for the i'th measurement, respectively. We employ the biharmonic RBF kernel $\phi(r) = r$. This globally supported kernel allows for smoother interpolation of sparsely scattered example poses than locally supported kernels, and avoids difficult tuning of the support radius [Carr et al. 2001].

As a preprocess, we compute the RBF weights $\mathbf{w}_i$. This reduces to solving $2T$ linear $M \times M$ systems for a deformable object with T elements due to the fact that

the stress-strain relationship is an element-wise description of the material. This also leads to scattered-data interpolation of the material parameters in a rather low-dimensional $\mathbb{R}^6$ domain. In contrast, interpolation of material properties is much more complicated in earlier approaches based on linear models with global support [Pai et al. 2001] due to the extremely high dimensionality of the parameterization.

4.2.4 Elastostatic FEM Simulation

We compute novel deformations using an elastostatic FEM formulation $\mathbf{Ku} = \mathbf{F}$, where the force $\mathbf{F}$ includes, among others, the load produced by a contact probe. To correctly capture the material's non-linearity during the deformation, we apply the load of the probe gradually, and solve the elastostatic FE problem for each load increment. In other words, at each loading step we measure the current strain ε, we compute the material parameters $\mathbf{p}(\varepsilon)$ by means of the interpolation described above, we formulate the elastostatic problem, and we solve it for the new deformations. The incremental loading procedure ensures that the non-linearity of the material is correctly captured during the complete deformation process, with the material parameters depending on the strain at all times.

For contact handling, we compute a distance field for the rigid probe object that produces the deformations. We test for collisions between points on the deformable object and the distance field and, upon collision, we compute the penetration depth and direction. We then define a linear force field at each colliding point and solve the FEM simulation through iterative quasi-static simulation. At

each iteration of the quasi-static FEM simulation, we first compute the material parameters for the current configuration based on the interpolation algorithm described above. Then, given the stiffness matrix and the linear collision force field, we define a quasi-static problem and solve for the new displacements. We compute several iterations until an equilibrium is reached.

4.3 Fitting the Material Parameters

We now describe how we compute the actual material parameters for a given object. This consists of two parts: first, estimating parameter values for each deformation example, and second, selecting a suitable basis from all the deformation examples.

4.3.1 Parameter Estimation Algorithm

In order to estimate a sample of the stress-strain relationship, we apply a known input force to the object under study. For each captured deformation we can distinguish three different regions on the object's surface: (i) the probing region, with measured non-zero forces and measured displacements, (ii) the attached region, with unknown forces and zero displacements, and (iii) the free region, with zero forces and measured displacements. We use $\bar{\mathbf{x}}$ and $\bar{\mathbf{F}}$ to denote the vectors of known displacements and forces, respectively, at the points corresponding to mesh nodes in the model.

Given measured displacements and forces, we compute spatially varying mate-

rial parameters $\mathbf{p}$ as:

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} \left\{ \sum_{i=1}^{n} ||\mathbf{x}_i(\mathbf{p}, \bar{\mathbf{F}}) - \bar{\mathbf{x}}_i||^2 + \gamma ||\mathbf{L}\mathbf{p}||^2 \right\}, \tag{4.10}$$

where $\mathbf{x}_i(\mathbf{p}, \bar{\mathbf{F}})$ denotes the position of a mesh node as a function of material parameters and the measured forces. The sparse Laplacian matrix $\mathbf{L}$ enforces spatial smoothness of parameters. We employ the umbrella operator [Zhang 2004] $(\mathbf{L}\mathbf{p})_i = \sum_j w_{i,j}(\mathbf{p}_i - \mathbf{p}_j)$, where i and j refer to tetrahedron labels, and $w_{i,j} = 1$ iff two tetrahedra share a vertex. This regularization is required to prevent overfitting due to noise in the acquired data. This is also mathematically required to obtain a well-posed problem because the number of parameters is always twice the number of tetrahedra, $|\mathbf{p}| = 2T$, whereas the number of measured positions $|\bar{\mathbf{x}}| = n$ may be smaller, which would result in an underconstrained problem. We also considered scattered data interpolation of material parameters in object space as an alternative for addressing the underconstrained problem, but it would be difficult to decide where to place the samples for highly heterogeneous objects.

We use the Levenberg-Marquardt algorithm [Levenberg 1944] to iteratively minimize the non-linear residual Equation 4.10. We derive the Jacobian matrix in the Appendix A. Instead of defining the residual in terms of measured positions, the error functional could also be described in terms of measured forces [Becker and Teschner 2007], yielding a linear optimization problem. However, our observations have shown that this approach is unstable when the force-displacement relationship is not close to linear material behavior.

4.3.2 Strain-Space Basis Selection

A material capture session consists of capturing N example deformations, from which we obtain the training dataset of N parameter vectors for each element in the mesh. However, this dataset may be rather large, and we are interested in selecting a compact set of M basis parameter vectors for each element. Note that M need not be the same for all elements.

We select the basis in the same greedy manner as proposed by Carr et al. [2001]. We start by setting a parameter vector at zero strain with the average parameters computed for very small-strain deformations. We then add the parameter vector with largest error, until a given error tolerance is achieved. After each parameter vector is added to the basis, we need to compute the RBF weights that best fit the parameter vectors for all N example deformations in a least-squares manner, as described in Equation 4.9. The inset figure on the right shows the evolution of the fitting error for the foam block in Figure 4.2.2.

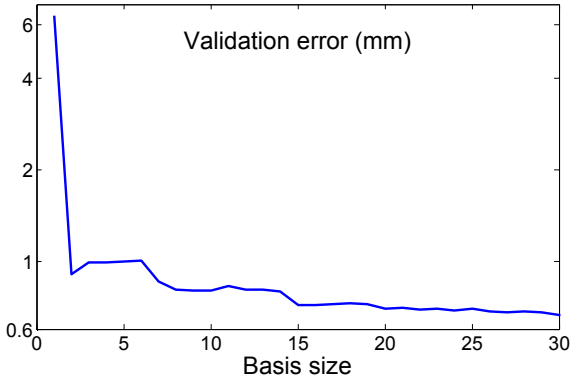
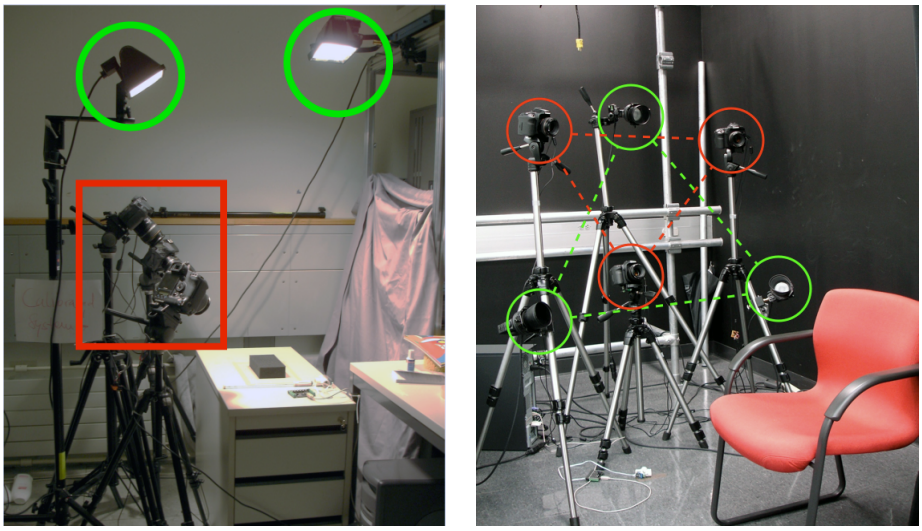


Figure 4.3.1: Evolution of Fitting Error

This error plot accumulates the error for all captured deformations, not only those added to the basis. The error drops quickly after adding the second parameter vector to the basis because the first vector may not represent the average material behavior well. We will provide more details on the validation of our method in the next section.



(a) (b)

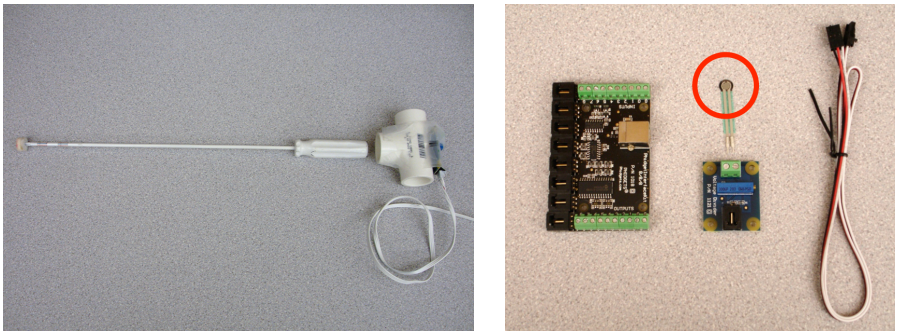
Figure 4.4.1: Trinocular Stereo Vision System Our trinocular stereo vision system consists of three high-resolution cameras (indicated in red) and two to three light sources (indicated in green). The cameras are arranged in a triangular setup, which helps maximize visibility during capture of a contact interaction. The light sources ensure uniform illumination during the acquisition.

4.4 Data Acquisition

We developed a simple data acquisition system consisting of force probes and a marker-based trinocular stereo system. Deformations are induced by physical interaction with the object. We decided to use a marker-based system due to its simplicity, robustness, and independence of the object's surface properties.

4.4.1 Trinocular Stereo Vision System

Figure 4.4.1 shows our trinocular stereo vision system, consisting of three Canon 40D cameras that capture images at a resolution of 3888×2592. These cameras are



(a) (b)

Figure 4.4.2: Contact Probe (a) Contact probe with integrated force sensor. (b) From left to right: USB Interface Kit, Force Sensing Resistor (red circle), Phidget Voltage Divider, and connection cable.

placed in a triangular configuration to minimize occlusions caused by the contact probes during data acquisition. We built an external trigger device to synchronize the three cameras, and use additional light sources to ensure uniform illumination during the acquisition process. The surface displacement during static deformations is measured using a set of markers that we paint on the object's visible surface. Our system is capable of measuring viewpoint-registered marker positions to an accuracy of < 1 mm.

We built contact probes with arbitrary shapes and circular disks of different diameters attached to the tip of a long screwdriver (see Figure 4.4.2). We estimate the position and orientation of the contact probe using two markers on the white shaft of the screwdriver. To measure the magnitude of the contact forces we use a 0.2 inch Force Sensing Resistor (FSR) (Item S-20-1000-FS2) connected to a Phidget Voltage Divider (Item S-50-P1121) and USB Interface Kit 8/8/8 (Item

C-200-P1018) by Trossen Robotics. The force sensor's read operation is synchronized with the external camera trigger signal.

4.4.2 Estimating Displacements

To process the acquired data, we first identify corresponding marker positions in the captured images, then reconstruct their 3D locations. Thereafter, we register the markers to a template mesh, recovering positions lost due to occlusions also.

For the extraction of markers from the three sets of frames, we use standard image processing: per-color thresholding in the CIELab space, followed by morphological closing. 2D marker positions are then estimated by averaging locations of pixels belonging to 8-connected components in the resulting binary images. This procedure, even though simple, allows for a robust extraction of markers.

To reconstruct our markers in 3D, we rely on accurate depth and correspondence estimation. To this end, we calibrate our trinocular system using Bouguet's toolbox [2006] and automatically establish marker correspondences within and across the three different views using proximity measures.

Given a template mesh such as, e.g., a face scan, we register the 3D marker locations using a quaternion-based formulation [Micheals and Boult 2000] of Horn's shape matching algorithm [1987]. While we avoid almost all marker occlusions with our three-view system, we employ a linear shell-based formulation [Bickel et al. 2007] with prescribed displacements (visible markers) as boundary constraints, minimizing surface stretching and bending to estimate displacements of occluded markers.

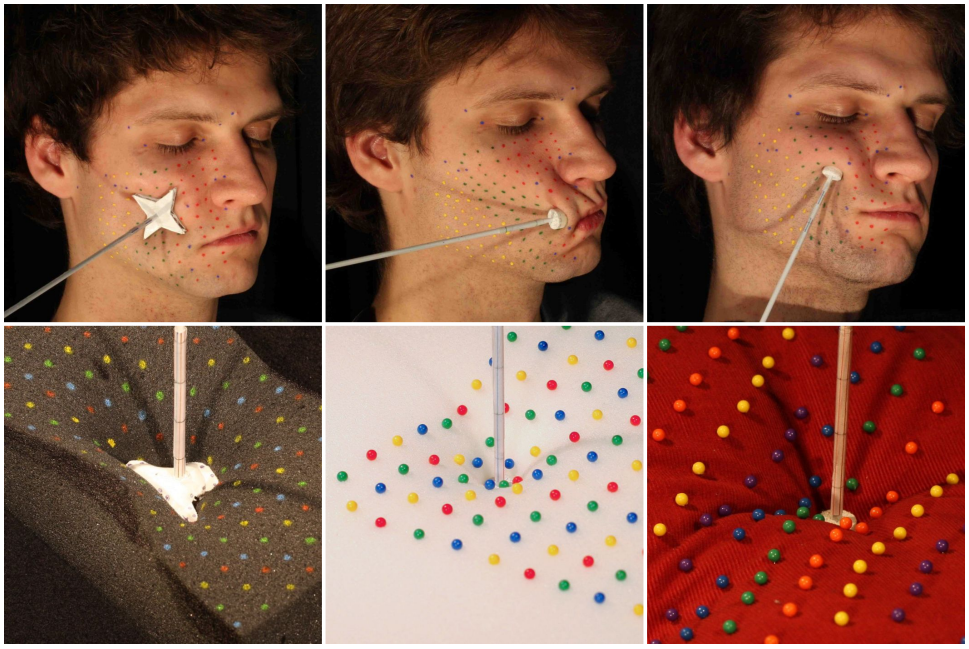


Figure 4.4.3: Sampling Deformation Behavior Our hand-held system enables the non-invasive acquisition of force-displacement samples of facial tissue (top) and physical objects (bottom).

Refer to Figure 4.4.3: Our system enables the non-invasive sampling of deformation behaviors of a wide range of physical objects (bottom) and facial tissue (top). Moreover, our hand-held probes facilitate the capture at arbitrary locations, varying angles, and with custom contact shapes.

4.5 Results

Model Evaluation: We have evaluated the quality of our material capture and modeling technique on several real-world objects, including two foam blocks, a

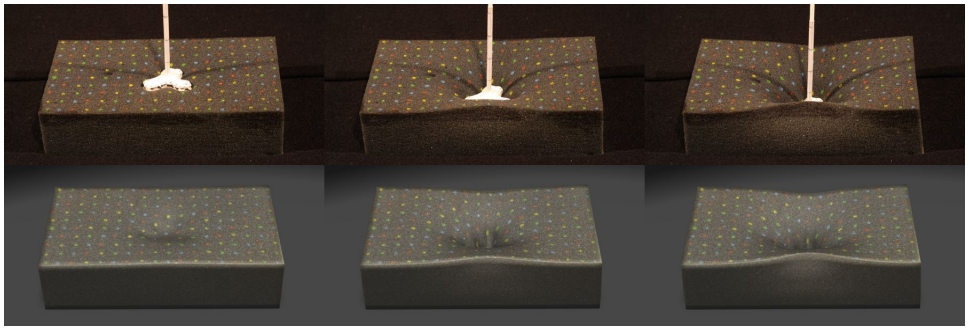


Figure 4.5.1: Modeled vs. Real Deformations Comparing real (top) and modeled (bottom) deformations with a different contact probe than the one used during the data acquisition phase.

heterogeneous soft pillow, and a human face.

Figure 4.2.2 shows a foam block with homogeneous material. We acquired 48 deformation examples, well distributed over the foam to induce deformations in all 1, 805 tetrahedra of our model. We then constructed the non-linear material representation, with bases of 8 samples per tetrahedron on average, using the procedure in Section 4.3.1. Even though the object is homogeneous, it should be noted that the material parameters that were estimated for each input example are non-homogeneous due to non-linearities in the stress-strain relationship. The average fitting error for the captured deformations is less than 1 mm (see inset figure in Section 4.3.2). Figure 4.2.2 shows synthesized deformations produced with our technique using a probe with a larger, different contact area than the probe used for data acquisition.

To compare our model to a uniform linear co-rotational model we use the homogeneous foam shown in Figures 4.5.1 and 4.5.2. We captured 12 deformation examples with the probe near the center of the block and modeled the object with

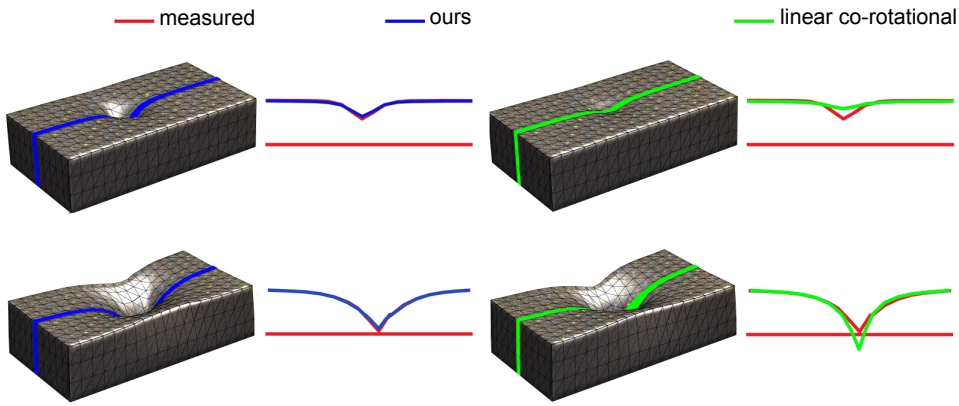


Figure 4.5.2: Linear Co-Rotational vs. Our Method Comparison of deformations using our method vs. an average-fit linear co-rotational model.

3, 240 tetrahedra. We computed an average-fit linear co-rotational model that best approximates all the input deformations. As shown in Figure 4.5.2, our model (blue) accurately captures the hyperelastic behavior of the foam, while the average-fit linear co-rotational model (green) underestimates the deformation at small force values and overestimates it at large ones. In addition, the linear co-rotational model suffers from element inversion for large forces.

Our model is of course not confined to the contact shapes that were used during data acquisition. Figure 4.5.1 shows a side-by-side comparison of our model (bottom) to real deformations (top) using a different contact probe than the circular one we used for data acquisition. We captured the applied force with the new contact probe, and then distribute it uniformly in the simulated setting. The figure shows high correspondence between the real and simulated scenarios. We refer the reader to the accompanying video for an animated side-by-side comparison.

To evaluate the sensitivity of our capture and modeling approach to measurement noise we created example deformations of a virtual block with three layers of user-defined non-linear materials (see inset figure). We then evaluated the accuracy in matching these deformations with our model under different levels of noise in the input data. Specifically, we applied Gaussian noise with a variance of 10%, 20% and 30% to the input displacements and then measured the L^2 error for all deformations and error levels. On average, we obtain an error of 0.3% of the maximum displacement for the case without error, and 2.1%, 3.1% and 4.4% for the cases with 10%, 20% and 30% input noise, respectively.

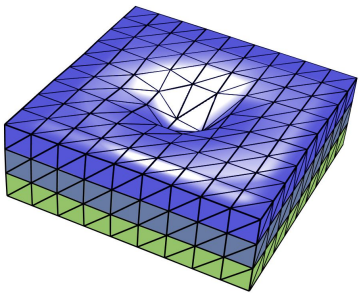


Figure 4.5.3: Virtual Block

Figure 4.5.4 shows a pillow object with heterogeneous behavior even in its rest state. The screenshots compare the captured deformations with the deformations of the 1,691 tetrahedra model synthesized with our algorithm. The figure also shows screenshots of deformations at interactive frame rates of about 10 Hz on a standard PC.

Facial Deformation: We have also applied our data-driven capture and modeling technique to the challenging task of facial deformations, as shown in Figure 4.5.5. We have modeled the facial tissue with a single layer of 8,261 tetrahedra that are attached to a low-resolution skull model. To model the sliding contacts between the tissue and the skull we use the same contact handling as for the probe object (see Section 4.2.4). Given the deformation of the tetrahedral mesh, we

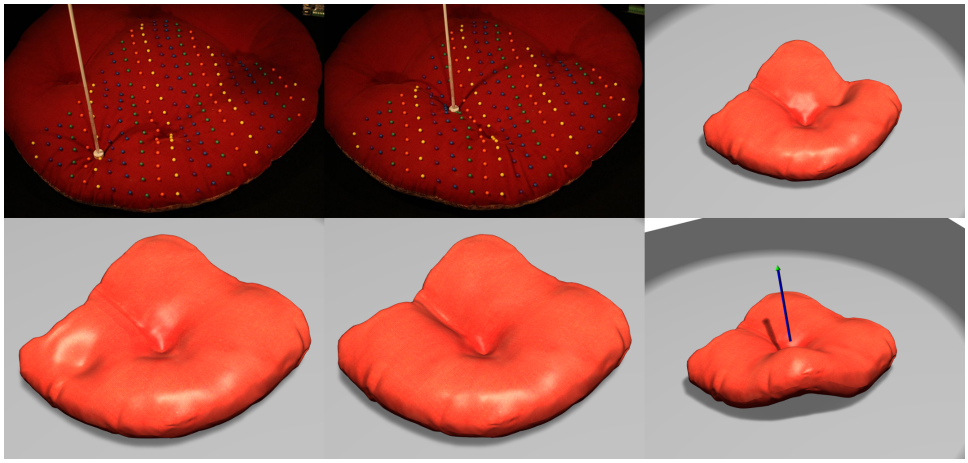


Figure 4.5.4: Captured and Synthesized Deformations (Pillow) (two left-most columns): comparisons of captured and synthesized deformations for a heterogeneous non-linear pillow. (right column): interactive deformations of the model produced by pushing (top) and pulling (bottom).

compute the deformation of a high-resolution triangle mesh using a smooth embedding based on moving least squares interpolation like Kaufmann et al. [2008].

Note that our face model does not correctly capture all types of deformations because we use a model with closed lips, and all the deformation examples in the training dataset were captured with relaxed muscles and closed jaw. Nevertheless, the model is able to produce compelling deformations even without anatomically correct modeling of the musculoskeletal structure of the face.

4.6 Limitations and Future Directions

Our work suggests a highly innovative approach to non-linear material modeling, but it also suffers from limitations. Due to its formulation, our technique is currently limited to capturing elastic properties. A fully dynamic simulation of

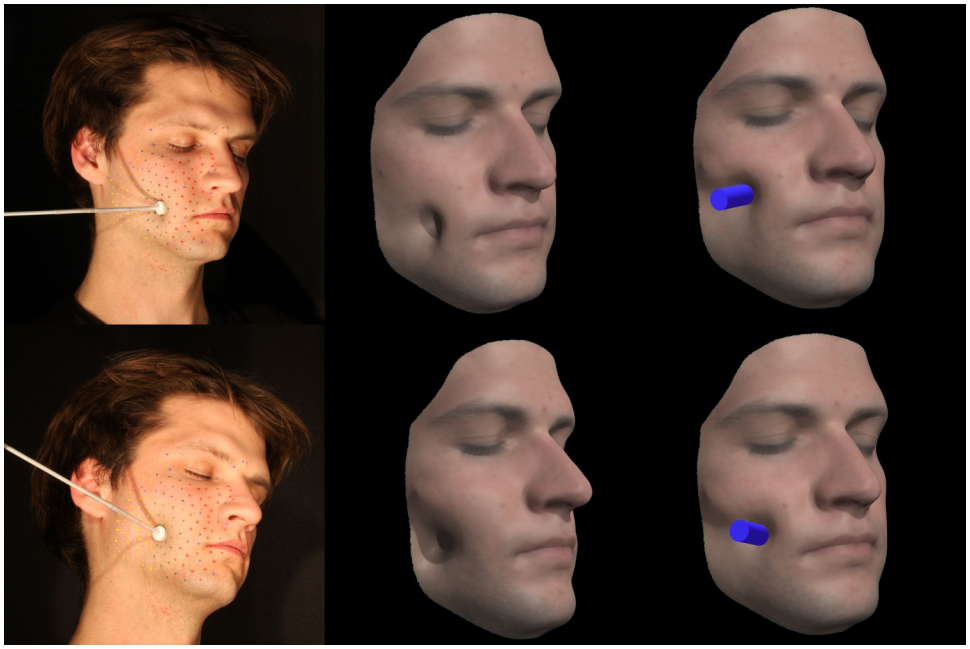


Figure 4.5.5: Captured and Synthesized Deformations (Face) (left): capture of facial deformations; (middle): synthesized deformations for the captured examples; (right): frames of an animation with a cylindrical probe pressing on the cheek.

materials would require capturing other properties such as viscosity and plasticity. One interesting conclusion of our work is that it is often possible to obtain compelling surface deformations with a volumetric meshing unaware of an object's actual volumetric structure. This is of course not valid for all situations. For example, our face model could be greatly enhanced with accurate lip contact and jaw motion models.

There are several aspects of our model that deserve further exploration. One of them is its ability for capturing anisotropic behavior. The underlying linear co-rotational material model that we use for representing deformation samples can only capture isotropic behavior, but deformation samples with the same total strain but in different directions will lead to anisotropic behavior. In other words, we locally model the material isotropic in strain space, yet strain-space interpolation of material parameters provides global anisotropic behavior. It is worth exploring to what extent our approach captures anisotropy.

Another aspect that deserves further analysis is the formulation of the quasi-static deformation problem. Given a certain strain, we employ a local linear co-rotational model to formulate a quasi-static deformation problem. However, our model is not strictly a local linearization, which means that the stiffness matrix of the quasi-static deformation problem does not employ correct force derivatives. At the same time, our linear model is more robust than a model obtained by local differentiation and avoids non-passive regimes.

Similar to other approaches, our parameter fitting algorithm is formulated as a minimization problem and may end up in a local minimum. In fact, we have iden-

tified fitting error as the major source of potential inaccuracies in the deformation synthesis. Sometimes, fitting error also appears because we limit Poisson's ratio to physically valid values during the minimization. Robust parameter identification is still an open research problem in material science, and some recent approaches explore alternative solutions including particle filters [Burion et al. 2008]. Multi-resolution fitting may be another way of increasing robustness.

Finally, using a more efficient parameter estimation algorithm for material fitting, one could evaluate the need for further samples of the stress-strain relationship online, and determine the optimal probing patterns on the fly.

4.7 Summary

We have presented a novel data-driven method for modeling non-linear heterogeneous materials. The major practical contribution of our work is the ability to model rich non-linear deformations in a very simple manner, without the complex task of carefully choosing material models and parameters. Instead, our data-driven method relies on a simple-to-build acquisition system (see Section 4.4), a novel representation of the material through spatially-varying interpolation of fitted linear models, and a simple deformation synthesis method.

In the next chapter, we will use our data-driven deformation model to represent our homogeneous base materials, a layered combination of which allow us to reproduce and fabricate a desired deformation behavior. Assuming homogeneity, we significantly increase the robustness of our fitting process, by fitting a single non-linear model to all acquired force-displacement pairs.

If you come to a fork in a road, take it.

Yogi Berra

5

Fabricating Deformation Behavior

In this chapter, we introduce a data-driven process for fabricating a desired deformation behavior using multi-material AM devices. Our process takes example deformations, either acquired using an automated measurement systems, or sampled from a virtual deformation simulator, as input. Given the input, we then estimate an approximate model consisting of a layered set of base materials, ready for

Figure 5.0.1: Physically Replicating Deformation Behavior Given the deformation behavior of real world objects in form of measured example deformations, we estimate layered approximate models, tailored for 3D manufacturing using multi-material AM technologies. Our replicas' deformation properties are in high agreement with those of the input.

3D printing (see Figure 5.0.1). To represent these base materials, we adopt our data-driven deformable model from the previous chapter.

Automating the fabrication of virtual and real deformation behavior is of high practical relevance as we point out in Section 5.1. We then describe the adjustments to our data-driven model to represent our homogeneous base materials. The adjusted model has fewer parameters than the original (Chapter 4), leading to an increase in fitting robustness (Section 5.4). Thereafter, we introduce our optimization that identifies the best combination of stacked layers of base materials in Section 5.5. We demonstrate our complete process by physically replicating complex heterogeneous materials in Section 5.8.

5.1 Introduction

Elastically deforming objects are omnipresent in our everyday live (e.g, our shoes, or chair cushions) and widely used in physics-based animation to increase realism. Yet, we lack algorithms for their automated fabrication using multi-material AM

devices. As mentioned in Chapter 3, these devices are capable of manufacturing a variety of soft and hard materials with complex internal structures, making it possible to fabricate complex 3D objects with aggregate materials quickly, inexpensively, and with high accuracy. Despite these technical advances, we do not have tools at our disposal that aid us with the design of such multi-material content.

We present a goal-based design process that, provided with a set of example deformations, physically reproduces the sampled behavior using a layered approximate model and a multi-material AM device (compare with Figure 5.1.1). For validation purposes, we compare probes of real world objects and their replica, measured with our automated acquisition system (Section 5.6). However, we are by no means restricted to define a desired behavior using acquired samples. Probes can also be taken from simulations of deformable models, enabling the fabrication of digital content and, hence, providing us with a design interface.

5.2 Overview

We have collected a database of base materials, fabricated using a Connex 500 multi-material 3D printer, but also a variety of standard foams, gels, and rubbers purchased from the McMaster-Carr catalogue. These materials span a wide gamut of different deformations: from very soft to very hard and rigid (see Section 5.8). We automatically measure deformations of these base materials subject to different forces using our robotic system (Section 5.6).

Next, we represent our base materials using a data-driven non-linear stress-strain

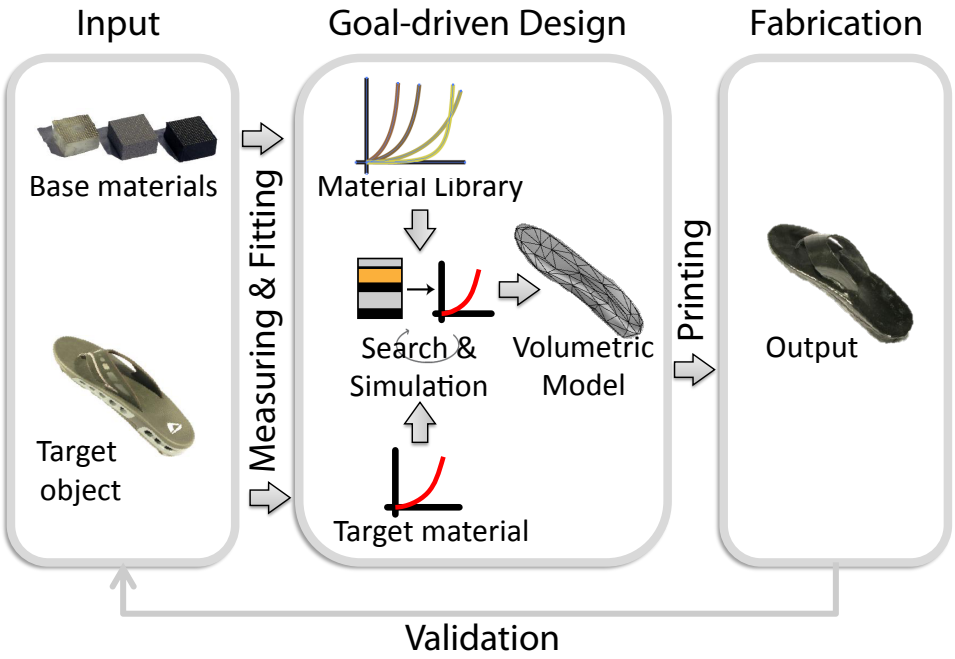


Figure 5.1.1: Goal-Driven Design of Deformable Materials

relationship in a Finite Element Method (FEM) (Sections 5.3 and 5.4). This compact representation allows us to predict deformations of thicker or thinner versions of the base material samples. More importantly, we show that we can accurately predict deformations of arbitrary combinations of stacked base materials.

As the last step, we design composite materials that best match a desired deformation behavior using our combinatorial optimization algorithm (Section 5.5). In order to simplify the material design process, we introduce a goal-based optimization approach. The user specifies a material by providing example deformations and their corresponding forces, and our algorithm automatically computes the best-matching composite material. Because the configuration space is combinatorial and exponentially large, we use an efficient search strategy that prunes

away states that yield poor matches to the desired material specifications.

We validate the simulation and material model by fabricating a number of different composite materials, measuring their deformations subject to a variety of different forces and comparing these measurements to the results of the simulation. We describe the results in Section 5.8.

5.3 Non-Linear Material Model

All our base materials exhibit a non-linear hyper-elastic stress-strain behaviour, as demonstrated by the measured force-displacement curves in Figure 5.7.1. Most of our base materials consist of complex structures which influences the deformation behavior significantly (see Figure 5.7.3). We use our data-driven approach from Chapter 4 to represent such non-linear behavior. Recall that our model captures such non-linearities by a non-linear interpolation of locally linear material properties. We obtain these linear properties from example deformations, probed from base materials in this context.

For linear materials, Hooke's generalized law

$$\sigma(\mathbf{u}) = \mathbf{E}\varepsilon(\mathbf{u}) \tag{5.1}$$

describes the relation between strain and stress with a 6×6 material-dependent matrix $\mathbf{E}$.

The key for achieving the non-linear behavior of our base materials is to define the matrix $\mathbf{E}$ as a function of local strain $\varepsilon(\mathbf{u})$. And, because we can typically pa-

rameterize $\mathbf{E}$ with fewer parameters $\mathbf{p}$, we define these parameters to non-linearly depend on $\varepsilon(\mathbf{u})$, resulting in a non-linear material representation $\mathbf{E}(\mathbf{p}(\mathbf{u}))$. Although this approach holds for general anisotropic behaviour, we describe subsequently the parameters $\mathbf{p}$ for two types of materials that are most relevant in practice: isotropic and transversely isotropic ones. We defer the discussion of transversely isotropic material to Section 5.7.

5.3.1 Isotropic Materials

For homogeneous linear isotropic materials, the matrix $\mathbf{E}$ can be represented by the two Lamé parameters λ and μ, hence, we have $\mathbf{p} = (\lambda, \mu)$. Using Lamé's parameters, the stress and strain tensors are related as

$$\sigma(\mathbf{u}) = 2\mu\varepsilon(\mathbf{u}) + \lambda tr(\varepsilon(\mathbf{u}))\mathbf{I}, \tag{5.2}$$

from which the matrix $\mathbf{E}$ can be derived [Bathe 1995].

In homogeneous linear isotropic materials, the strain is well captured by the three invariants of the symmetric strain tensor $I_1(\varepsilon), I_2(\varepsilon), I_3(\varepsilon)$. These invariants do not change under rotation of the coordinate system. Using the invariants to represent the strain, our non-linear material model can be considered as a 2-valued function in a 3-dimensional domain, $\mathbf{p}(I_1, I_2, I_3) : \mathbb{R}^3 \rightarrow \mathbb{R}^2$.

5.3.2 Non-Linear Interpolation of Material Properties

Given a base material, we describe its non-linear stress-strain relation through a small set of P parameter vectors, $\{p_i\}$, corresponding to different strain values,

$\{\varepsilon_i\}$. Then, using the (parameter, strain) pairs as centers of Radial Basis Functions (RBF), we define the complete material behavior through RBF interpolation (see Chapter 4):

$$\mathbf{p}(\varepsilon) = \sum_{i=1}^{M} \mathbf{w}_i \cdot \phi(||\varepsilon - \varepsilon_i||) , \tag{5.3}$$

Since our base materials are homogeneous, a single set of parameter vectors is sufficient to describe the behavior of an arbitrary object consisting of a single base material. This reduces the number of parameters of a base material to $|\mathbf{p}| \cdot P$, where $|\mathbf{p}|$ is the cardinality of the parameter vector (2 for isotropic materials, and 5 for transversely isotropic ones). In our examples, the number of RBF centers is typically between $P = 6$ for the isotropic foams and $P = 12$ for printed materials with complex internal microstructure. Computing the RBF interpolation based on the local strain in a spatially-varying manner allows us to simulate different non-linear behavior in different regions of an object.

In order to simulate the behavior of composite objects made of base materials, we follow the quasi-static FEM approach described earlier in Section 4.2.4: given a simulation state, we compute the strain of all elements and perform a per-element computation of the parameter vector according to 5.3. We then recompute the per-element stiffness matrices, and perform a new step of the FEM simulation.

5.4 Fitting Base Materials

We estimate the properties of base materials such that simulated deformations match best a set of input examples. In our fitting process, we compute the RBF cen-

ters $\{\varepsilon_i\}$ (i.e., strain values used as data points), and their corresponding weights $\mathbf{w}_i$ (see Equation 5.3). Assuming the P RBF centers to be known, we compute the RBF weights $\mathbf{w}$ as follows: given a set of example deformations of measured displacements $\{\bar{\mathbf{x}}_i\}$ and corresponding forces $\bar{\mathbf{F}}_i$, we minimize the error in the displacement using

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left\{ \sum_{i=1}^{n} ||\mathbf{x}_i(\mathbf{p}, \bar{\mathbf{F}}_i) - \bar{\mathbf{x}}_i||^2 \right\}. \tag{5.4}$$

To define the RBF centers, we first fit a homogeneous linear material to obtain a constant set of material parameters. Using these parameters, we run FEM simulations for all measurements, and record strain values. We select the RBF centers by sampling the strain space with P points that cover the range of measured values well. Using these RBF centers, we can fit the material parameters but run several iterations to obtain a better coverage of the strain space.

There are two main differences between our material fitting strategy and the one proposed earlier in Section 4.3.1. First, since the base materials are homogeneous, the RBF weights are not spatially-varying, and the size of the problem reduces to $|\mathbf{p}| \cdot P$. Second, the objective function is defined by grouping the measured displacements of all example deformations at once. These two differences lead to improved robustness and fitting accuracy.

As before, we use Levenberg-Marquardt optimization and compute the Jacobians as described in Appendix A. However, an unconstrained minimization may lead to material parameters not physically feasible, causing instabilities during FEM

simulations. In case of isotropic materials, we bound Lamé's parameters by computing the corresponding Young's modulus and Poisson's ratio, then projecting them to physically feasible ones. For transversely isotropic materials, we ensure that the stiffness matrix stays positive definite using the technique by Rebonato and Jäckel [1999].

Our measured forces $\bar{\mathbf{F}}$ are normal to the surface. However, the contact area below the force probe also undergoes small tangential forces during acquisition, and we found that these missing forces cause fitting errors. Hence, we compute these missing tangential forces by constraining probed surface points to fall together. We then reintroduce these tangential forces as known forces, leading to increased quality of our fitting. We evaluate our fitting by reporting errors when comparing simulated base materials to measurements in Section 5.8.

5.5 Goal-Driven Material Design

Our goal-based material design approach approximates a desired deformation behavior with a composite of base materials. We now describe the optimization algorithm to obtain composite structures made of a set of base materials. Our algorithm receives as input a description of the object surface, examples of desired force-displacement pairs, and a set of base materials with known deformation properties, expressed with our non-linear material model.

5.5.1 Problem Statement

We formulate the design process as an optimization problem where we need to choose the distribution of M possible base materials inside the fabricated object such that it matches the input force-displacement samples.

We discretize the problem by dividing the desired object shape in a set of N regular *cells*, each made of a uniform base material. The desired inhomogeneity and possible anisotropy of the final object are achieved by the appropriate distribution of base materials. For each cell, one may choose a single material from M possible base materials. We call a certain choice of base materials and their distribution a *design*. We denote each design as a vector $\mathbf{m} = (m_1, m_2, \ldots m_N)$, where m_i is an integer value that indicates the type of base material in the i^{th} cell out of the $\{\bar{m}_j, 1 \leq j \leq M\}$ possible base materials.

In order to test each design, we assign its particular material choices to the cells of the object, simulate the object with the quasi-static FEM approach from Section 5.3 using the user-specified force profiles, and measure the error in surface displacements. The surface displacements of all input examples are grouped in one large vector $\mathbf{x}$. Given the targeted displacements $\bar{\mathbf{x}}$, the *displacement error* of a design is simply $\|\mathbf{x} - \bar{\mathbf{x}}\|$. Finding the optimal design with minimal displacement error is an exponential problem, with M^N to-be-tested designs.

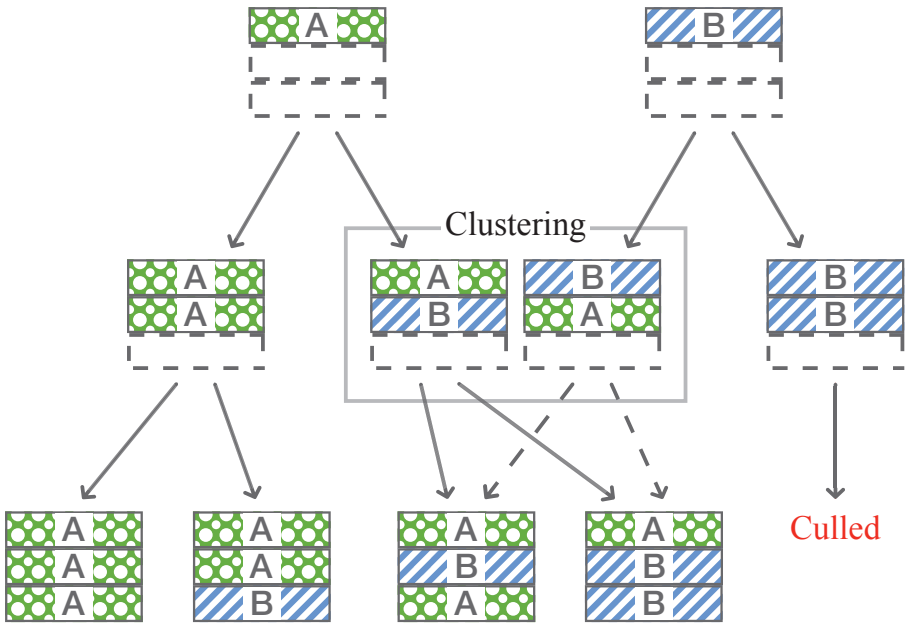


Figure 5.5.1: Branch-and-Bound with Clustering The root of the tree shows the two materials A and B for the first out of three cells. Each level of the tree spans the possible options for the subsequent cells. Sub-optimal branches of the tree can be *culled*, and similar deformations can be *clustered*.

5.5.2 Branch-and-Bound with Clustering

The major problem when solving such a design optimization is the non-convexity of the design space and therefore the risk of ending up with a locally optimal solution if only the local neighborhood is taken into account [Lund and Stegmann 2005]. To solve this discrete optimization problem we use a decision tree such that at each level of the tree we span the options for one cell in the design. The root of the tree has M children, where each child represents one of the material choices for the first cell, while the other $N - 1$ remain undecided. Figure 5.5.1 shows the decision tree for an object with three cells and two possible material choices.

Entire branches of the decision tree can be culled away using a branch-and-bound algorithm [Land and Doig 1960]. During tree traversal, we store the minimum error d_{min} for the designs tested so far. When a new node of the tree is visited, i.e., a new cell is refined, we use this minimum error to cull (if possible) the complete subtree rooted at the node.

Given the breadth of the tree, branch culling still leads to an intractable number of possible designs. However, often several designs produce similar deformation results. Hence, we cluster these nodes together to limit the breadth of the decision tree at every level.

5.5.3 Bound Estimation

We define $\{\mathbf{m}\}_a = (m_1, m_2, \ldots m_l, x \ldots x)$ to denote the designs rooted at a node a and located at level l. The first l cells are already determined along this branch, while the rest are still undecided (denoted by x). We estimate a bound

on the deformations produced by $\{\mathbf{m}\}_a$ by considering the cases where the undecided cells are uniform.

In other words, we estimate bounds by filling the undecided cells with each base material $\bar{m}_j$, computing the resulting deformation for all input examples $\mathbf{x}_j$, and bounding the result of the M cases as $[\mathbf{x}_j]$. We use axis-aligned bounding boxes in high dimensions as bounds, i.e., maximum and minimum values for each dimension of the resulting displacement vectors. We cull the branch rooted at node a if $dist(\bar{\mathbf{x}}, [\mathbf{x}_i]) < d_{min}$. When new designs are tested we update d_{min} appropriately.

Due to material non-linearities and the existence of non-monotonic functions along the simulation process, our bound estimation is not conservative. While efficient methods for bounding displacements in linear FEM settings exist [Neumaier and Pownuk 2007], practical bounds for the non-linear setting are still an open research problem. However, the uniform blocks can be regarded as extreme behaviors (from very soft to very hard), and we can expect that combinations of these materials will produce in-between deformations, in which case our bound estimation will not cull any optimal designs.

5.5.4 Clustering Strategy

We traverse the decision tree in a breadth-first manner, and hence a parent level with n nodes produces another level with $n \cdot M$ nodes. Evaluating bounds on this new level requires the computation of $n \cdot M^2$ designs. In order to limit the breadth of the tree, and thereby the total number of designs that need testing, we cluster nodes at every level before applying the split operation.

We cluster the n nodes at a level into K clusters using K-means clustering, using as distance $dist(a, b)$ between two nodes the sum of squared example displacement differences, evaluated for the pairwise uniform descendants. Formally, the distance metric is:

$$dist(a, b) = \sum_{j}^{M} \|\mathbf{x}(m_{1a}, \ldots m_{la}, \bar{m}_j, \ldots, \bar{m}_j) - \mathbf{x}(m_{1b}, \ldots m_{lb}, \bar{m}_j, \ldots, \bar{m}_j)\|^2. \tag{5.5}$$

The cluster representative is the node that is closest to the centroid of the cluster. Every time we split a level we need to test only KM^2 designs. Since the height of the tree is equal to the number of cells N, our clustering strategy limits the total number of design evaluations to roughly $O(KM^2N)$. Note that the actual number of tested designs is smaller due to bound-based culling. In our implementation, we usually use $K = 20$ clusters. This clustering approach comes at the cost of missing the global optimal solution.

5.6 Automated Deformation Capture

To acquire surface deformations of objects with a wide range of material properties, we built an automatic measurement system that is able to acquire many different materials with varying geometry and surface properties. We use our system to probe base material samples, composites of base materials for model validation, and complex real-world objects together with their reproductions.

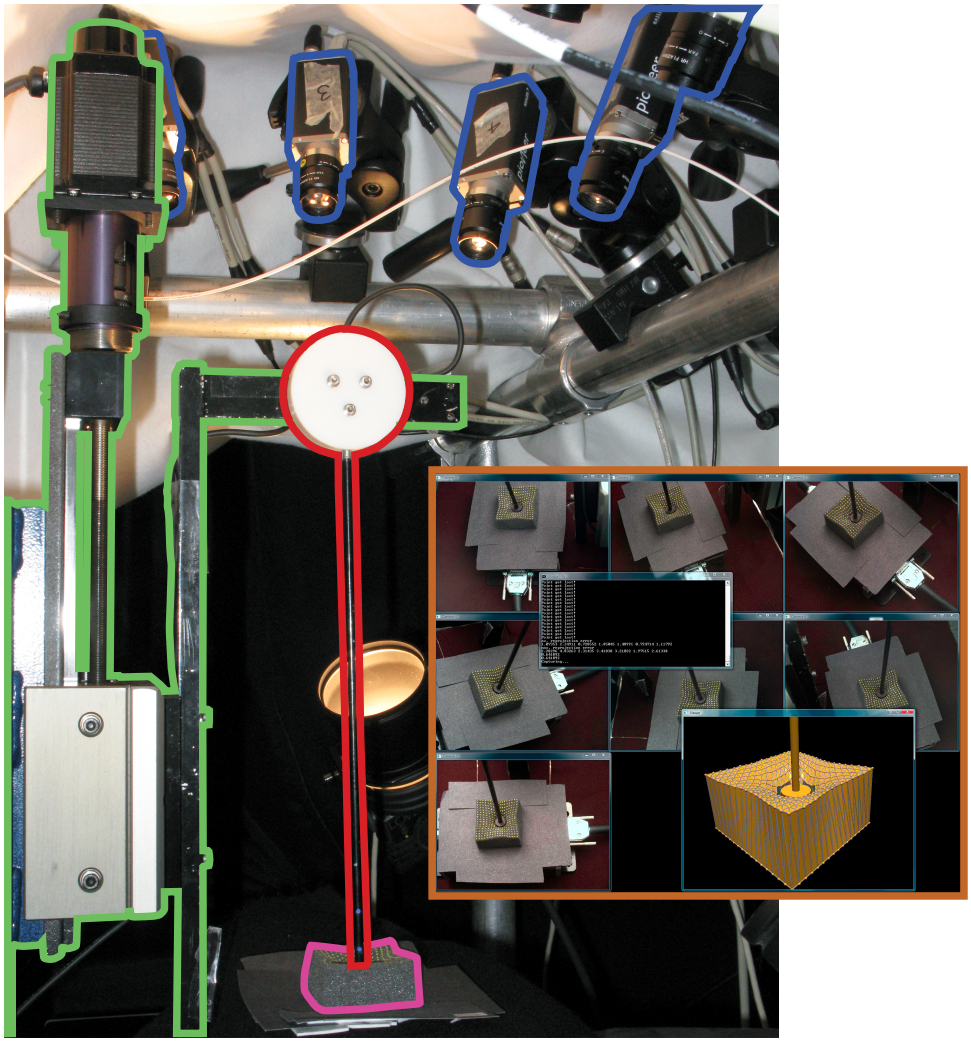


Figure 5.6.1: Automated Deformation Capture Our automated system for measuring material deformations consists of cameras (blue), a robot arm (green), and a force sensor attached to a stick (red). A sample material block is shown in pink and the inset shows a screen shot of our processing software.

Our measurement setup (Figure 5.6.1) consists of a four DOF robot arm (from MicroProto Systems), a six-axis force-torque sensor (Nano 25 from ATI), and a vision subsystem to track surface displacements. The resolution of the robot arm is 0.003 mm and its repeatability is 0.01 mm. The maximum range of the force sensor is 125 N with a resolution of 1/24 N. The vision subsystem consists of seven high-resolution Basler Pilot cameras running at a resolution of 1600 x 1200 pixels. We set up the calibrated cameras [Svoboda et al. 2005] on a half-circle above the robot arm to minimize occlusions and added diffuse lighting.

We paint regular, dotted grids with 3 mm spacing on the objects' surfaces, then extract these markers from the captured frames using a scale and affine invariant blob detector [Mikolajczyk and Schmid 2004] and track them. For each acquisition, we use 30 to 200 deformation steps, depending on the stiffness of the material. The maximal forces are in the range of 35 to 50 N. Finally, the tracked markers and corresponding forces are registered to a surface mesh.

5.7 Representing Base Materials

5.7.1 Isotropic Base Materials

To print 3D deformable objects and a set of base materials we use the OBJET Connex 500 multi-material printer. In each run, the printer can use up to two different materials, e.g., Vero White (rigid) and Tango Black Plus (soft). As discussed in an earlier chapter, the printer can also mix these two materials in predefined proportions, producing isotropic materials of intermediate stiffness. We mainly

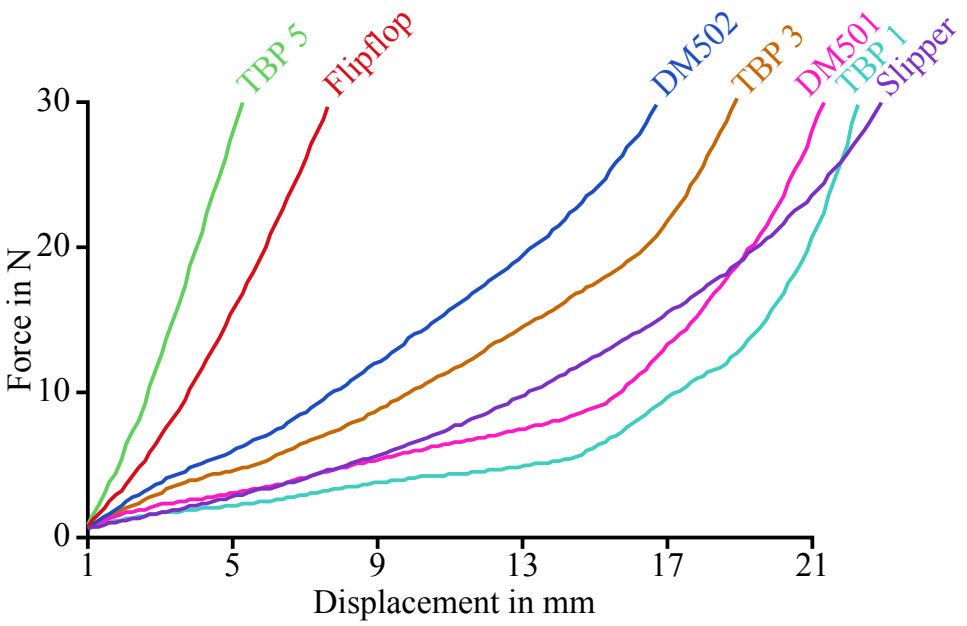


Figure 5.7.1: Force-Displacement Curves of Measured Materials When plotting displacements (horizontal axis) against applied forces (vertical axis), we unveil the high non-linearity inherent in most of our measured materials.

use *Tango Black Plus* (TBP) and a mixed material called *digital material with shore 50* (DM50). In addition to these two isotropic base materials we measured eight isotropic materials from the McMaster-Carr online catalog, including rubbers and foams. Figure 5.7.1 shows a plot of surface displacement as a function of applied force for a subset of measured materials.

5.7.2 Transversely Isotropic Base Materials

In order to model and fabricate materials with even larger deformation gamut (in particular, materials that are much softer) we introduce internal

void spaces into the printed objects. Unfortunately, the current printer only allows printing void spaces that span the entire object along the z-axis. We use tubes of four different right). These objects are isotropic in the horizontal plane, perpendicular to the tube direction. The material can be regarded as *transversely isotropic*.

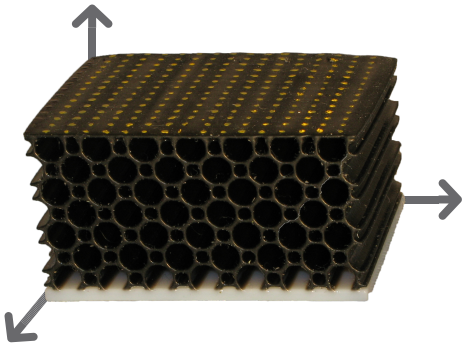


Figure 5.7.2: Transverse Isotropy

For such materials, the matrix $\mathbf{E}$ can be represented as:

$$\mathbf{E} = \begin{bmatrix} E_{11} & E_{12} & E_{13} & 0 & 0 & 0 \\ E_{12} & E_{11} & E_{13} & 0 & 0 & 0 \\ E_{13} & E_{13} & E_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & E_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & E_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{(E_{11}-E_{12})}{2}, \end{bmatrix}. \tag{5.6}$$

with five degrees-of-freedom, $\{E_{11}, E_{12}, E_{13}, E_{33}, E_{44}\}$. Our non-linear material model can then be considered as a five-valued function in a six-dimensional strain domain, $\mathbf{p}(\varepsilon) : \mathbb{R}^6 \rightarrow \mathbb{R}^5$.

All base materials were printed as 4 cm (width) $\times$ 5 cm (length) $\times$ 2.5 cm (height) blocks. The deformations (side view) of some of these materials under 15 Newtons force are shown in Figure 5.7.3.

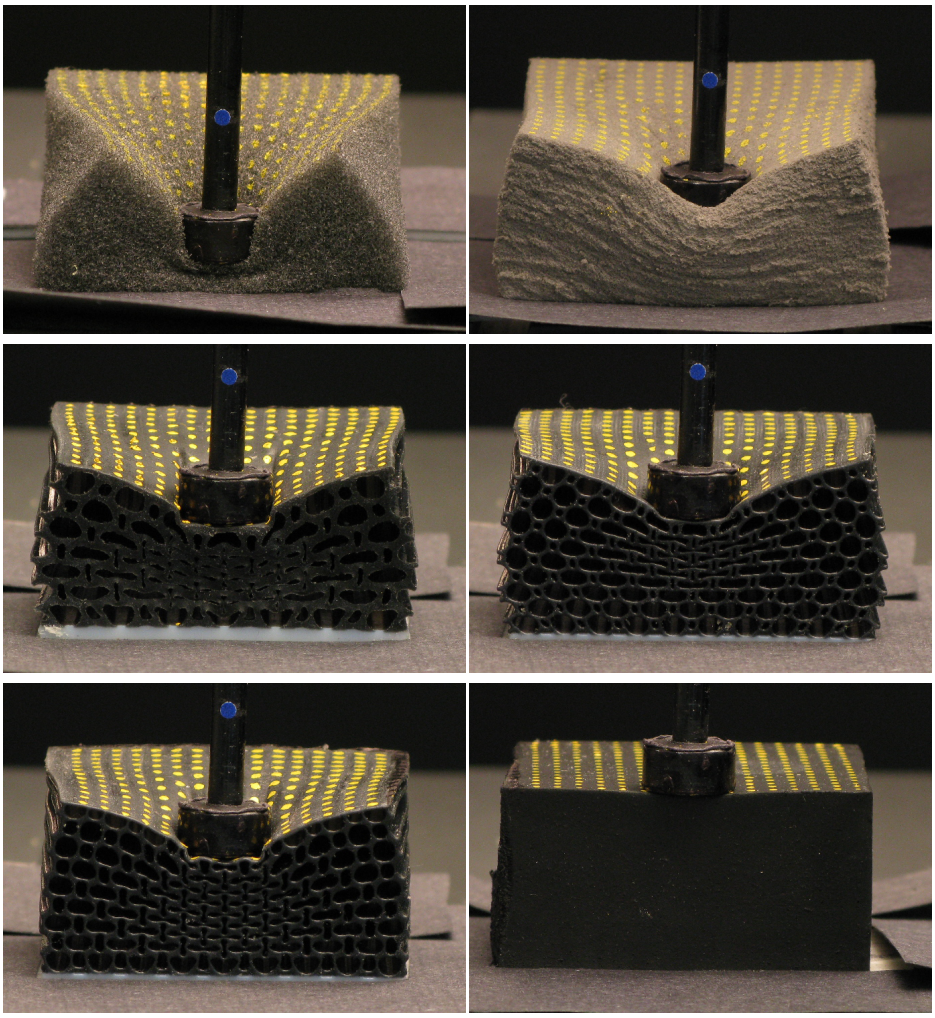


Figure 5.7.3: Base Materials Side view of several base materials during data acquisition. The magnitude of the applied force is 15 Newton in all views.

5.8 Validation and Results

5.8.1 Validation of the Fitting

Our material model represents elastic behavior of the base materials at the mesoscopic level very well. In Figure 5.8.1 we compare images from our measurement system, the reconstructed deformed surface, and the corresponding simulation using FEM. We also show an error plot between the measured surface and the simulation. Note that the error is only evaluated at the surface marker positions and then interpolated for visualization purposes. Furthermore, the error evaluation is dependent on the accuracy of the measurement system which is in the range of < 1 mm. Very small pitching effects at the microscale of the material cannot be tracked by our system and are therefore missing in the error visualization. Refer to our video for more results. For isotropic base materials we use six and for the transversal isotropic materials 12 RBF centers, resulting in 12 and 60 parameters for each base material, respectively. Fitting the material model takes two hours on average but has to be performed only once. We also report the average, standard deviation, and maximum errors for the materials under varying applied loads in Table 5.8.1.

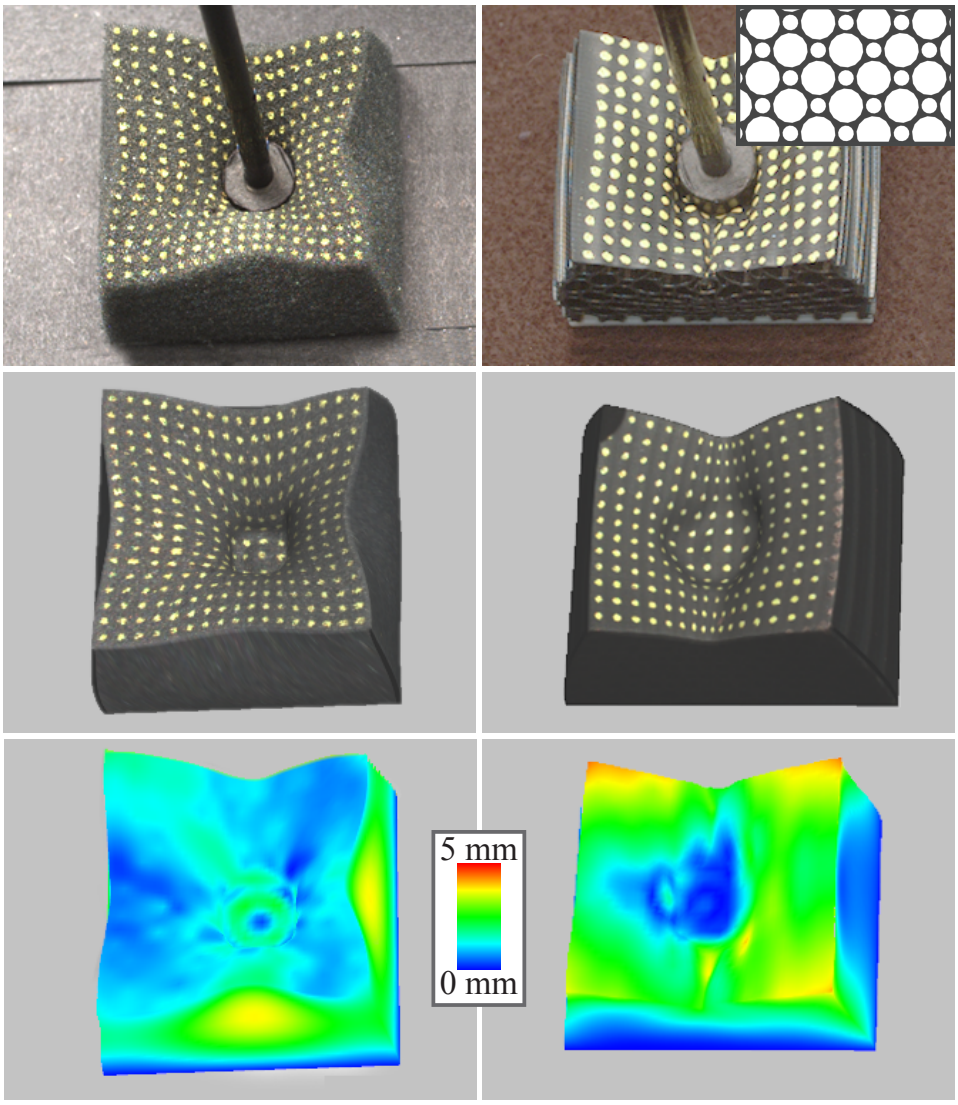


Figure 5.8.1: Side-by-Side Comparison of Real and Simulated Materials Deformation of an isotropic (left column) and transversely isotropic material (right column), comparing acquisition (top row) with the simulation (middle row) and the displacement error (bottom row).

Material	Force (N)	Displacement error (mm) avg.	std.dev	max
Foam (very soft)	1N	0.84	0.45	2.55
	3N	1.72	1.00	6.74
	5N	2.04	0.88	5.70
Foam (medium)	5N	1.40	0.57	3.44
	15N	1.06	0.36	2.46
	25N	1.33	0.90	5.19
Foam (stiff)	10N	0.73	0.43	2.70
	20N	0.94	0.40	2.71
	30N	1.20	0.38	2.22
Printed TBP1 (soft)	5N	2.14	0.68	4.47
	10N	2.40	0.77	4.67
	20N	3.60	1.22	6.55
Printed DM502 (medium)	5N	0.69	0.26	1.44
	15N	0.85	0.41	2.09
	25N	1.31	0.51	3.00
Printed TBP5 (stiff)	10N	0.68	0.22	1.15
	20N	0.99	0.27	1.70
	30N	1.30	0.311	2.61

Table 5.8.1: Error Evaluation of the Model We fitted parameters for various isotropic (soft/medium/hard foams) and transversely isotropic materials (printed, with cylindrical hole structures) and evaluated the surface displacement error under small, medium, and high force loads by comparing to measured deformations of material blocks (size isotropic 5x5x2.5cm, printed 5x4x2.5cm).

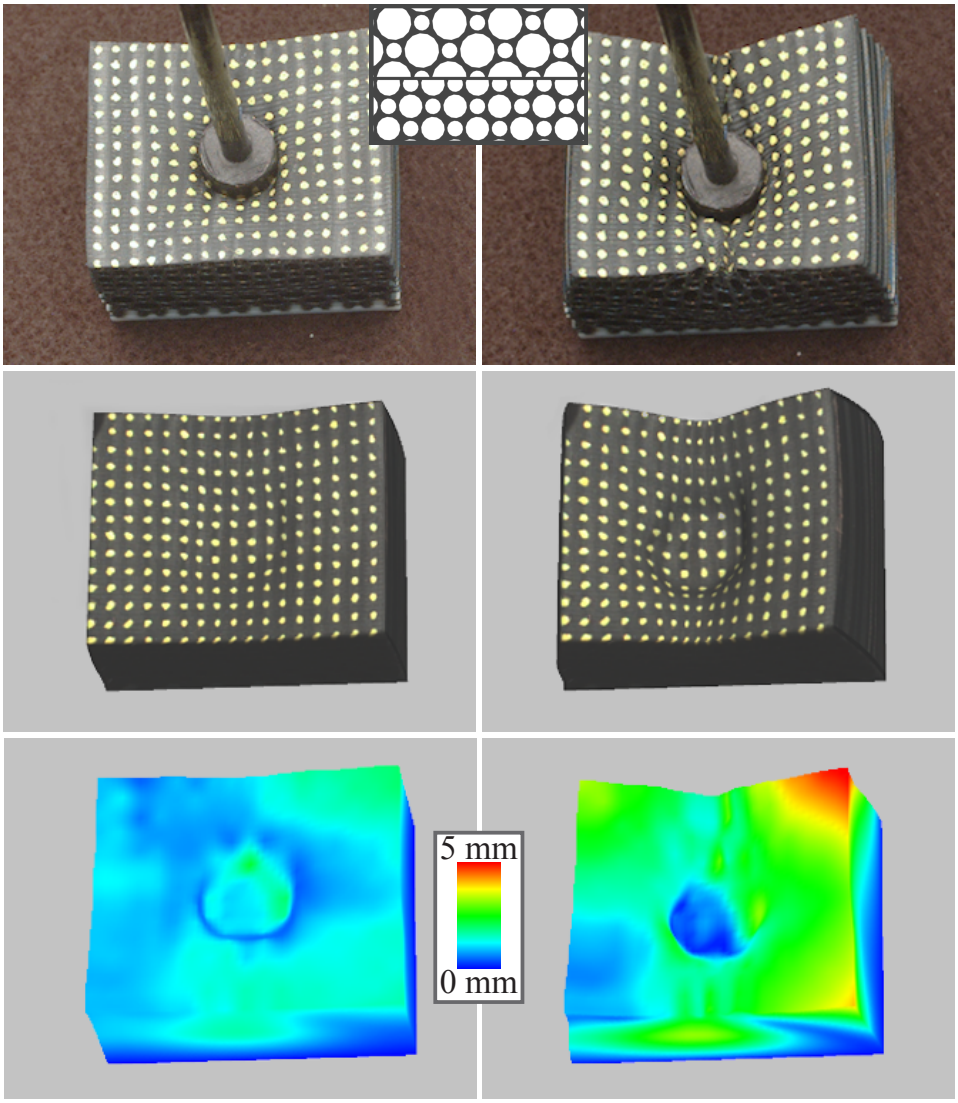


Figure 5.8.2: Validation of Composite Materials We assigned the material properties obtained from two independent fits of base materials (DM501 and DM502) to a composite, consisting of two layers. We then printed the composite and compared the deformations of the real object (top row) to the simulation (middle) under a load of 8 and 21 Newton. (bottom row): error visualization.

5.8.2 Validation of the Stacking

Next, we show that we can accurately predict the behavior of composite materials made from arbitrary combinations of base materials. We ran a number of simulations for different composites and also fabricated those using the Connex 500 printer. Next, we measured the behavior of these composite materials and compared them to their corresponding simulations. We report this validation for a few example deformations and materials in Figure 5.8.2 and in our video. In the composite example shown in Figure 5.8.2, we obtain average errors of 1.98 mm and 2.16 mm under loads of 10 N and 20 N.

5.8.3 Validation of the Goal-based Design

Next, we validate our goal-based design process. We first tested our process on materials that we know we can reproduce. We picked a given combination of layers and their thicknesses. We then simulated this composite material and used its deformations as the input to the search algorithm. We report the result of this validation in Figure 5.8.3. Thereafter, we tested this strategy on 20 different randomly chosen material designs (5 layers, each with 9 different material choices and 5 force-displacement pairs). Although our search is not guaranteed to find the global optimum, it always found a very close solution (average RMS error of 0.067 mm). The optimization time is usually below one hour. To carry this validation even further, we have fabricated these composites and remeasured their properties. We show these results in Figure 5.8.3.

We also tried to approximate one of the foams with a combination of materials

printed using the Connex 500. The obtained spatial combination and the error evaluation are shown in Figure 5.8.4.

5.8.4 Replicating Objects

We ran our complete replication process on several challenging real-world objects, including a pair of flip-flops, felt slippers and a heterogeneous leather stool. We first 3D scanned each object using a Cyberware scanner. Thereupon, we acquired their deformation behavior using our automated mesaurement system (Section 5.6), followed by fitting corresponding material parameters. For the leather stool, we segmented the volume into two areas, and approximated each of them as a homogeneous material. Next, we used the goal-based design process to find the best approximation of the material's deformation properties using our base materials. For all results, we used between 5 and 10 force-displacement pairs. Finally, we printed replicas of these objects using the multi-material printer. As we can be seen in Figures 5.9.1 and 5.9.2 and the accompanying video, the replicas show very similar behavior to the original objects. To further validate this approach, we show force-displacement curves in Figures 5.9.1 and 5.9.2 for corresponding points on replicas and originals.

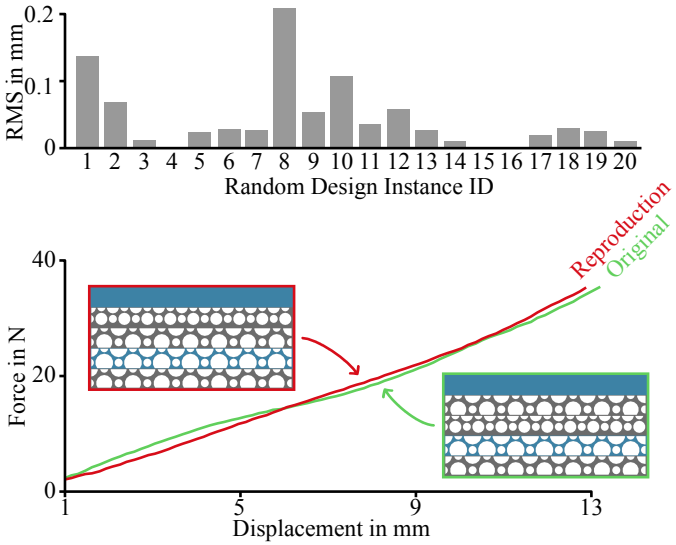


Figure 5.8.3: Validation of Goal-Based Design Algorithm We randomly generate a set of material designs. We then simulate these designs and use their simulated deformations as input to the goal-based design search algorithm. We then compare the obtained designs of the search algorithm with the known ground truth. The upper bar plot shows the RMS error. We also fabricated one of those randomly generated designs and its corresponding search output and compared their force-displacement curves.

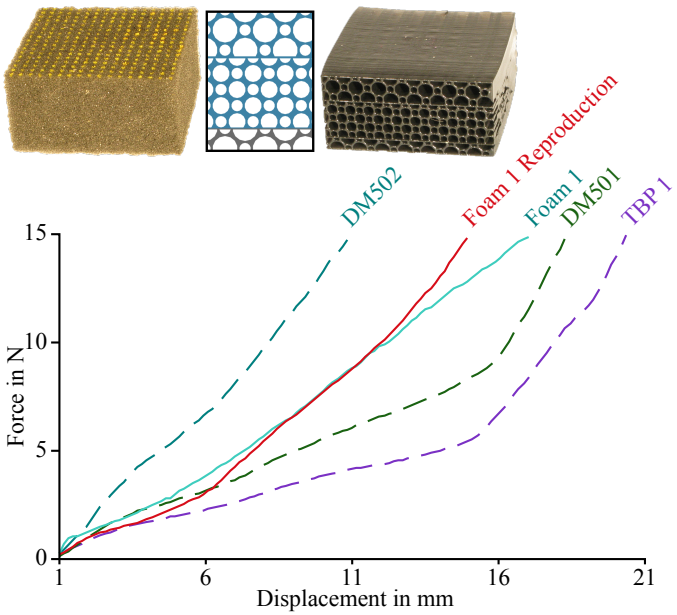


Figure 5.8.4: Fabricated Example of Goal-Based Design We ran our goal-based design algorithm on a foam block's deformation behavior (upper left), specified by 5 example deformations. The desired deformation behavior is approximated by finer scale materials obtained through combinatorial optimization, and then fabricated using a 3D printer (upper right). The lower curve shows the force-displacement relationship of used base materials, foam, and fabricated approximation.

5.9 Discussion

5.9.1 Limitations and Future Work

We believe that our system has many potential avenues for improvements and future work. We predict that this process will be a template for many future systems that expand the range of simulated and fabricated material properties (such as dynamic deformation properties or plasticity). More specifically, we plan to extend our model to dynamic and plastic deformation behavior and improve our measurement system such that it can acquire a wider range of deformation properties (e.g., material stretching and dynamic deformation measurements) or can guarantee and incorporate prior physical knowledge, such as volume preservation. Additionally, we plan to investigate strategies for optimally choosing the number of degrees of freedom (RBF centers) of our material model, striking a balance between accuracy and overfitting. Furthermore, we would like to examine material homogenization strategies [Kharevych et al. 2009] to improve the speed of the forward (simulation) step for non-linear materials. This improvement along with more advanced search strategies could, in turn, speed up the inverse step, making the design and fabrication of extremely complex heterogeneous materials feasible.

5.9.2 Spatial Combinations

Currently, we only print layers of different materials. However, we believe our algorithm could be extended in a straightforward manner to arbitrary spatial combinations (e.g., voxels) of base materials. The decision tree could be directly ap-

plied to 2D or 3D problems, by having a one-to-one mapping of layers in 1D to voxels in 3D. Also, our pruning strategy (clustering and bounds) can be directly translated to the 3D case. Our search algorithm linearly scales with the number of layers or volume elements.

For current printers, the mechanical range of isotropic base materials without any holes or tube structure is limited. The OBJET Connex 500 printer can mix two different materials, and the material properties are restricted to the range between the two loaded materials. To significantly expand this range, we deliberately decided to create tube-structured materials. Due to current physical printer limitations, these void tube structures can only be printed along the z-axis of the printer, otherwise they would get filled with structure material, which is difficult to remove. Printing blocks or objects with isotropic hole structures (similar to Swiss cheese) is currently not possible. This comes at the cost of requiring a transversely isotropic material model.

Looking into the future, we predict that the next generation of 3D multi-material printers will be able to use many more base materials with a wider range of material properties and more complex internal structures. As the cost of these printers decreases and their capabilities increase, we believe that our goal of personalized design, modeling, simulation, and fabrication will become reality.

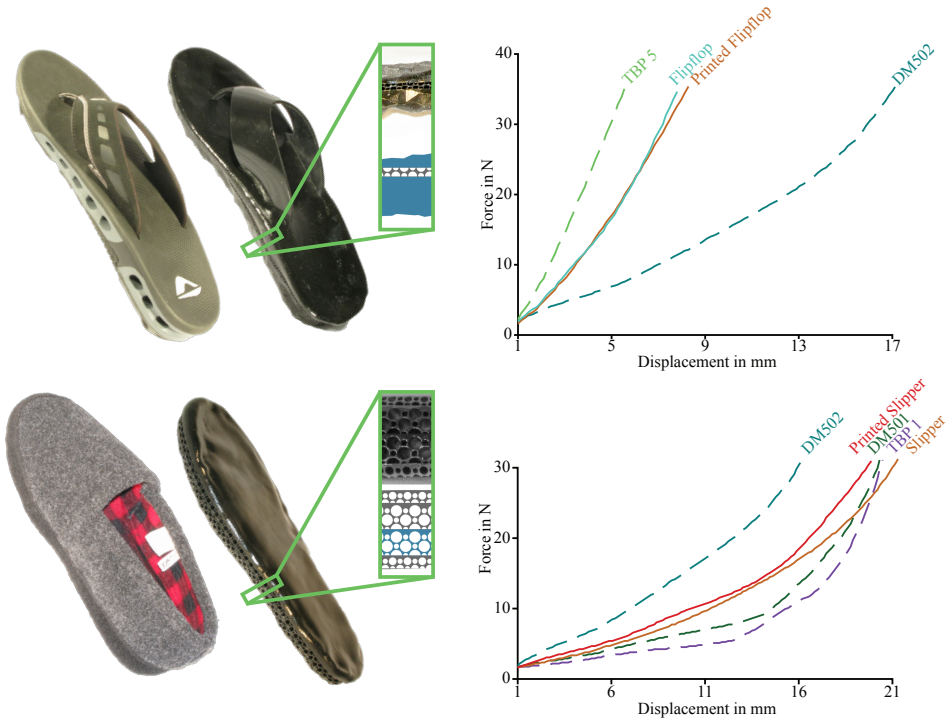


Figure 5.9.1: Replicating Flip-Flops and Slippers Flip-flops (top row) and slippers (lower row). The left column shows the original images, the middle column the replicated flip-flop and slipper with the spatial combination of base materials obtained by our goal-based optimization approach. Our replication matches the deformation behaviour of the original well, as shown in the force displacement plots (right column) for a corresponding point on the original and replica. The dotted curves characterize the base materials.

5.10 Summary

We presented a complete process for measuring, designing, and fabricating materials with desired deformation behavior. Our model is able to represent and simulate the non-linear elastic deformation behavior of objects with complex internal structures. To ensure high agreement between deformations of real materials and their simulated behavior, we use a data-driven measurement process to estimate non-linear stress-strain models for each material.

Furthermore, we show that a goal-based material design approach can approximate a desired global deformation behavior by finer scale materials through combinatorial optimization. By closing the loop between measurements, simulation, goal-based material design, and 3D printing, we validate the complete pipeline and show that close matches between simulated and fabricated objects are achievable. We believe that our goal-based design is a significant step towards 3D hardcopying.

Our design approach also allows to fabricate any virtual deformable content as long as we can sample displacement-force pairs. Hence, most physically-based deformable models commonly used in graphics and other fields can be fabricated using our processing.

Next, we will fabricate animated characters from skinned meshes. In contrast to our work presented in previous chapters, skinned meshes are non-physical descriptors of deformable models. Non-physical content is particularly challenging to fabricate because it is unclear how we best estimate models approximate the given non-physical content.

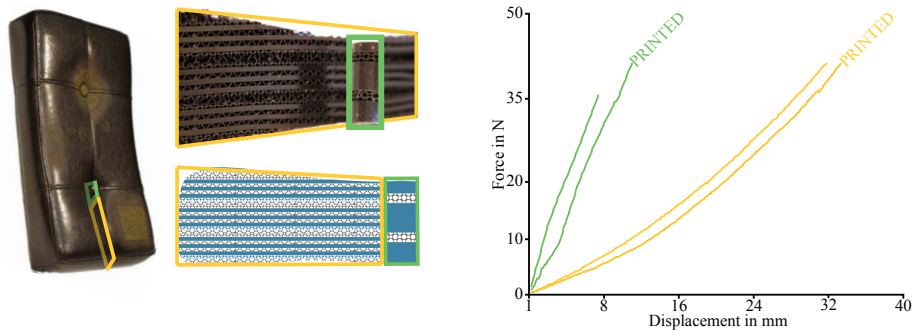


Figure 5.9.2: Replicating a Leather Stool The left column shows the original object, the middle column a cross section of the replicated object and the spatial combination of base materials. We segmented the stool into two regions, a stiff region below the button (indicated in green) and the remaining softer region (indicated in orange). We validated the deformation behaviour by comparing the force displacement plots (right column) in the button region (orange) as well as in the softer region (green).

I'm obsessively detail-oriented.

Donald Knuth

6

Fabricating Articulated Characters from Skinned Meshes

So far we automated the fabrication of deformable models that are physically plausible. In this chapter, we propose a method for the fabrication of skinned meshes, encoding deformation models that typically exhibit a highly non-physical

behavior. Hence, our technique can be understood as a fabricator of solely digital content. Given such a skinned mesh, we estimate a fabricatable single-material model that approximates the 3D kinematics of the corresponding virtual articulated character in a piecewise linear manner.

After further motivating our work in Section 6.1 and outlining our method in Section 6.2, we discuss manufacturing considerations, then detail on our articulated model estimation in Section 6.4. We provide several demonstrations, manufactured as single assembled pieces using a 3D printer in Section 6.5, before concluding with a discussion and summary in Section 6.6.

6.1 Introduction

Skinned characters are among the most widespread models in computer graphics and have received tremendous attention in recent decades. Skilled artists have years of experience in creating weighted associations between a hierarchical set of bones (*rig*) and groups of vertices on the character's mesh (*skin*). Content creation systems, such as the one built into SPORE [Hecker et al. 2008], allow even naive users to create sophisticated skinned characters.

Recently, online services such as Shapeways have become available, making personalized manufacturing on cutting edge AM technologies accessible to a broad audience. Affordable desktop printers will soon take over, enabling people to fabricate custom-made 3D models at home. However, animation software packages such as Maya or Blender lack a "3D print button" to facilitate converting a vir-

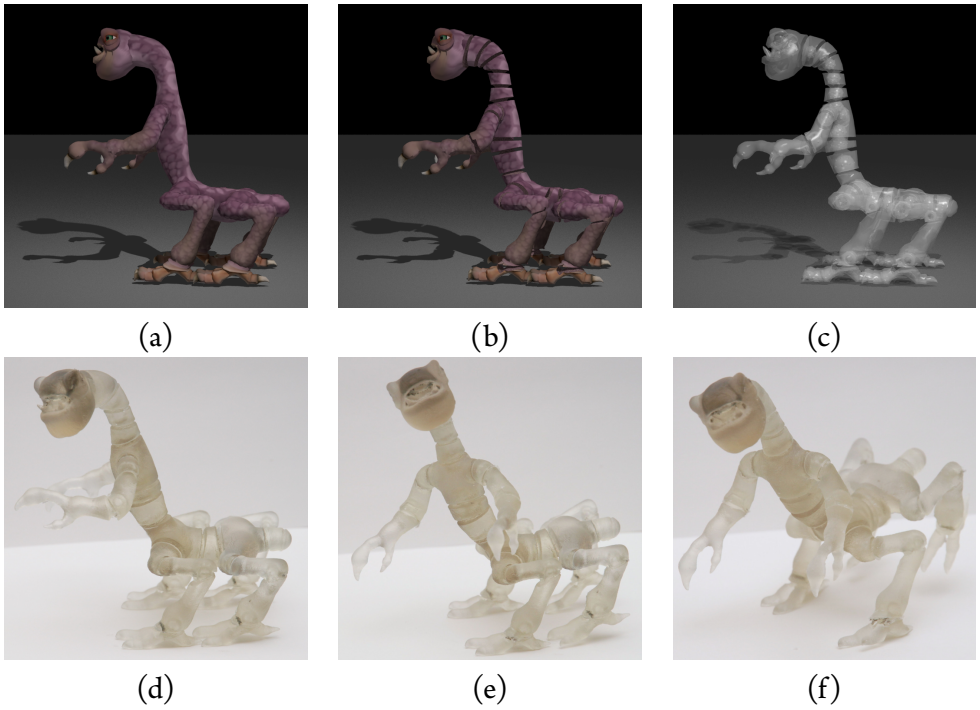


Figure 6.1.1: Fabricating Articulated Characters Given a skinned mesh (a), we estimate (b) a fabricatable articulated character with (c) internal joints of hinge and ball-and-socket type. (d-f) Final 3D printed characters (transparent material) have durable joints with a frictional design for character posing.

tual articulated model into a fabricatable format. While tools and services that map static properties such as geometry and appearance exist, the articulated behavior—a key property of posable skinned models—remains unmapped.

In this chapter, we present a technique that estimates an articulated character model suitable for manufacturing with AM technologies from a given skinned mesh (see Figure 6.1.1 (a)). Our method is capable of generating posable models consisting of a set of piecewise rigid pieces with non-overlapping, physically meaningful ball-and-socket or hinge joint parts (Figure 6.1.1 (b,c)).

Note that a direct mapping from virtual articulated to manufacturable, jointed models does not exist. For starters, rig joints are close to physically meaningless as they can move out of the deformed geometry as illustrated in Figure 6.1.2 left with a rigged cylinder. Furthermore, because they are also not guaranteed to be embedded in the character's geometry in its rest pose, they are not a reliable estimate for joint center placement. Also, while rig joints are zero-dimensional points, mechanical joints are volumetric entities that need to be large enough for structural strength, and as such can potentially "collide" with each other if care is not taken in the joint design process (see Figure 6.1.2 right). Our approach addresses these concerns.

While our method is capable of automatically generating articulated models with ball-and-socket joints set to default ranges, these 3-DOF (degrees of freedom) defaults may restrict the posing space of fabricated characters either too little or too much. We therefore allow users to switch individual joints to hinge type (1-

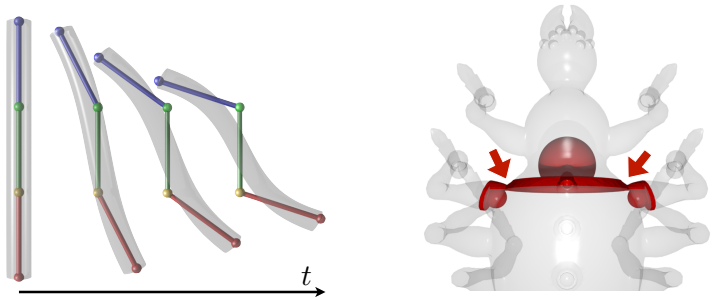


Figure 6.1.2: Virtual Rig vs. Mechanical Joints: When animating a rigged cylinder (left), we observe that the rig joints do not fall together with actual rotation centers and move out of the deformed geometry. (right) If we maximize the individual sizes of mechanical joints (and thus their strength), they could collide (red).

DOF) and to specify range parameters differing from defaults for both of our joint designs. For all our demonstrations, user-intervention is limited to a subset of the joints.

After first analyzing the mesh and skinning weights, we estimate proxy joint locations, and assign custom parametric models for volumetric joint geometry that are consistent with any user-specified joint limits. We then proceed to optimize joint parameters (location, size, etc.) to increase joint strength while avoiding overlapping joint geometry. By augmenting our joint models with tiny bumps to increase joint friction, our output models can be posed and will retain their configuration (see Figure 6.1.1 (d-f)). Finally, the estimated joints are carved out of the character mesh using CSG operations. Additional overviews of our approach are given in Figure 6.2.1 and Section 6.2.

For completeness and to assure high quality of our output models, we approximate the characters' surface appearance also. Because the resolution of the geometry of many skinned characters is kept low for fast rendering, we estimate micro-geometric detail from normal maps if available. Carving out joints from character meshes also works on textured content. We demonstrate the applicability of our approach on a number of examples (see Figures 6.1.1, 6.5.2, 6.5.3, 6.5.4).

We show that an analysis of skinning weights leads to a plausible segmentation of the character's geometry into rigid body parts. Furthermore, we present novel, geometric approximate models of joint strength, that, together with our method to avoid joint-joint collisions, ensure strong and functional joints in our output models. Also, our collision resolution allows us to keep as much of the "fabricat-

able" input articulation in our posable output models as possible. To the best of our knowledge, we are the first to present a technique to automatically convert skinned meshes into durable, articulated models.

6.2 Overview

For articulated characters, we have to successfully map three components from the virtual model to reality: two static properties, namely *geometry* and *appearance*, and the model's *articulation* that allows it to be posed. See Figure 6.2.1 for an overview of our fabrication pipeline. Next, we identify the properties we use.

6.2.1 Input: Skinned Characters

The input to our estimation process is a skinned character (see Figure 6.2.1 left). The input *geometry* is specified as semi-organized set of oriented face tuples $\mathbf{f} \in F$ whose components f_j index into a set of vertices $\mathbf{v} \in V$. Optionally, *appearance* is specified with color information provided as diffuse texture, and micro-geometric detail encoded in a normal map. As indicated in Figure 6.2.1 (d), our input mesh could potentially consist of a set of individual, overlapping mesh components. By repairing (removing duplicate vertices, resolving violations of manifoldness, etc.) and unifying this set of components, we compute a manifold, closed surface mesh $(V_{\mathcal{F}}, F_{\mathcal{F}})$. Because this mesh fulfills the requirements of manufacturing, we call it a fabrication mesh $\mathcal{F}$. Without loss of generality, we hereafter assume faces and vertices to refer to entities of repaired meshes, and the faces to be triangles.

The *articulation* behavior is specified by a LBS model wherein each vertex i in V is weighted to link $l \in L$ by a (nonnegative) skinning weight w_{il}, such that the deformed vertex position is given by

$$\mathbf{v}'_i = \sum_l w_{il}\ \mathbf{T}_l\ \mathbf{v}_i, \tag{6.1}$$

where $\mathbf{T}_l$ are some unknown time-varying link transforms. Moreover, we require the set of link correspondences L to have tree-structured connectivity defined by a function P that maps every link $l \in L$ to its unique parent $P(l)$. We also add an index $\omega \notin L$ and denote the link r whose parent is $P(r) = \omega$ the root node. Note that such a LBS description is the lowest common denominator of practically all articulated characters found in games.

6.2.2 Pipeline Process

Given the skinned input mesh, our method proceeds to estimate an articulated model as follows (refer to Figure 6.2.1). In the *joint estimation* branch (lower part in Figure 6.2.1) of our pipeline (see Section 6.4.1), we first analyze the skinning weights and their link correspondences to segment the original geometry into an approximate set of body parts (f). From this segmentation, we then derive a filtered set of *oriented joint locations* (g) that consist of orientation vectors, and the joint's rotation centers that we place on an approximate medial axis representation of the fabrication mesh (h).

The fabrication mesh $\mathcal{F}$ (e) together with the articulation data (g) is then fed into our *joint optimization* procedure (i) where posable joints with maximal cross-

sectional areas are being generated from corresponding oriented joint locations together with any user-specified range constraints. Pairwise collisions between generated joints are resolved while keeping the joints' rotation centers fixed (see Section 6.4.4). Overall, our mapping tries to keep as much of the input articulation as possible, while also keeping the model structurally strong. The final set of non-colliding, mechanical joints are then carved out of $\mathcal{F}$ using CSG (j) and we get a ready-to-print, structurally strong, articulated model (k) consisting of a set of piecewise-rigid parts that are jointed together with hinges, or balls and sockets. The models are statically posable using a joint friction design discussed in Section 6.4.3.

Optionally, the joint carving can be performed on a colored, high resolution fabrication mesh whose geometric detail is computed by inverting normal mapping using the weighted least squares version of Nehab et al. [2005].

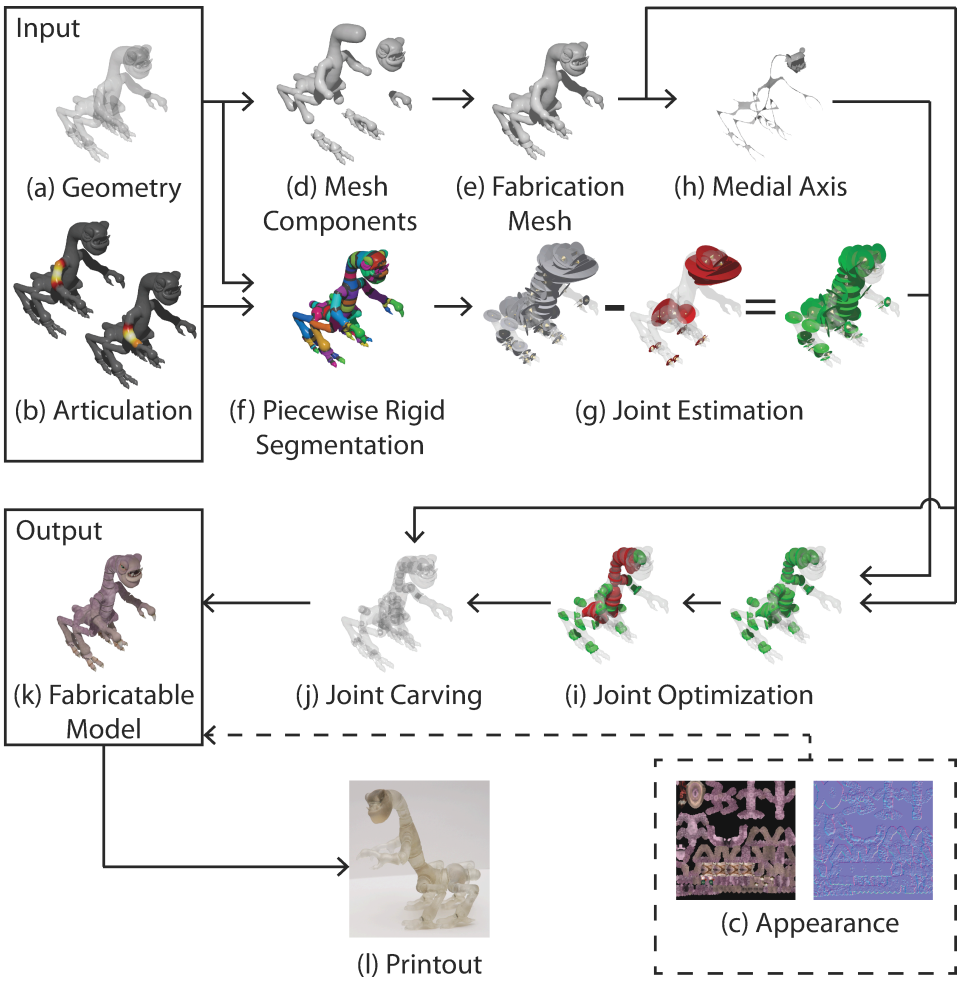


Figure 6.2.1: Pipeline Overview: Given a skinned input mesh with (a) geometry, (b) skinning weights whose link correspondences are organized in a single rooted tree structure, and optional (c) diffuse texture and normal map, our approach estimates a (k) fabricatable 3D model as follows: (d) mesh components are identified, and (e) fused into a single, closed surface we call the fabrication mesh $\mathcal{F}$. Joints are computed by (f) estimating a rigid link segmentation from skinning weights, and (g) estimating proxy joint locations and filtering problematic joints. To optimize joint center placement, we use (h) an approximate medial axis representation of $\mathcal{F}$. (i) The parameters of volumetric joints with optional user-specified range constraints are optimized for strength and to avoid inter-joint collisions. (j) The joints are carved out of $\mathcal{F}$ using CSG operations. The final 3D printout (l) is a posable reproduction of the virtual articulated character.

6.3 Manufacturing Considerations

Our posable output models are tailored to be fabricated on AM devices as single, assembled pieces. To manufacture overhanging or assembled geometry like our mechanical joints, layered approaches use some kind of supporting structure as illustrated on the left in blue and discussed earlier in Section 3.4. After printing, this *support material* can either be blown (for powders), broken, or washed off. To ensure that the individual, assembled parts (in grey) are movable, and not fused during printing, we ensure a device-dependent *minimal distance d* (in yellow) between these pieces. Hence, we treat *d* as hard constraint when estimating our geometric joint models in Section 6.4.2.

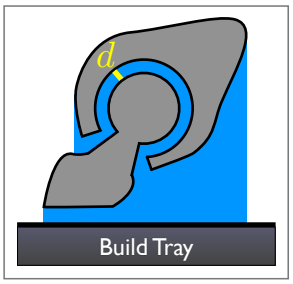
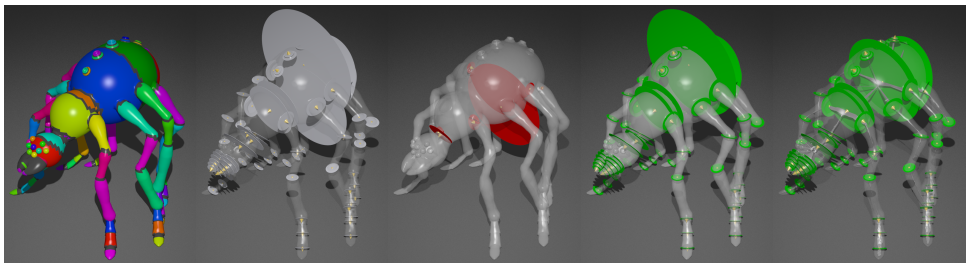


Figure 6.3.1

An important factor for manufacturability on AM devices is the models' structural strength because it puts a limit on the feasibility of desired output dimensions and largely affects the models' durability. If substructures are too fine, they either break off during fabrication, or when interacting with the final printouts.

When designing simple structures (e.g., trusses), civil engineers repeatedly identify their weakest link, and adjust its dimensions. Inspired by this basic analysis, we seek to increase the articulated models' overall strength by identifying and maximizing each of their mechanical joints' critical cross-sectional areas. We reject joints if their minimal cross-section falls below a technology imposed global, *critical area threshold* $A_{\min}$. While this heuristic does not ensure structural optimality,



(a) (b) (c) (d) (e)

Figure 6.4.1: Estimating Articulation Behavior: (a) Piecewise rigid segmentation using skinning weights. Faces whose vertices belong to different segments, are shown in black. (b) Transitions oriented from the root towards the leafs in the link connectivity P, (c) degenerate, and (d) filtered transitions. (e) Final set of joint locations on the scale axis transform of $\mathcal{F}$.

it allows us to formulate our hinges and ball-and-sockets using parametric, geometric models of joint strength (see Section 6.4.2). Note that, because our virtual input characters might be nonphysical, e.g., cartoon characters, their corresponding fabrication meshes could themselves have critical sections below $A_{\min}$ as, e.g., in long and slim necks. While we do not improve the structural strength of our input geometry, our scale-aware simplification (Chapter 7) or the technique by Stava et al. [2012] could be used to further process our fabrication meshes.

6.4 Articulated Model Estimation

We now describe the estimation of oriented joint locations from the character's skin, and cast our hinges and ball-and-sockets as geometric models of joint strength that are then optimized while avoiding joint-joint collisions.

6.4.1 Estimating Rigid Parts and Joint Locations

To estimate oriented locations where mechanical joints are best placed (see Figure 6.4.1), we exploit the link correspondence P encoded in the skinning weights w_{il} and ignore the character's rig. We observe that a segmentation of the character's input geometry (V, F) into piecewise rigid parts is naturally given by assigning each vertex i to the link l with maximal weight $\max_{l \in L} w_{il}$, as visualized in Figure 6.4.1 (a) with a unique hue per link.

Most LBS descriptors lack information about skeletal joint motion (as implicitly encoded in the link transform parts $\mathbf{T}_l$ in Equation 6.1), and often include rig joint locations for the characters' rest pose only. Unlike skinning weights, rig joint locations are not a reliable source for mechanical joint placement because they are nonphysical, zero-dimensional points. Firstly, they are not guaranteed to be embedded in the character's geometry as demonstrated on the right for a spider's mandible. Secondly, rig joints typically do not fall together with actual rotation centers during animations as the cylinder example in Figure 6.1.2 left illustrates. Hence, it is better to place joints at transitions of maximal link influence as shown in black in Figure 6.4.1 (a). Such transitions are by default found in regions where the model bends most during animations and where joints are natural.

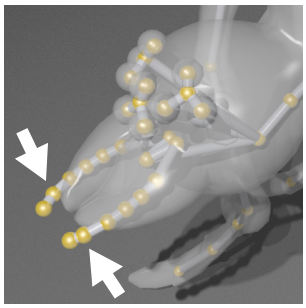


Figure 6.4.2

After segmentation, we approximate each transition with a plane (compare with Figure 6.4.3) as illustrated in Figure 6.4.1 (b) with gray disks. We first identify all unique edges in (V, F) whose end vertices j and k have maximal link influences

$l_j = \arg\max_{l \in L} w_{jl}$ and $l_k = \arg\max_{l \in L} w_{kl}$ with $l_j \neq l_k$. Note that links l_j and l_k do not have to be direct neighbors in the tree-structured connectivity P even though they usually are. We then partition this set of *transition edges* with respect to matching ordered link-pairs (m, o)

$$\bigcup_{(m,o)} \{\{j,k\} \,|A(\{j,k\})\}\,, \tag{6.2}$$

with $A := \big((l_j = m) \wedge (l_k = o)\big) \vee \big((l_k = m) \wedge \big(l_j = o\big)\big)$ and where link m is closer (or equal) to the root than o. Note that in rare cases where transitions (m, o) span over branches in P and where both links m and o have the same distance to the root, the link order is ambiguous. Transition (l_1, l_0) in Figure 6.4.3 left provides an instance of such a case as both links l_1 and l_0 have r as a parent. To resolve this ambiguity, we randomly choose the link order (m, o). Alternatively, the user could specify it. For each edge $\{j, k\}$ in each *transition* (m, o) (see Figure 6.4.3 right), we then compute a *transition point* $\mathbf{p}_{jk}$

$$\frac{w_{j,l_j}}{w_{j,l_j} + w_{k,l_k}} \mathbf{v}_j + \frac{w_{k,l_k}}{w_{k,l_k} + w_{j,l_j}} \mathbf{v}_k, \tag{6.3}$$

with normalized maximal weights w_{j,l_j} and w_{k,l_k}, and, finally, linearly approximate each transition by running Principal Component Analysis (PCA) on the set of corresponding transition points, resulting in a mean point $\mathbf{p}^{(m,o)}$ and principle components $\mathbf{e}_{\lambda_1}$, $\mathbf{e}_{\lambda_2}$, and $\mathbf{e}_{\lambda_3}$, sorted by their variances $\lambda_1 \leq \lambda_2 \leq \lambda_3$. We call the mean point *transition center* and the vector $\mathbf{n}^{(m,o)} = s\mathbf{e}_{\lambda_1}$, the *transition's orientation*. Next, we consistently orient planes (choosing the sign $s = \pm 1$) w.r.t. the

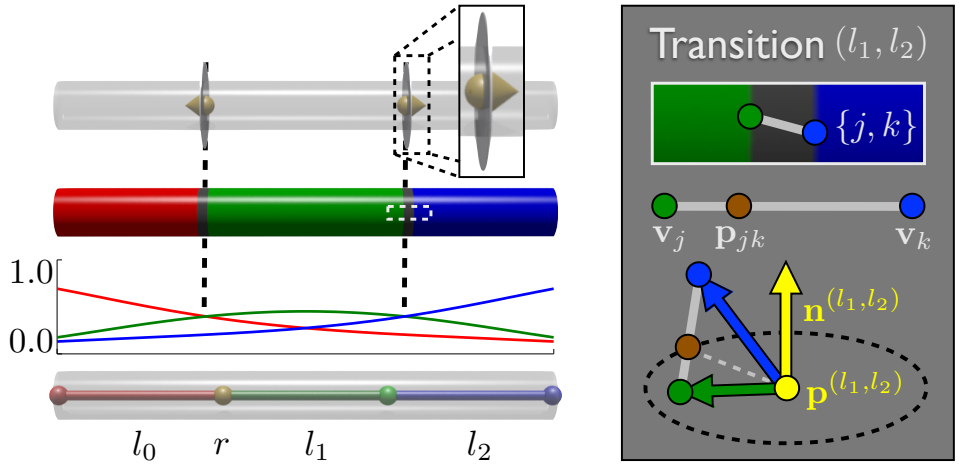


Figure 6.4.3: Estimating Transitions: (left) Skinned cylinder with root r and three links (l_0 in red, l_1 in green, l_2 in blue) with their corresponding skinning weights (bottom). The link connectivity P is defined by $P(l_0) = r$, $P(l_1) = r$, $P(l_2) = l_1$, and $P(r) = \omega$. The two transitions (l_1, l_0) and (l_1, l_2) together with the final oriented transition planes pointing from the root towards the leaves in P (top). (right) A transition edge (in gray) with corresponding transition point (top) for transition (l_1, l_2). (right) From the transition points, and their edges' end vertices (in blue and green), we compute the transition's center and orientation (in yellow, bottom).

hierarchical structure in P (from the root towards the leaves). While orientations do not affect the DOFs of individual mechanical joints in the posable output models, it allows us to pack the volumetric joints more closely, hence to keep more of the overall input articulation. We set s to 1 if more of the edge end vertices $\mathbf{v}_j$ (corresponding to the link m closer to the root, assuming $l_j = m$) are on the positive side of the transition plane ($(\mathbf{v}_j - \mathbf{p}^{(m,o)}) \cdot \mathbf{n}^{(m,o)} > 0$) than end vertices $\mathbf{v}_k$ on the plane's negative side ($(\mathbf{v}_k - \mathbf{p}^{(m,o)}) \cdot \mathbf{n}^{(m,o)} < 0$).

Taking a closer look at the estimated transitions (see Figure 6.4.4), we observe that their corresponding transition points do not always span a closed loop on the

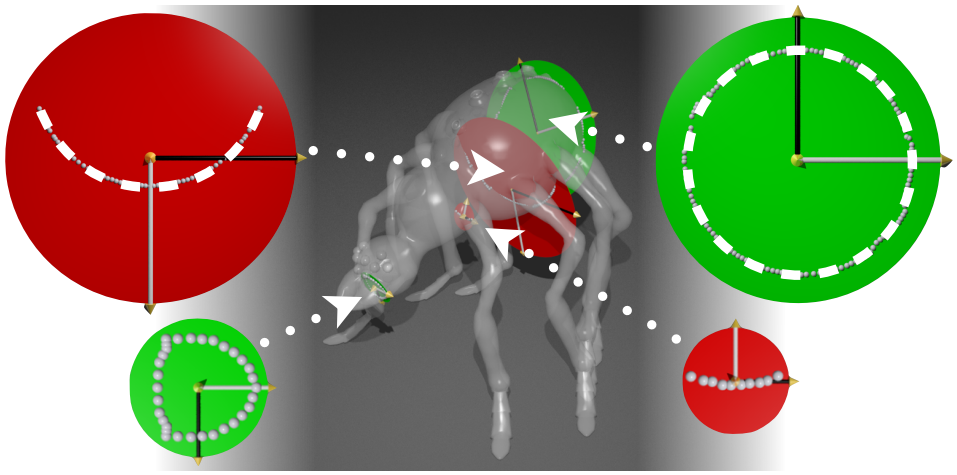


Figure 6.4.4: Filtering Transitions: For valid joints, transition points (gray) span a closed loop on the input geometry (green disks). However, for a subset of transitions (red disks), they only cover a partial loop on the geometry, indicating that the two corresponding body parts are semi-rigidly connected. Because it is unclear how a mechanical joint should be placed for such degenerate transitions, we filter them out.

input geometry, as illustrated in Figure 6.4.4 and Figure 6.4.1 (c) with red disks. Because it is unclear how a mechanical joint should be placed on a transition that, e.g., only covers half of the geometry, we filter out such *degenerate transitions*. We find that a good measure for degeneracy is given by the ratio of the largest- and mid-eigenvalue of the 3x3 PCA covariance matrix at (m, o) because it clearly discriminates between cases where transition points are close to circularly distributed (green disks in Figure 6.4.4) and the degenerate cases. If the largest variance λ_3 is at least a factor f larger than the mid-variance λ_2, we reject the transition. This leaves us with the set of transitions shown in Figure 6.4.1 (d).

Because it is unclear from the articulation data where to best place joint centers on the transitions, we set the centers to the closest intersection $\mathbf{c}^{(m,o)}$ of transition

$(\mathbf{p}^{(m,o)}, \mathbf{n}^{(m,o)})$ with an approximate medial axis representation of the fabrication mesh $\mathcal{F}$. Because the medial axis transform [Blum 1967] is unstable and leads to many unintuitive branches, we use the recent scale axis transform [Miklos et al. 2010] instead. Placing joint centers on the scale axis is reasonable because it allows to maximize the mechanical joints' sizes, hence, to leverage their structural strength. Furthermore, this choice guarantees that the joints' center is always in the interior of $\mathcal{F}$. The final set of oriented joint locations $(\mathbf{c}, \mathbf{n})$ is shown in Figure 6.4.1 (e).

6.4.2 Optimizing Parametric Joints for Strength

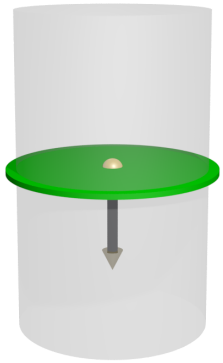


Figure 6.4.5

Given an oriented joint location $(\mathbf{c}, \mathbf{n})$, as illustrated on the left with a cylinder with a single mid-transition, we now estimate mechanical joints. To this end, we cast our hinge and ball-and-socket designs as parametric, geometric models of joint strength (see Figure 6.4.6 left). To minimize interference of the joints with the character's overall appearance, we limit their parameters so that the sockets for both designs are guaranteed to be embedded in the maximum inscribed sphere of radius $r_{\max}$ in the fabrication mesh $\mathcal{F}$, at the joint's rotation center $\mathbf{c}$ (see dotted, black circles in Figure 6.4.6). Furthermore, we keep a minimal distance d between the joint parts to prevent their fusion during manufacturing.

When designing structures, civil engineers repeatedly analyze the stress distribution within the structures' bodies under a set of typical loading scenarios (see,

e.g., [Beer et al. 2011]). A simple view is that the average *stress* across a given *cross-section* A is given by the force per area $\sigma = F/A$, where F is the residual load. If a local stress level is too high, a structure could break, hence, they adjust the design's dimensions in that particular region, thereby increasing the corresponding *critical* area. In the same spirit, we identify a total of three critical cross-sectional areas for each of our designs (see Figure 6.4.6 right) and maximize each joint's minimal area. While these critical areas are parameterized with only two parameters for our ball-and-sockets (the socket's radius r and a height parameter h, see Figure 6.4.6 top row, left), we need three parameters for our hinges: The outer and inner radii R and r, and the width b, limiting the hinge's toroid (see Figure 6.4.6 bottom row, left). This leads to the following two constrained max-min optimization problems.

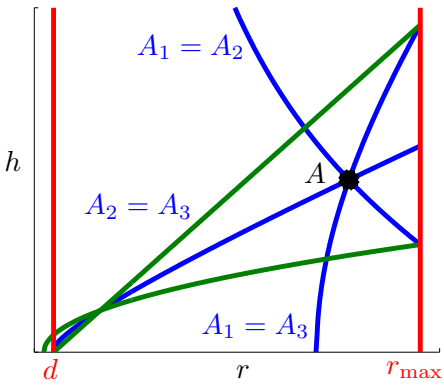


Figure 6.4.7

Ball-And-Socket Joint: For our *ball-and-socket design*, we get

$$\max_{\{r,h\}} \min_{i \in I} A_i(r,h), \qquad (6.4)$$

with $I = \{1,2,3\}$ and constraints $r_{\max} > r > d$ and $r - d > h > \sqrt{r^2 - (r-d)^2}$ limiting the joint's feasibility as shown on the right in red and green, respectively. Note how the curves corresponding to equal areas (in blue) meet at a single point A. For almost all pairs $(d, r_{\max})$, our max-min problem leads to three equal critical areas. If the joint is infeasible or its minimal critical area is below the global threshold $A_{\min}$, we reject it.

Hinge Joint: Similarly, we get

$$\max_{\{R,r,b\}} \min_{i \in I} A_i(R, r, b), \tag{6.5}$$

with $I = \{1, 2, 3\}$ and constrained by $r > d$, $R > d + r$, $r_{\max}^2 > \left(\frac{b}{2}\right)^2 + R^2$, and $b > 2d$ for our *hinge design*.

Note, however, that the ranges for our current designs are limited in directions perpendicular to the joint's orientation (compare with Figure 6.4.6 left). While rotational joint motion is too restrictive for our current hinges, joint motion around axis **n** is unrestricted for our ball-and-sockets. These spherical joints are therefore well-suited for common joints found in hips and spines. For elbow, knee, or shoulder joints, however, they are unfit. Because it is unclear how to estimate joint types, ranges, and the hinges' rotation axes from the character's skin, we give the user the option to specify them. Because general ranges are not rotation-invariant w.r.t. angle-axis $(\theta, \mathbf{n})$, we disambiguate by introducing a right-handed, orthogonal *joint frame* $[\mathbf{a}, \mathbf{n}, \mathbf{f}]$ whose *forward* axis **f** (red arrow on the left) is aligned with the direction where θ is zero. Note how axis **a** (in blue) falls together with our hinge's rotation axis.

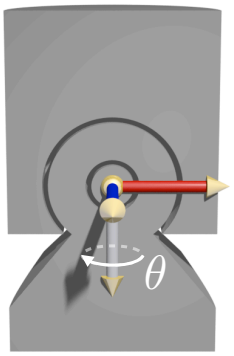


Figure 6.4.8

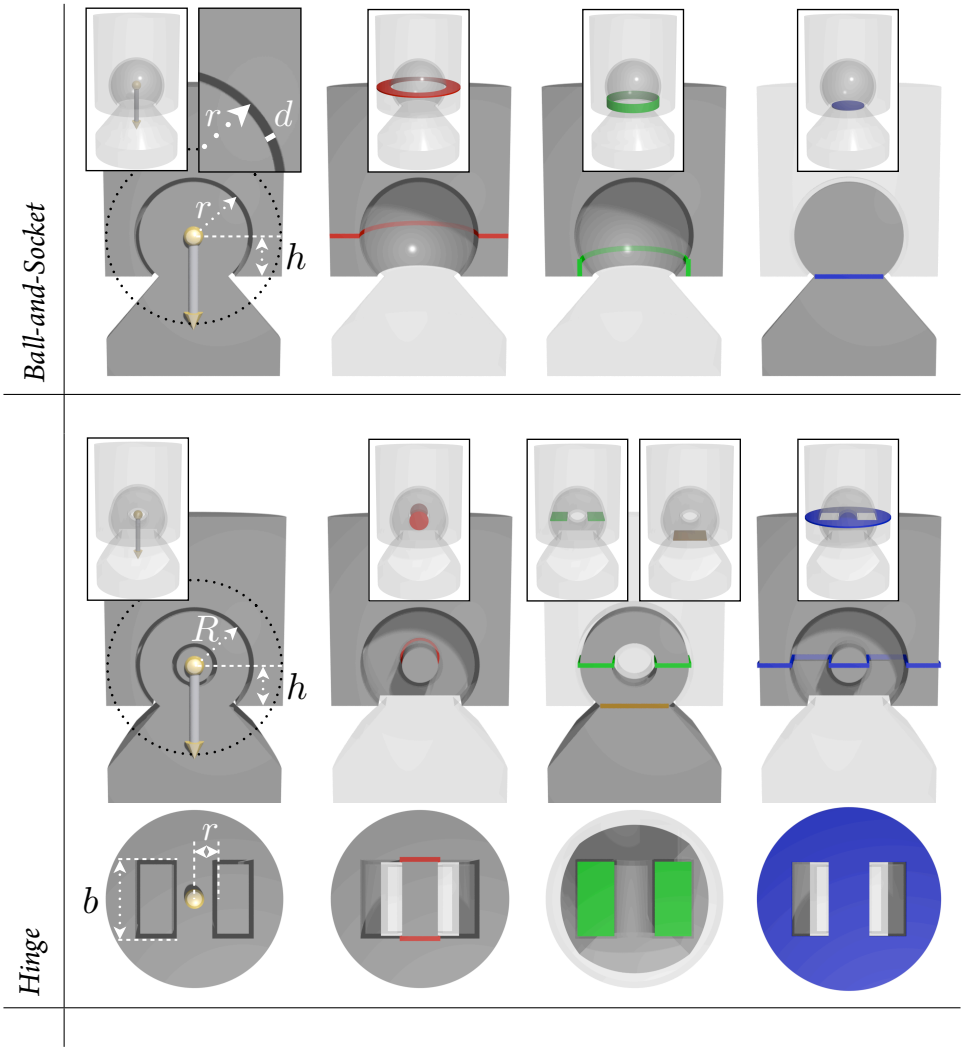


Figure 6.4.6: Critical Cross-Sectional Areas: (top) Our ball-and-socket design with its critical areas A_1 (red, circle of radius $r_{\max}$ with centric hole of radius r), A_2 (green, open cylinder of radius $r-d$ and height $h-\sqrt{r^2-(r-d)^2}$), and A_3 (blue, circle of radius $\sqrt{(r-d)^2-h^2}$). (bottom) For our hinge design, we get A_1 (red, twice the area of circle with radius $r-d$, assuming this section to break in double-shear [Beer et al. 2011]), A_2 (green, twice the rectangular area with sides $b-2d$ and $R-d-r$), and A_3 (blue, circle with radius $r_{\max}$ reduced by twice the rectangular area with sides b and $R-(r-d)$). Area A_4 (brown) is non-critical because for all feasible hinges, there is a h so that $A_4 \geq A_2$. In practice, we choose h so that areas A_2 and A_4 are equal.

User-Intervention: This frame is uniquely defined by our estimated joint locations, up to the axis' **a** rotation angle w.r.t. the joint's orientation that we let the user choose. Ranges can then be specified by direction-dependent *opening angles* $\varphi(\theta)$ for our ball-and-sockets, and forward (γ_f) and backward (γ_b) *swing angles* for our hinges (see Figure 6.4.9 left).

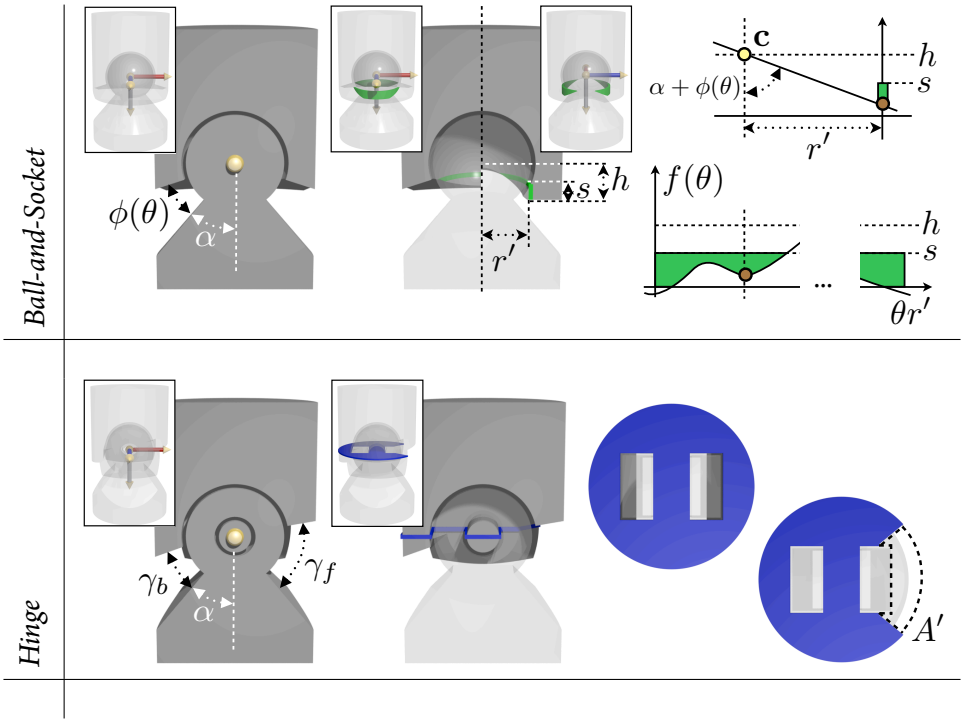


Figure 6.4.9: Joint Ranges: (top) Range constraints for our ball-and-sockets may reduce open cylinder area A_2 (green) of radius r' and height s. The "unrolled" cylinder area (see graph in the lower right) is reduced by the area under $f(\theta)$ that overlaps with range $[0, s]$. Value f at a θ (brown point) is given by the intersection of line through joint center $\mathbf{c}$ and slope $\tan(\alpha + \varphi(\theta))^{-1}$, with the infinite cylinder of radius r' (see upper right, note that $\cos\alpha = \frac{h}{r'}$). (bottom) Forward and backward constraints for our hinges may reduce critical area A_4 by A' each, as illustrated with a swing angle γ_f that leads to a combined angle $\alpha + \gamma_f$ larger than 90° (with $\cos\alpha = \frac{h}{R-d}$).

Range Constraints: These *range constraints* may reduce critical areas of our joint designs as illustrated in Figure 6.4.9 right. For our hinges (bottom row), a swing angle that is – when combined with α – larger than 90°, reduces section A_3 by an amount A'. This reduction can be expressed in closed form, parametrized by the hinge's set of parameters. To incorporate the range constraint $\varphi(\theta)$ into our ball-and-socket design (top row in Figure 6.4.9), we reduce the cylindric area A_2 with circumference $2\pi r'$ ($r' = r - d$) by

$$\int_0^{2\pi} \min\left(s, \max\left(0, f(\theta)\right)\right) \theta r' d\theta, \tag{6.6}$$

with cylinder height $s = \sqrt{r^2 - r'^2}$ and $f(\theta) = h - \frac{r'}{\tan(\alpha+\varphi(\theta))}$. A similar derivation leads to a reduction of area A_1 in cases where the sum of the maximal opening angle and α is larger than 90°.

Note that we recompute these critical areas with their reductions in each iteration of our joint optimizations, and that our max-min formulations balance these areas up to equality as long as the constraints allow it. Infeasible designs, such as a socket that cannot hold its ball, are caught by our feasibility constraints. Without user-intervention, we can automatically generate articulated models with spherical *default joints* with constant, global constraint $\varphi(\theta) = \beta$. Our geometric formulations, however, are only approximate models for joint strength and optimality w.r.t. structural strength is not guaranteed. Nevertheless, we avoid weak joints by maximizing their minimal critical cross-section and rejecting them if this section has a value below the global threshold $A_{\min}$. Also, while our two basic joint types lead to output models with sufficient DOFs, our recipe of identifying critical sec-

tions and maximizing their minima is general and applies to other joint designs also.

6.4.3 Fabricating Posable Joints with Friction

From the joints' blue prints (see Figures 6.4.6 and 6.4.9 left) together with device-dependent manufacturing, user-provided range, and estimated joint parameters, we then generate an implicit CSG representation of the volume (in green in Figure 6.4.10) that we have to remove from fabrication mesh $\mathcal{F}$ to introduce a joint at its estimated location. We call this volume *joint hull*. After polygonizing these hulls, we carve them out of $\mathcal{F}$ with mesh-boolean difference operations (see Figure 6.4.10 right), resulting in fabricatable output models with desired kinematics. These models, however, are unlikely to retain a pose once placed into it, and are more like a printed "rag doll." To overcome this limitation, we fabricate small bump spheres of radius r_b onto the positive joint parts similar to [Grey 1999; Wai 2006]. To prevent fusion of movable parts during manufacturing, we extent their ideas by subtracting spheres with same centers but extended radius $r_b + d$ from the negative joint parts also, as illustrated in the top, right corner in Figure 6.4.10. This additional friction mechanism results in posable joints with continuous position control. While these *friction bumps* could potentially stick out of $\mathcal{F}$ after joint carving, we did not observe such cases when estimating our demonstration models. To guarantee embeddedness, we could reduce radii $r_{\max}$ by r_b or, alternatively, invert the bumps and add them to the negative joint parts instead.

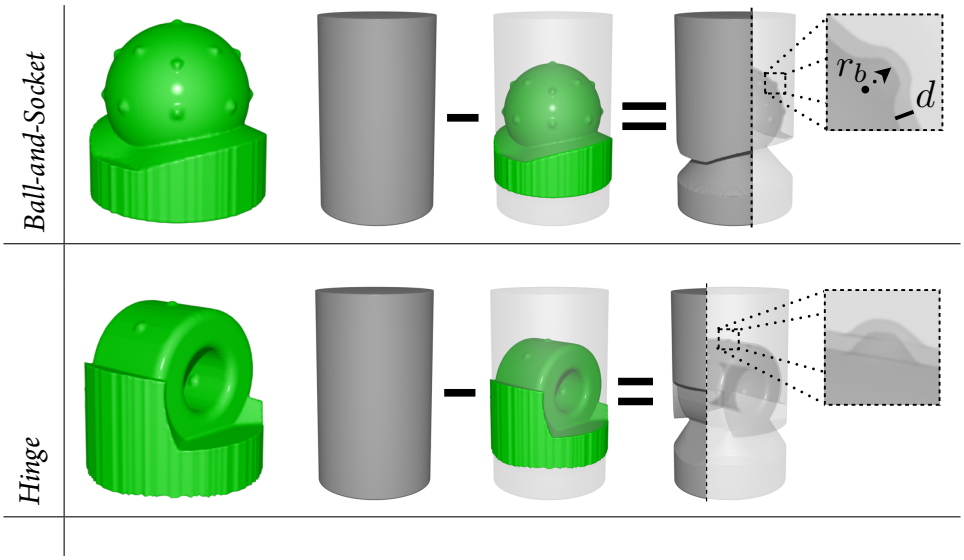


Figure 6.4.10: Frictional Joint Designs based on adding small calibrated bumps. (top) Ball-and-socket joint hull with friction bumps on the ball part and (bottom) hinge joint hull with bumps on the toroidal part. Printed articulated models can then retain their pose.

6.4.4 Avoiding Joint-Joint Collisions

As of now, we can successfully turn simple skins into posable output models, consisting of a set of jointed, rigid pieces that we can print assembled. For sophisticated input skins, however, estimated joint locations are often in close proximity to one another, and, as aforementioned and illustrated in Figure 6.1.2 right, corresponding joint hulls are likely to collide when we maximize the individual joints' sizes. Such overlaps may lead to broken joints, as a closer look at an example of two colliding hulls unveils: if, e.g., a hull volume of one joint contains the part of another spherical joint's socket that prevents its corresponding ball from popping out, we get two disassembled pieces in our output. Hence, we resolve such joint-joint collisions before carving their hulls out of the fabrication mesh $\mathcal{F}$.

In a first naive approach, we could simply remove individual joints, until there are no further hull collisions left. However, while this strategy guarantees functioning joints in our output models, it is not optimal, because we would reject far more of the "fabricatable" input articulation than necessary. A second approach would act directly on what causes the collisions in the first place: the proximity between estimated joint locations. By moving these locations, we could "fit" more joints in $\mathcal{F}$. However, because we set the joints' rotation centers to these locations, this second strategy would significantly change the semantics encoded in our input articulation (if locations were moved away from their corresponding transitions). In the following, we describe our collision resolution procedure that tries to keep as much of the input articulation as possible while avoiding weak joints and keeping their rotation centers fixed. See Figure 6.4.11 and the accompanying video for illustrations.

To initialize our resolution process, we proceed as previously described (Sections 6.4.1, 6.4.2, and 6.4.3). We compute the radius $r_{\max}$ of the maximum inscribed sphere, then optimize a parametric joint model consistent with any user-specified ranges at each estimated location, resulting in a set of joint hulls. Next, we compute all pairwise collisions between these hulls that we inflate by half the distance d, to guarantee a minimal offset between individual joints also. (Note that when we speak of collisions in the following we refer to collisions between such *inflated joint hulls*). To coordinate further processing, we then abstract joint hulls with nodes and pairwise collisions with undirected edges of what we call a *collision graph*. Thereafter, we extract all connected components of this graph with orders

larger than one, and push this *collision groups* onto a *collision stack*. Refer to Figure 6.4.11 (a), where we use the notation C_i^t to uniquely identify each group i at time step t of our resolution.

As long as there are groups on this stack, we pop the topmost and repeatedly reduce the radius $r_{\max}$ for the joint with largest minimal cross section, as it is currently the strongest within this group. We then reestimate its optimal parameters, and check for collisions with its updated joint hull. We stop when either a collision (or several) got resolved, a joint gets infeasible (e.g., a joint's minimal critical area gets smaller than $A_{\min}$), or a joint hull is colliding with a hull outside of its collision group. While such *outside collisions* are rare in practice, it is crucial to check for them, as the example of three spherical joint hulls in the inset figure on the left illustrates. When we reduce the size of the "strongest" of the upper pair of colliding joints, we introduce a second collision with a "node" outside of that group.

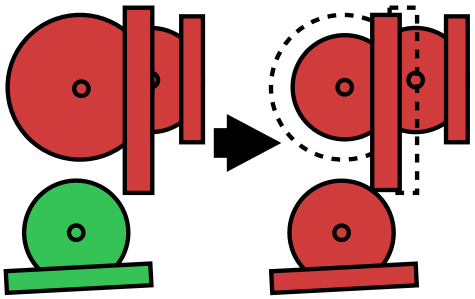


Figure 6.4.12

If collisions got resolved, we are either done (no more collisions within this group) and continue (see Figure 6.4.11 (c)), or split the collision group into subgroups, if necessary, and push those onto the stack. See Figure 6.4.11 (b) for an illustration, where we use $C_{x,j}^t$ to denote the subgroup j with previous *group correspondence history* x. If no split is required (single group), we simply push back C_x^t, without the resolved "edges" and "nodes". However, if a joint becomes infeasible or

a member collides with a joint outside of its collision group, this group is unresolvable without excluding a joint. (Note that while we could add outside collisions to groups or merge groups of the involved members, such "additions" or "merges" may lead to cyclic behavior in our resolution process. Hence, we exclude a joint instead thereby guaranteeing convergence.) We observe that a good candidate for exclusion is given by the member of the current group that was "weakest" after initialization (smallest A_{min}). While this heuristic leads to pleasing output models in practice, this to-be-excluded joint could also be chosen by the user. After an exclusion, we pop all descendants of the original collision group (all groups that have first index k in their correspondence history, if k is the original group's index after initialization), and push the original collision group (k) with reset radii r_{max} and without the excluded joint back onto the stack. Such a reset is necessary because an exclusion of a joint might make previous reductions of joint sizes unnecessary.

Note that our collision resolution process performs evenly well on any other parametric joint designs (other than our hinges and ball-and-sockets from Section 6.4.2) as our collision handling is evaluated on arbitrary hulls, with the only requirement that the joints have to have a single rotation center. Because joints can only get smaller and we exclude a joint if a member gets infeasible or collides with an outside joint, our collision process converges.

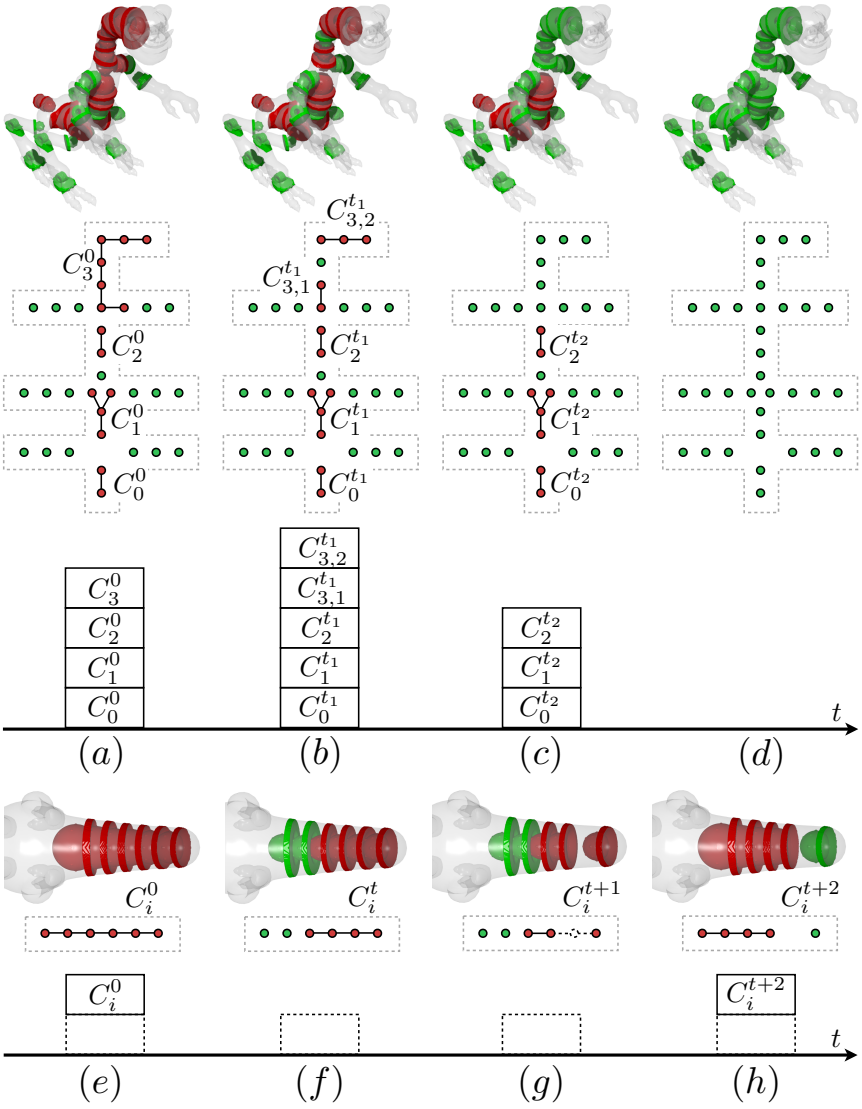


Figure 6.4.11: Resolving Collisions: Colliding joints are shown in red, non-colliding joints in green. For top (a-d) and bottom row (e-h), we have joint hulls on top, corresponding collision graph, and stack in the middle and at the bottom. (a) Initial collision groups for a full character, (b) group split after a resolution, (c) completion of a collision group, and (d) final set of non-colliding joint hulls that we then carve out of $\mathcal{F}$. (e) Initial collision group for a character's tail, (f) a joint gets infeasible (A_{min} too small), (g) exclusion of a joint, (h) updated joint hulls and collisions after a group reset.

6.5 Results

We have created and printed a total of six models based on five skinned characters generated by the SPORE Content Creator ("Grumpy" in Figure 6.1.1, "Chicks" and "Dinofrog" in Figure 6.5.2, "Cristal Frog" and "Lippy" in Figure 6.5.3), and a realistic human hand model that we rigged and skinned in Maya (see Figure 6.5.4). Our five SPORE examples include diffuse and normal maps, and joints were carved out of their colored fabrication meshes, whose geometric detail we computed by inverting normal mapping [Nehab et al. 2005]. This inversion leads to significant quality improvements in $\mathcal{F}$, hence, also in our printouts, as illustrated on the right with a comparison of input and reconstructed geometry for our "Grumpy" character. All of our articulated output models were printed with an Objet Connex 500 printer. We used three of Objet's hard, plastic-like materials called "VeroBlack" ("Lippy" and "Cristalfrog"), "VeroClear" ("Grumpy," "Chicks," and "Dinofrog"), and "ABS-like digital material" (hand model). While "VeroClear" is transparent and the embedded joints, therefore, visible, the ABS-like material is the structurally strongest (e.g., LEGO is made out of ABS). Objet's support material is gel-like and can be removed with a water-jet.

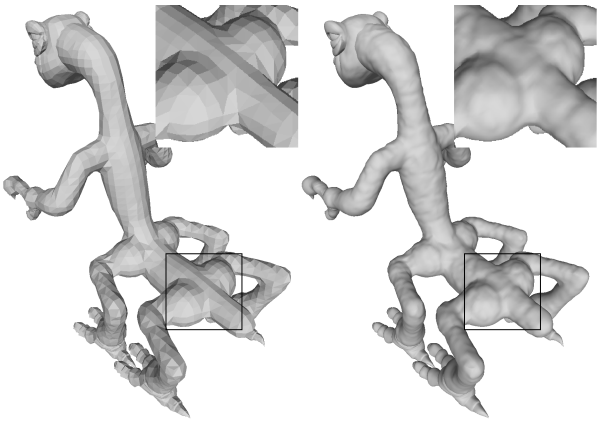


Figure 6.5.1

To identify the minimal offset d to ensure jointed parts to be movable, and the critical area threshold $A_{\min}$ to avoid weak mechanical joints, we estimated hinges and ball-and-sockets for a single-transition cylinder (see Figure 6.4.10 right) with varying radius and for different offsets d, and then printed them with the three printer materials: beyond offsets of 0.3 mm, parts started fusing and the support material could not be water-jetted or "broken out" any longer, and joints with minimal critical areas smaller than 10 mm² for "VeroClear" and "VeroBlack", and 3 mm² for the ABS-like material, started to get brittle. With a similar empirical experiment, we identified a friction bump radius r_b of 0.7 mm. Note that this bump radius is larger than the minimal distance d.

Prior to our articulation estimation, we scaled our input to target sizes (in direction normal to the ground plane shown in gray in Figures 6.1.1, 6.5.2, 6.5.3, 6.5.4) of our output models: 150 mm for "Grumpy" and our hand model, 85 mm for "Chicks," and 100 mm for "Lippy," "Dinofrog," and "Cristalfrog". To filter degenerate transitions, we used factors $f \in [3.5, 4.0]$. Generally, very little user-intervention is needed. E.g., for "Grumpy," the user-intervention was restricted to switching 10 joints to hinge type and specifying three angles each (forward and backward swing angles, and rotation angle around the joint's estimated orientation axis). In addition, we specified spherical range constraints for three neck joints (with again, three user-specified angles each, because we use elliptical opening angles $\varphi(\theta) = \varphi_a \sin\theta + \varphi_b \cos\theta$). All other joints are defaults with global, rotation-invariant range β of a few degrees. With our unoptimized implementation that uses an implicit, extended, regular-grid-based marching cubes approach,

it takes approximately 5.5 hours to process "Grumpy," which is still a fraction of the needed manufacturing time of 18 hours. The time required for processing highly depends on the number of collisions that have to be resolved prior to joint carving. While our SPORE examples had many collisions to resolve, our hand model only had a single collision between two neighboring knuckle joints (overall processing time under 10 min).

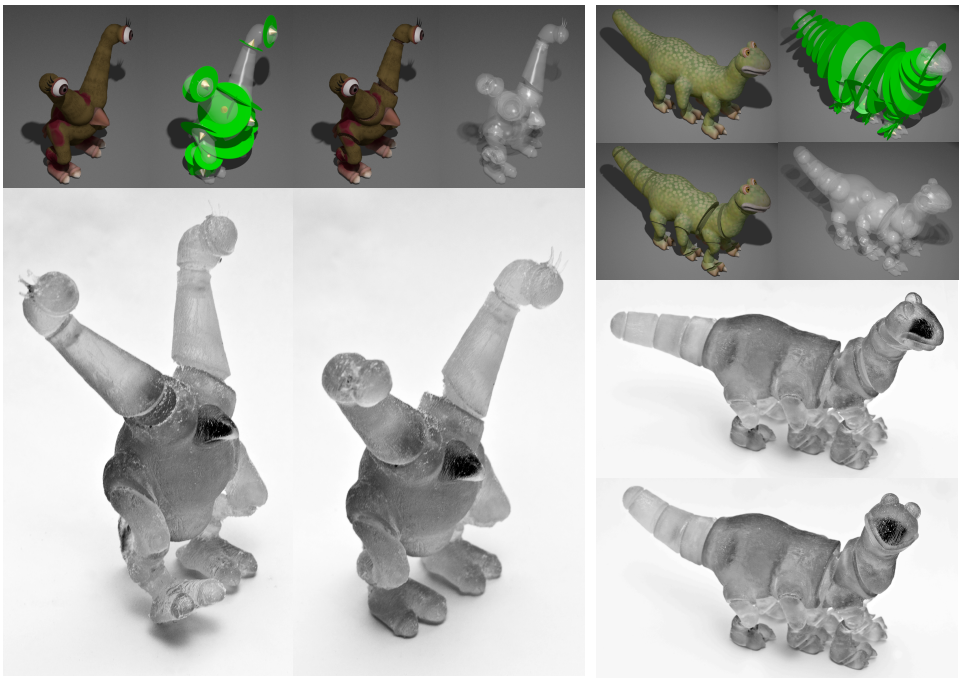


Figure 6.5.2: "Chicks" and "Dinofrog"

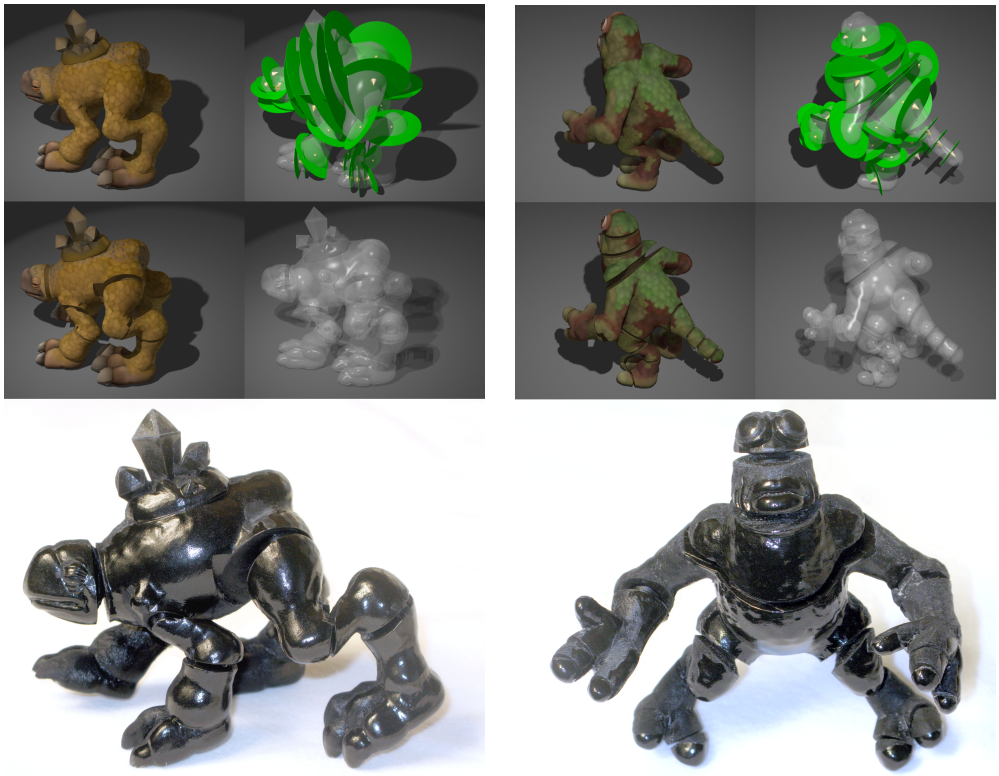


Figure 6.5.3: “Cristal Frog” and “Lippy”

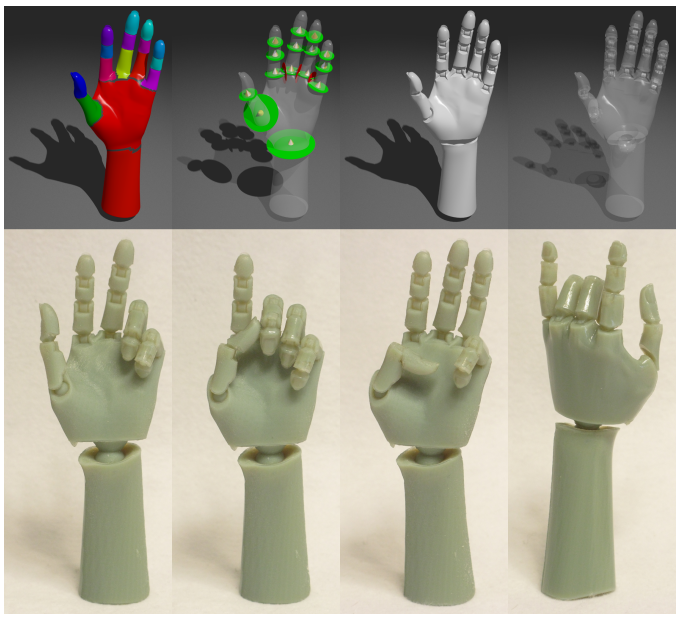


Figure 6.5.4: "Hand"

6.6 Conclusions and Discussion

We have devised a method to generate fabricatable characters from skinned input meshes, e.g., suitable for personalized posable toys. While we are able to generate characters with spherical default joints fully automatically, we allow users to specify joint types and ranges for joints where defaults are not as natural. Note that input skins have transitions where joints are expected, because transitions between joint influences are naturally at places where the model's geometry bends the most during animations. However, while we could always have the user remove unwanted transitions and corresponding joints if there are too many, our system is not able to estimate joints where there is no input data. In the future, we expect that our method and its successors will enable a fully "automatic 3D print button" for characters.

There are several remaining challenges. Current 3D printers introduce many limitations on what we can print. Although our system fully supports colored characters, we were not able to print posable articulated output models in full color. Furthermore, while we avoid weak joints by optimizing parameters of our geometric approximate models of joint strength, our hinge and ball-and-socket designs are not modeling structural strength to a level of accuracy where our system could be fed with a set of measured material parameters to estimate *structurally optimal* joints. As aforementioned, our input skins could also include fine geometric detail with cross sections smaller than $A_{\min}$, or even parts that are completely disconnected from the model's main body, or overlap in the character's rest pose. This

would require to either significantly changing the input geometry (locally inflate geometry, adding artificial connectors, etc.) or rejecting those parts completely. Also, our articulated outputs can be understood as first order, piecewise linear approximate reproductions of the virtual input articulation. Complete piecewise continuous reproductions that include a deformable skin, are left as future work.

In theory there is no difference between theory and practice.

In practice there is.

Yogi Berra

7

Scale-Aware Fabrication

3D geometry is ubiquitous and a fundamental part of practically all CG content. In this chapter, we propose a method for the automated, scale-aware fabrication of an object's static geometry. 3D modeling tools are unaware of manufacturing constraints and their output models are tailored for rendering rather than fabrication. If they contain features too thin and fine, they break during or after 3D printing.

We will discuss a method to abstract such geometric features to ensure that models print correctly, then engrave detail to make sure these features are still perceived in final printouts.

After a brief introduction in the next Section, we examine desired properties of abstraction for manufacturing in Section 7.1. Thereafter, we introduce our calibration part to identify the engraving depth for a targeted AM device, then detail on our abstraction, and engraving in Sections 7.4 and 7.5, respectively. We provide demonstrations of our geometry processing in Section 7.6. In Section 7.7, we summarize and discuss how we plan to extent our processing to support feature-aware, local thickening also.

7.1 Introduction

Recently, affordable desktop printers such as MakerBot's Replicator2 or 3D Systems' Cubify became available allowing us to print custom parts at home at the press of a button. However, the vast majority of 3D modeling tools are unaware of manufacturing constraints and their output models are tailored for rendering rather than fabrication. While detailed 3D models such as the Eiffel tower in Figure 7.1.1 (left) render correctly at any screen resolution and far camera views, fine features are skipped and fuse during or break after printing. Similar to the way 2D printers print a document line after line, a 3D printer builds a given model layer-by-layer. Since models are built in such an additive manner, incorrectly printed or skipped features will effect all the features above it (*error accumulation*).

We present a geometry processing capable of estimating a model tailored for

Figure 7.1.1: Rendering vs. Manufacturing 3D models capturing every hole and protrusion of real-world objects (left) render correctly at any far camera view. We draw inspiration from souvenir manufacturing where detail is manually abstracted and engraved to ensure that miniature models are fabricatable (right).

small-scale manufacturing from a given polygonal mesh. We first represent our input model and its embedding with unions of interior and exterior medial balls. Next, we detect non-fabricatable intrusions by analyzing radii along edges connecting the exterior medial balls, then mark a subset as belonging to the interior. We then extract a watertight, intersection free mesh by identifying the surface between the edited unions of exterior and interior balls. To engrave features, we offset our abstracted model by shrinking interior and growing exterior balls by a device-dependent, calibrated offset parameter. Thereafter, we unify the unions of balls representations of our offset, abstracted model with those representing the original input, resulting in a fabricatable model with preserved detail.

While Stava and colleagues [2012] were the first to present a technique capable of improving the structural strength of weak links in 3D-printable models and Luo

et al.'s [2012] method allows us to divide such models into smaller parts if too large (*scalability*), our method is closer related to model simplification in that it abstracts detail [Mehra et al. 2009]. However, unlike typical mesh simplification [Garland and Heckbert 1997] and LOD representations [Hoppe 1996], we are focusing on manufacturing rather than rendering. Stava et al. [Stava et al. 2012] propose local thickening, besides strut insertion, and hollowing. Thickening of features too thin and fine, however, may lead to blobby output models when targeting small scales on a low resolution printer. In contrast, we draw inspiration from souvenir manufacturing where miniature models with engraved detail are manually designed (see Figure 7.1.1 right).

Manufacturing adjustments to polygonal meshes are challenging because the output models need to be manifold, closed, and self-intersection free. Otherwise, voxel classification is ambiguous and AM devices cannot identify the part of the volume belonging to the interior ("place material"). To this end, we base our geometric processing on the rigorous power crust algorithm by Amenta et al. [2001a; 2001b]. We extent their framework with a *minimal* adaptive Poisson-disk sampling of the input geometry, then analyze the resulting graphs connecting medial ball centers in our abstraction. Our framework allows us to deal with any topological changes during abstraction, while preserving sharp corners and edges in non-critical regions, and guaranteeing that the output is water-tight and intersection free.

7.2 Overview

The major objective behind mesh simplification and LOD is reduction of the overall number of triangles while resembling the original geometry as good as possible when viewed from a far camera. This processing is typically unaware of the enclosed volume, hence, may introduce non-manifoldness [Rossignac and Borrel 1993] and self-intersections [Garland and Heckbert 1997].

In contrast, the main goal in scale-aware manufacturing is to keep as much detail as possible while keeping the model fabricatable. We seek for an abstraction operator capable of "naturally" filling small and deep intrusions. To visually preserve the non-fabricatable detail, we can engrave such indentations. Unlike for mesh simplification, our processing must be volume-aware.

To craft a proper algorithm, we first compile a list of desiderata for model abstraction in a manufacturing context. Before examining this list, we discuss our input. Finally, we restate the required properties of our output.

7.2.1 Input: Geometry with Fine and Thin Features

Our processing takes a detailed polygonal mesh such as, e.g., the Eiffel tower in Figure 7.1.1 (left) as input. While we assume the input itself to be manifold, closed, and self-intersection free, we can pre-process problematic meshes with, e.g., the recent generalized winding numbers [Jacobson et al. 2013]. Without loss of generality, we assume the polygonal faces to be triangles.

7.2.2 Abstracting

We can think of an abstraction as "tighter" convex hull, adaptively filling in concave indentations if narrow and deep as illustrated in Figure 7.2.1 (top, red circles) in 2D. However, concave but wide corners shall be preserved (top, green circles). A well-suited measure capable of differentiating between these cases is *rate of change of medial ball radii*, starting where an intrusion becomes too narrow (top rows, middle).

3D abstraction shall preserve sharp edges, interrupted by an indentation with high fidelity (bottom, upper left). Furthermore, closing shall take the curvature at the boundary of a concavity into account (bottom, upper right). Conceptually, we can achieve 3D abstraction by rolling a ball over such narrow intrusions, steadily connecting the two points where the ball is touching the intrusions' boundaries (bottom, lower left). Moreover, abstraction shall avoid alternating any geometry other than non-fabricatable concavities, especially sharp corners and edges (bottom, lower right).

7.2.3 Output: Geometry with Engraved Detail

Our output models are tailored for additive manufacturing at small scales. To ensure that thin and fine features are still perceived, we engrave them using a calibrated engraving depth (Section 7.3). While we generally avoid weak links in our output models, we cannot give global structural strength guarantees and the technique by Stava and colleagues [Stava et al. 2012] could be used as a post-processing

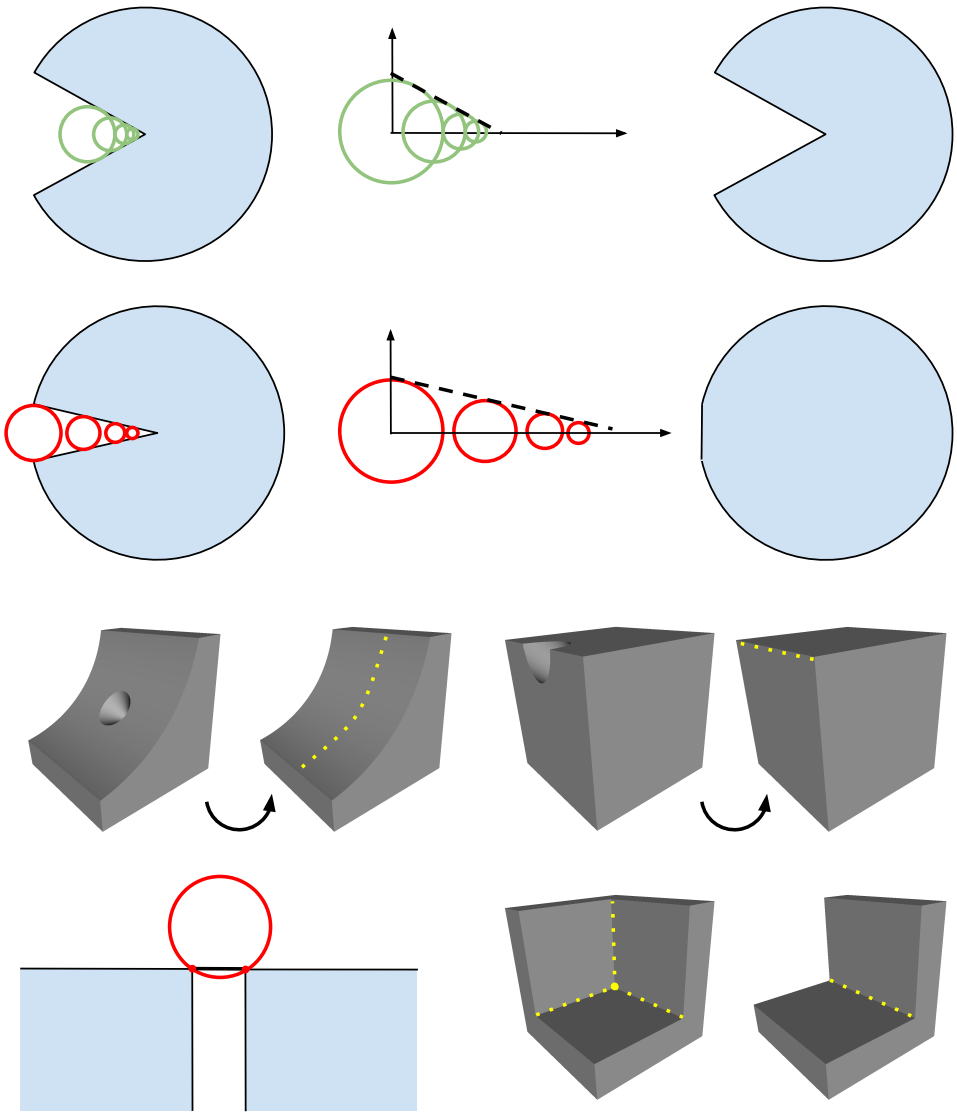


Figure 7.2.1: Abstraction Concavities that are wide (top, green) shall be left unchanged by our abstraction, while narrow and deep ones (top, red) shall be closed. A well-suited measure for differentiating between these cases is rate of change of medial ball radii, starting where intrusions become too narrow (top, middle). Closing shall be aware of interrupted edges (bottom, upper left) and curvature (bottom, upper right) at the boundary of the intrusion. Hence, a well-suited abstraction operator acts like a ball rolling over intrusions, steadily connecting the two points where the ball is touching (bottom, lower right). 3D abstraction shall leave concave corners and edges in non-critical regions unchanged (bottom, lower left)

Figure 7.3.1: Calibrating Engraving Depth We designed a part with cylindrical intrusions of varying depths to identify the minimal engraving depth for a particular printer-material pair.

for models with unusual weight distributions (such as, e.g., a heavy blob attached at the tip of a thin cylindrical support). Note, however, that a physics simulation is tricky because it is almost impossible to simulate all interactions of a person with a printed object. Our technique, while simple, leads to perceptually pleasing results while avoiding complex physics simulations.

We require our output models to describe their enclosed volume unambiguously. Otherwise, they are either rejected by the AM software or printed incorrectly. Specifically, we require our output models to be *free* of *self-intersecting faces*, *manifold*, and *closed*.

7.3 Calibrating Engraving Depth

To identify the *minimal engraving depth* d_{min} for each printer-material pair, we created a calibration part with cylindrical intrusions of varying depths (compare with Figure 7.3.1). Note that d_{min} might vary with orientations, especially on printers with significant differences between vertical and horizontal resolution. We calibrate for the "worst-case" and choose d_{min} such that engraved detail is perceived from all viewing directions. The motivation for such a direction-invariant calibration is two-fold: we do not always have control over the alignment of parts

prior to printing (e.g, when using online services or printers with automatic placement). Furthermore, direction-dependent adjustments generally break symmetries for man-made shapes and, hence, are undesirable from a perceptual point also.

7.4 Abstracting Geometry

We now describe the estimation of an abstraction of a given input geometry. We first review a Voronoi-based medial axis transform (MAT), providing us with a union of balls representation for both, our input and its embedding. We then develop our editing of ball correspondence (interior vs. exterior), resulting in scale-aware abstracted surfaces. To give the reader intuition, we start our discussion with the 2D case, then generalize the developed algorithms to the third dimension.

7.4.1 Representing Geometry using Unions of Medial Balls

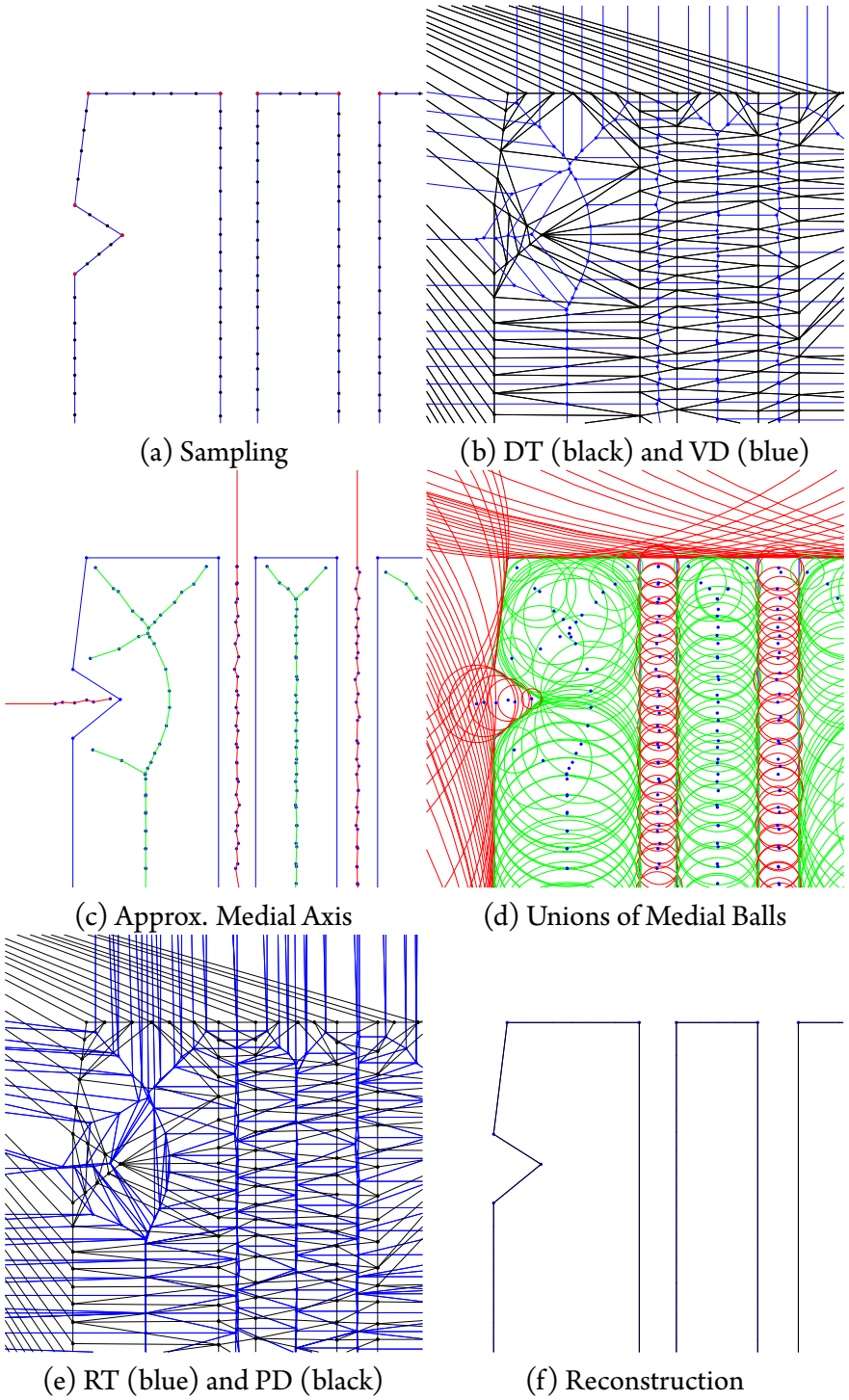
Given a dense enough sampling of a 2D curve, it was Blum [1967] who first observed that a subset of the corresponding Voronoi diagram (VD) provides us with an approximate medial axis. This axis consists of all Voronoi vertices (points closest to more than two samples) and the subset of Voronoi edges (set of points closest to exactly two samples) not crossing the input curve (see Figure 7.4.1 (a-c) for an illustration). It can be shown that the approximate axis of a uniformly increasing sampling converges to the curve's actual medial axis [Schmitt 1989].

If we associate each Voronoi vertex with a radius set to the distance to its closest samples and mark the resulting 2D balls as either interior (green circles in Fig-

ure 7.4.1 (d)) or exterior (red circles), we get a union of medial balls representation (MAT) for both, the interior and exterior of the given curve. (Note that we restrict the exterior of the curve to a bounded, convex region. Hence, all Voronoi cells are finite.)

Given these unions of interior and exterior balls, we can reconstruct the curve by forming the power diagram (PD) of these balls (Figure 7.4.1 (e)). Power diagrams are a generalization of Voronoi diagrams to weighted sites. Specifically, sites with centers and associated radii. While we use the Euclidean metric to determine to which Voronoi cell a given point $\mathbf{p}$ belongs to, we use the power metric when dealing with sites with associated radii: $\mathbf{p}$ belongs to a power cell if the squared Euclidean distance to the site's center minus its squared radius is smaller than for any other site. While the dual of Voronoi diagrams are Delaunay triangulations (DT), the dual of power diagrams are so called regular triangulations (RT). For convenience, we provide the reader with a formal definition, intuition, and algorithms for the construction of both diagrams and their duals in Appendix B. The piecewise linear reconstruction of the curve (see Figure 7.4.1 (f)) is then given by the set of power edges that are dual to edges connecting interior and exterior medial balls in the regular triangulation. The reconstructed curve is interpolating, meaning that the samples are part of the reconstruction.

Most interesting about this algorithm is that it comes with provable guarantees, conditioned on a sampling criterion and an assumption on the smoothness of the sampled curve [Amenta et al. 1998; Amenta and Bern 1998]: Given a twice-differentiable curve $\mathcal{C}$, we call a sampling of $\mathcal{C}$ *ε-sampling* (with ε smaller than 1)



(a) Sampling (b) DT (black) and VD (blue)

(c) Approx. Medial Axis (d) Unions of Medial Balls

(e) RT (blue) and PD (black) (f) Reconstruction

Figure 7.4.1: Unions of Medial Balls: For- and Backward Transformation

if the distance from any point $\mathbf{p} \in \mathcal{C}$ to the nearest sample is at most ε times the distance from $\mathbf{p}$ to the nearest point on the medial axis of $\mathcal{C}$. Intuitively, this criterion requires a denser sampling in regions of high curvature or where other parts of the curve are close. Given a ε-sampling of a smooth $\mathcal{C}$, both, the interior medial axis and the reconstruction are *topologically equivalent* to $\mathcal{C}$ (same connected components, number of loops, etc.). Most relevant in our context, however, is the unconditional guarantee [Amenta et al. 2001a] that the reconstructed curve is manifold, closed, and self-intersection free.

Next, we discuss how we can generate a *minimal*, adaptive Poisson-disk sampling that – in average – fulfills this criterion everywhere but near sharp features. At corners or edges, the medial axis is touching the surface. Hence, we would need an infinite number of samples to satisfy ε-sampling. Given a polygon (piecewise linear curve), we first create a dense pool E_p of uniform edge samples: a sample lies on any particular edge with probability proportional to its length. To efficiently draw edges according to these probabilities (in $O(1)$), we use Vose's alias method [Vose 1991]. After drawing a particular edge with end points $\mathbf{a}$ and $\mathbf{b}$, we place a sample according to $\mathbf{a} + U(\mathbf{b} - \mathbf{a})$ where U denotes a uniform random variable on the unit interval.

To generate a Poisson-disk sampling, we repeatedly pick a sample $\mathbf{s} \in E_p$, then invalidate all samples within a radius ε times the distance r of $\mathbf{s}$ to the closest point on the medial axis, until there are no (valid) samples left in E_p. This sampling, however, does not guarantee that samples are placed onto sharp edge corners. Hence, our reconstruction would perform poorly near such edge ends. Inspired by Corsini

et al.'s work [2012], we add a second pool C_p, containing all edge corners at which $\frac{\pi-a}{\pi} > 0.1$ where a denotes the angle between the two incident edges. During Poisson subsampling, we first draw samples from this second pool C_p, invalidate all samples within a εr-neighborhood from *both* pools, until C_p is empty. Thereafter, we continue sampling from E_p in the manner described above.

What remains is a discussion on how to best choose the radii r used to invalidate neighboring samples in the above procedure. Recall that the radius r associated with a sample $\mathbf{s}$ can at most fall together with the Euclidean distance of $\mathbf{s}$ to the closest point on the medial axis. However, because the medial axis is unknown, we use closeness to an initial approximate axis instead. To this end, we compute the Voronoi diagram of a Poisson sampling with a reasonably small but constant r, then initialize the radii with the distance of the respective sample $\mathbf{s}$ to the closest Voronoi vertex. Because the medial axis is touching the curve at its sharp corners, we also consider closeness to these additional vertices for improved performance. To ensure robustness, we set the radii to values of at least r_{min}, guaranteeing that no two Poisson samples are closer than this minimal radius times ε.

Unfortunately, the above algorithm and theoretical guarantees do not directly translate to the third dimension. As Amenta et al. illustrate in their work ([Amenta et al. 1998], Figure 6), most but not all 3D Voronoi vertices lie close to the medial axis, independent of how densely we sample. However, with a small adjustment, the above construction and guarantees still hold. Specifically, they suggest to only use the subset of Voronoi vertices that are furthest away from its corresponding sample $\mathbf{s}$, one on either side of the input geometry. They call this subset of vertices

poles. The intuition behind this choice is simple: if we sample densely enough, corresponding Voronoi cells are long and skinny. Hence, the Voronoi vertices furthest away from the corresponding sample **s** are most likely the ones closest to the true axis. With this reduced set of Voronoi vertices and corresponding medial balls, the above construction, ε-sampling condition, and guarantees translate to 3D.

Our Poisson-disk sampling for triangular input works in a similar manner than the one for polygons. Instead of one sharp corner pool, we have two: one for edges and one for 3D edge corners. Given an input mesh, we first extract all edges with a normalized angle $\frac{\beta}{\pi} > 0.1$ between the face normals of the two adjacent triangles. Next, we identify the sharp corners in this set of edges in the same manner as described above for polygons. In 3D, we have – in addition to these corner and edge pools – a third pool for triangles, denoted T_p: a sample falls onto a triangle with probability proportional to the triangle's area. Hence, we can again use Vose's method to draw a triangle (with vertices **a**, **b**, and **a**), then generate a uniform sample using $(1-\sqrt{U_1})\mathbf{a}+\sqrt{U_1}(1-\sqrt{U_2})\mathbf{b}+\sqrt{U_1}U_2\mathbf{c}$ with two uniform random variables U_1, U_2 on the unit interval. In contrast, Corsini et al. [Corsini et al. 2012] propose and use heuristics for drawing uniform triangle samples. Unlike their methods, Vose's algorithm allows to generate a sample with 2 table look-ups, independent of the triangles' shapes. During Poisson-disk sampling, we first draw samples from the corner pool, invalidate all samples in an εr-neighborhood from all three pools, until C_p is empty. Analogously, we continue with the edge pool E_p, then with T_p, invalidating samples in both E_p and T_p, then only in T_p. Corsini

et al. [2012] show that the above uniform sampling methodology has blue noise characteristics, hence, is independent of the connectivity of the input mesh. This is crucial when dealing with inputs with sliver triangles. Moreover, the subsampling is extremely efficient when storing the pool samples in a spatial hash.

We applied both, the direct translation from 2- to 3D with the unfiltered set of medial balls (Voronoi-based approach), and Amenta et al.'s pole-based variation [Amenta et al. 2001a] to simple shapes like the star in Figure 7.4.2. While the pole-based clearly outperforms the Voronoi-based approach close to sharp edges and corners (top row, middle column), the latter leads to an overall better quality triangulation (bottom row, right column). When further examining both reconstructions, we observed that the pole-based reconstruction is noisy, especially in flat regions. The reason for this uneven reconstruction are not – as one might first think – numerical instabilities in geometric computations (we ruled this out by switching to an exact kernel supporting both, exact predicates and constructors [CGA]). The problem is, in fact, that the pole-based approach is far more conservative than necessary in that it marks far too many Voronoi vertices as not being part of the approximate axis. As the 2D comparison in Figure 7.4.3 clearly unveils, the pole-based approach filters out a significant fraction of the overall medial balls (compare (c) with (d)). This leads to the formation of new neighbor relationships in the regular triangulation, hence, to dual power edges with end vertices not part of the initial sampling (compare (e) with (f)). While the pole-based approach is still interpolating the samples (this is guaranteed if we keep one ball on either side of a sample), these additional vertices tend to be slightly off the input geometry.

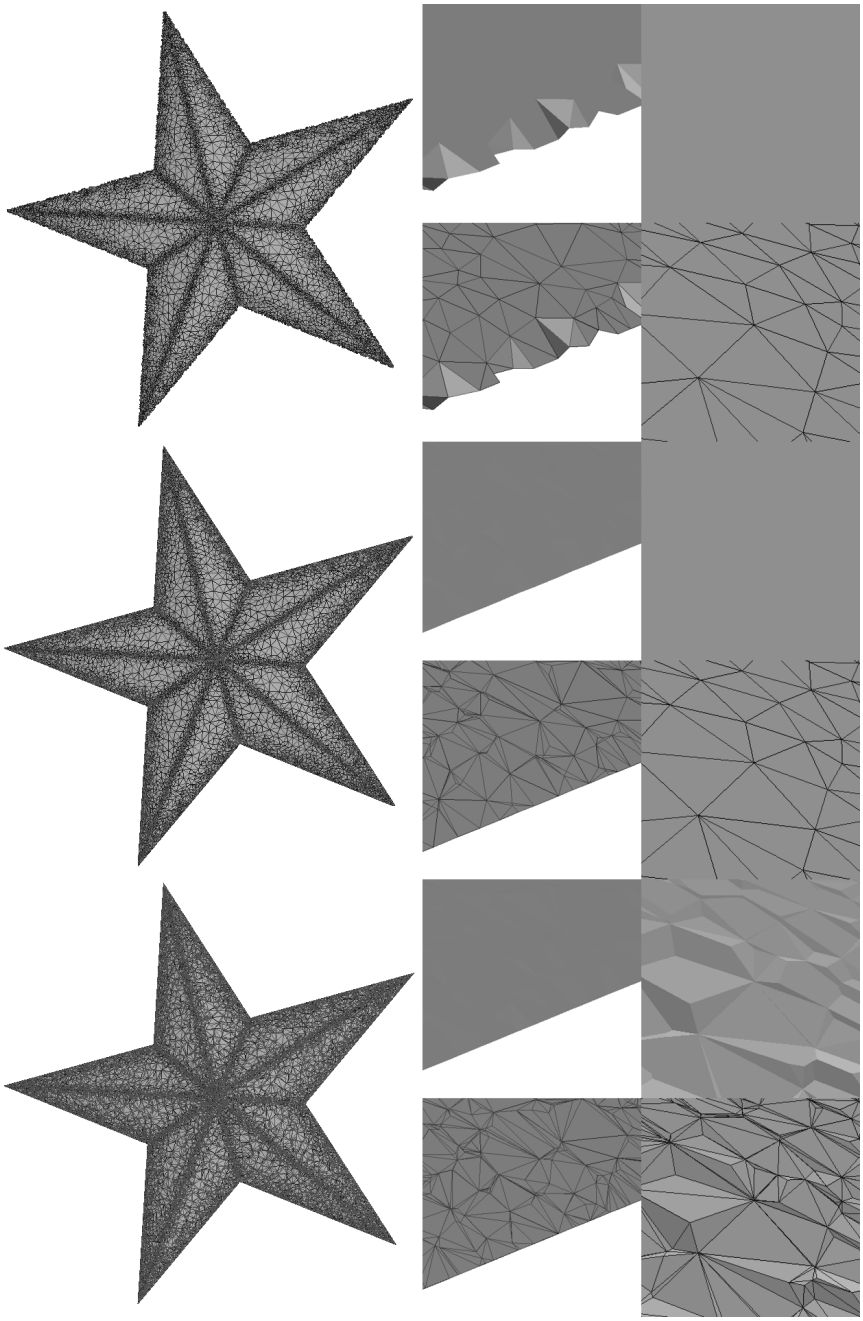
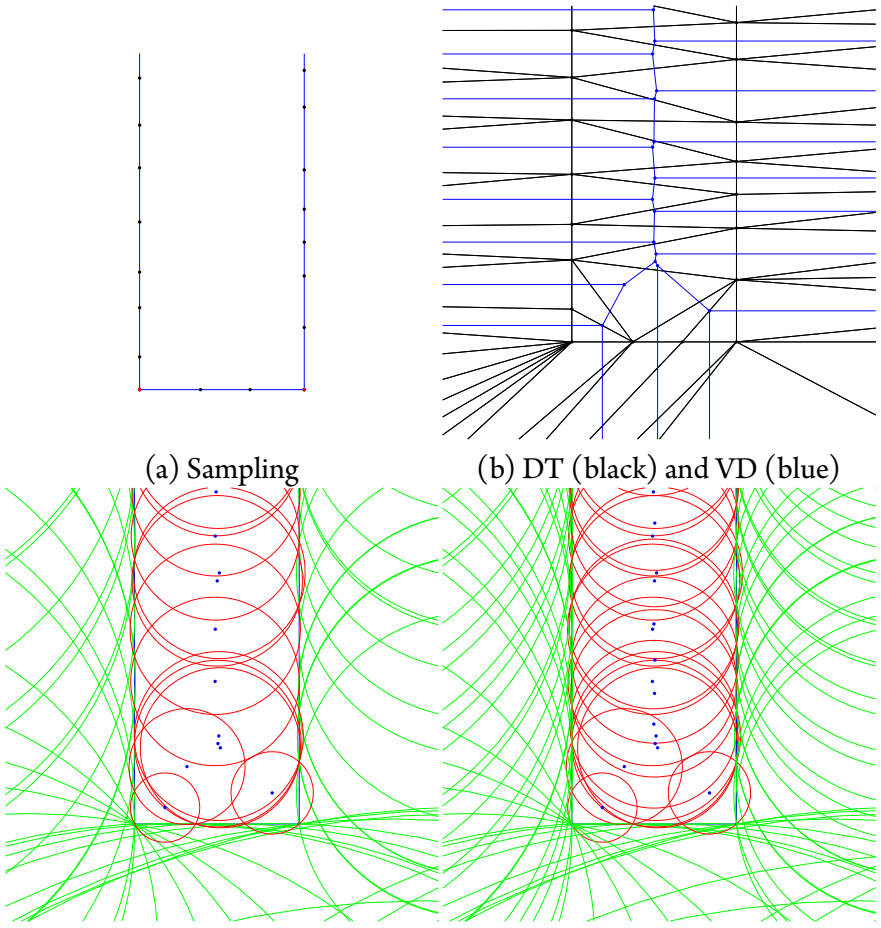


Figure 7.4.2: Reconstructions using the unfiltered set of Voronoi balls (top row), poles and all Voronoi balls with radii larger than a threshold (middle row), and poles only (bottom row). While the pole-based approach performs better on edges (middle column), the unfiltered set leads to a smoother reconstruction in flat regions (right column). Our heuristic (middle row) performs well on both.

These observations suggest the following heuristic for the 3D case: first, identify all poles, then add back all medial balls with radii larger than a (small) threshold. Because small medial balls are only kept if they are poles, we avoid the artifacts near edges of the Voroni-based approach (in the middle column in Figure 7.4.2, compare top with middle row). On the other hand, we keep almost all medial balls in flat regions, leading to a less noisy surface besides a boost in the quality of the resulting triangulation (bottom and middle row, right colum). While we cannot guarantee an error-free reconstruction (errors in vertex positions), we can make errors arbitrarily small by setting the minimal sampling radius r_{min} accordingly. For additive manufacturing, we can choose r_{min} to be smaller than the resolution of the printer.

7.4.2 Abstraction: Alternating Medial Ball Correspondences

The medial axis transform from the previous Section provides us with the ideal building block for our abstraction operations as illustrated in Figure 7.4.4 with a 2D example. We first extract all bridges from the exterior medial axis graph (b) using Tarjan's adopted depth-first-search [1974]. Bridges are edges belonging to trees within a cyclic graph. This step is necessary because our input may consist of several individual curves, leading to several cycles in the exterior axis. Because we bound the exterior, we always have at least one cycle. Next, we detect all connected components, resulting in trees rooted at a cycle, with branches lasting to each individual intrusion of our input (b, red). Thereafter, we iterate over all branches, detecting intrusions where the rate of change of radii along a branch end is slow,



(a) Sampling (b) DT (black) and VD (blue)

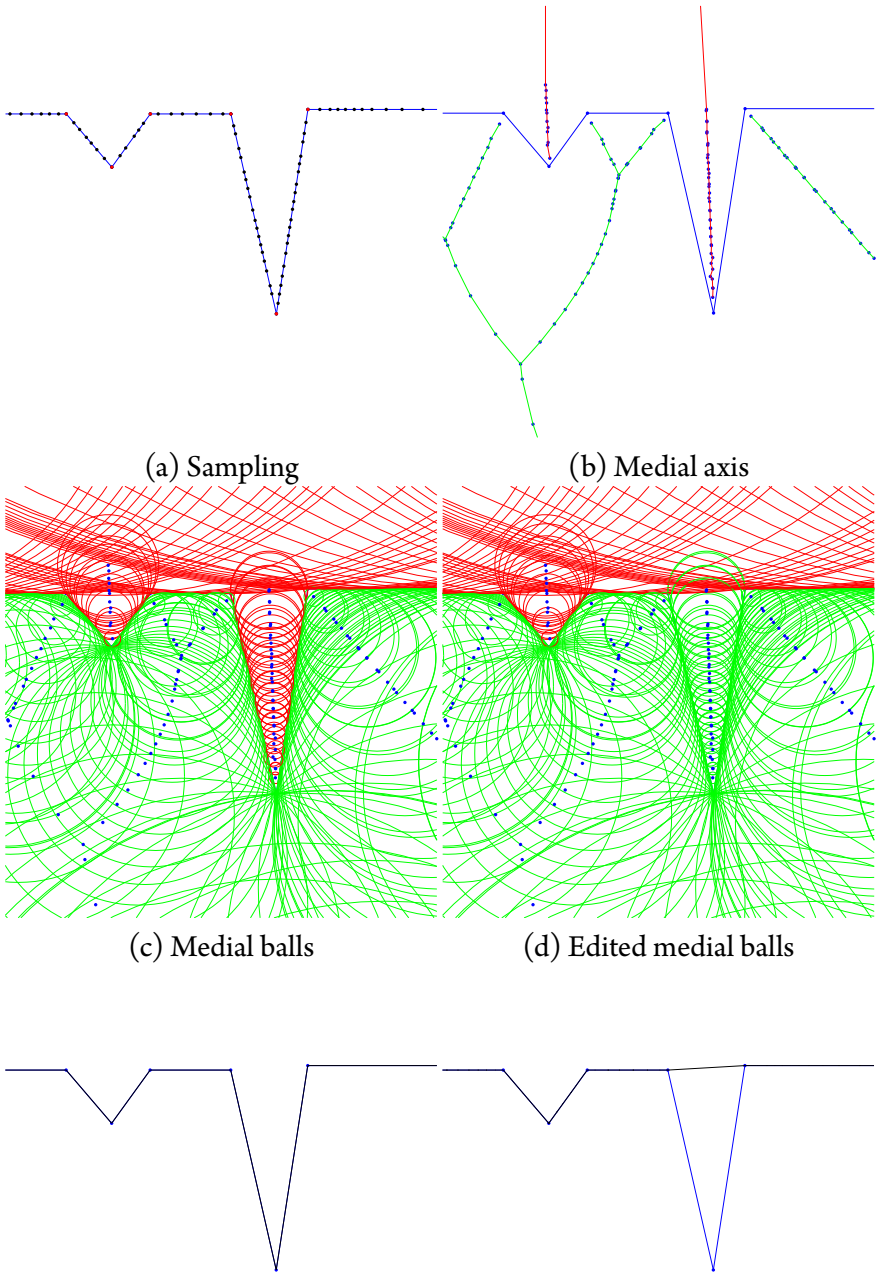
(c) Poles (d) Unfiltered medial balls

(e) Reconstruction (poles) (f) Reconstruction (unfiltered)

Figure 7.4.3: Poles vs. Unfiltered Voronoi Balls (2D)

hence, the intrusion deep. We then mark corresponding medial balls as belonging to the interior (d), followed by identifying the surface between the edited unions of medial balls, resulting in an abstraction (f). Concavities where ball radii are changing quickly, are left untouched.

Unfortunately, our 2D abstraction does not easily translate over to 3D. This is due to the fact that medial axis graphs for 3D inputs consist of curve and surface patches. Because not all 3D Voronoi vertices lie close the medial axis, we only need to consider a subset of the resulting 3D graph. Amenta et al. [2001a] propose the power shape, an approximate axis based on the set of poles, extracted from the power diagram. However, power shapes have unintuitive edges, especially near junctions. We tried several heuristics on both axis representations, with parings of breath- and depth-frist-search strategies for rate of change measures most promising. However, it is too early to report on a best candidate.



(a) Sampling (b) Medial axis

(c) Medial balls (d) Edited medial balls

(e) Reconstruction (f) Reconstruction (edited)

Figure 7.4.4: Abstraction (2D)

7.5 Engraving Detail

For engraving detail, we need two operations: offsetting and set union. With offsetting, we can shrink our abstracted surface by a calibrated engraving depth. With a set union, we can unify the offset abstraction with the original geometry, resulting in an engraved output. While we only discuss engraving in the following, embossing can be done in a similar manner.

7.5.1 Offsetting

For offsetting, we first compute the MAT as we described in Section 7.4.1, then shrink interior and grow exterior balls by our calibrated, device-dependent engraving depth d_{min}. When shrinking, we reject all balls with radii smaller or equal to d_{min}, avoiding sites with negative weights. Thereafter, we extract the offset surface by constructing a regular triangulation and its dual. Note that Amenta and colleagues [2001a] mention this operation as an application of their power crust algorithm.

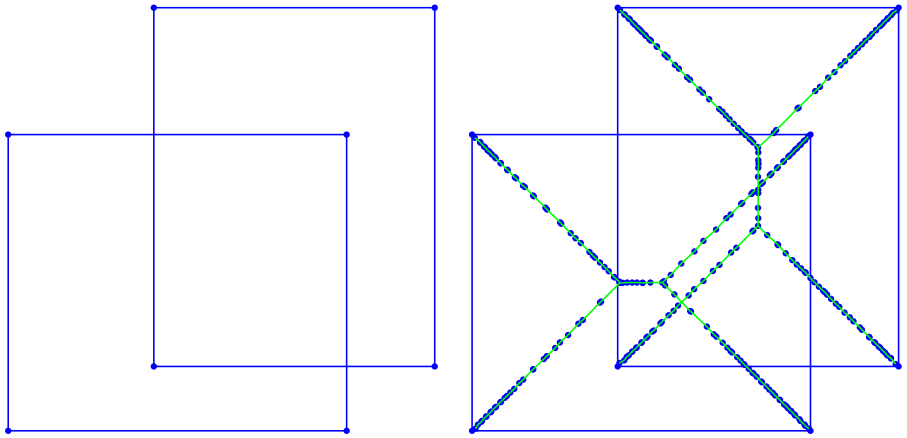
7.5.2 Set Union and other Boolean Operations

Interestingly, we can easily extent our processing to support robust set booleans. Given two input meshes $\mathcal{A}$ and $\mathcal{B}$, we first compute an adaptive Poisson-disk sampling, then compute their MATs. For unions, we reject all samples of $\mathcal{A}$ inside $\mathcal{B}$ and vice versa. We then construct a MAT on the remaining samples from both sets, followed by a reconstruction of the surface between the filtered set of interior

and exterior balls. Similarly, we can compute the mesh bounding the intersection and difference volume of $\mathcal{A}$ and $\mathcal{B}$. Because exterior and interior medial balls intersect only shallowly, we could use our unions of medial balls for inside-outside testing. However, the common ray-mesh intersection counting using an AABB (Axis Aligned Bounding Boxes) tree on the input triangles, is more efficient and avoids unnecessary rejects.

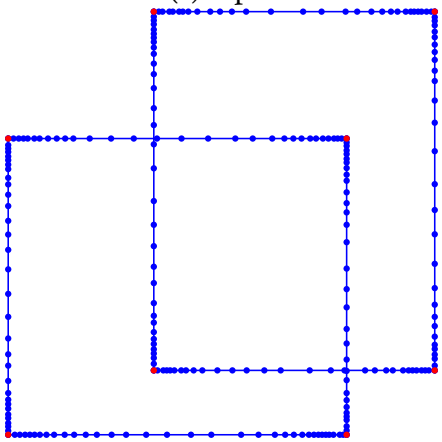
A 2D result of these operations is shown in Figure 7.5.1 (e) for a union of two rectangles. While the proposed processing leads to a high fidelity reconstruction overall, it smooths over a concave corner near one of the two intersection points. This is not surprising because the intersection point is far from both medial axes (b), hence, its immediate neighborhood not sampled densely enough during subsampling (c). To overcome this limitation, we add the intersection points to both corner pools C_p and both sets of Voronoi vertices when initializing the sample radii for $\mathcal{A}$ and $\mathcal{B}$, respectively. This results in an adaptive sampling (d) that leads to a high quality reconstruction everywhere (f). Adding intersection points for the initialization of approximate distances to the respective medial axes is reasonable because the combined axis has branches ending at these points (sharp corners).

Similarly, we compute all intersections between triangles of $\mathcal{A}$ and $\mathcal{B}$ in 3D, then add intersection points to the critical corner pool C_p, and uniform samples of the intersection edges to E_p (we ignore intersecting faces because we sample them anyways). A 3D union of two spheres is shown in Figure 7.5.2.

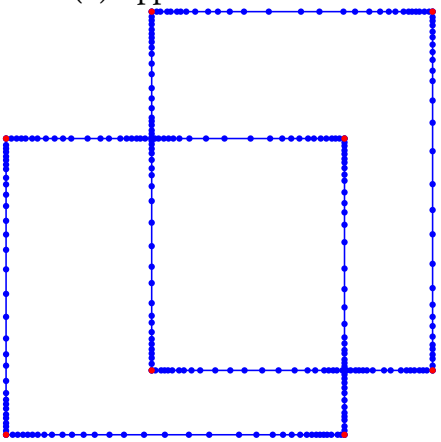


(a) Input

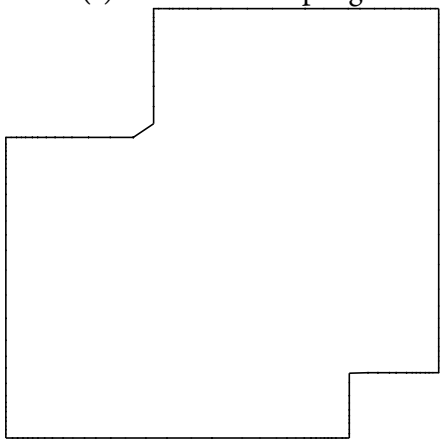
(b) Approx. Medial Axes



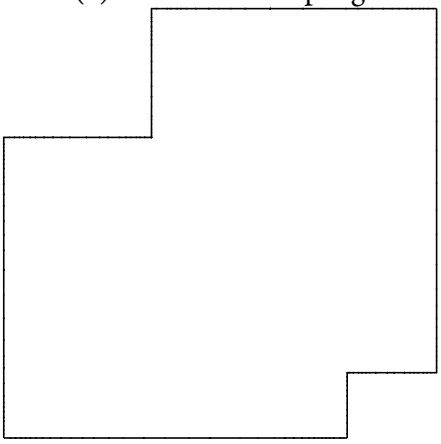
(c) Individual Sampling



(d) Combined Sampling



(e) Union (Individual Sampling)



(f) Union (Combined Sampling)

Figure 7.5.1: Set Union (2D)

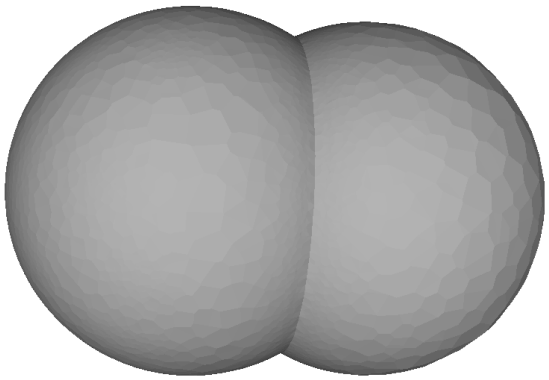
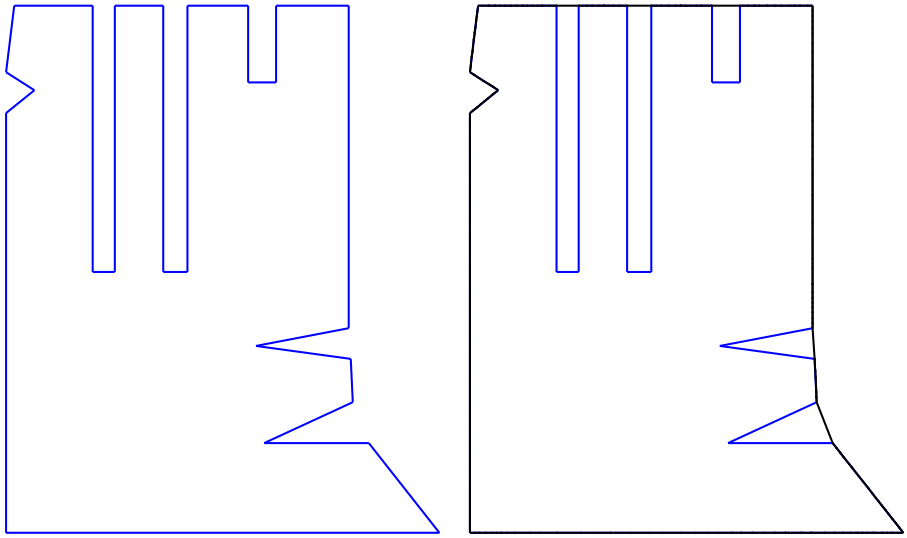


Figure 7.5.2: Union of Spheres

7.6 Demonstrations

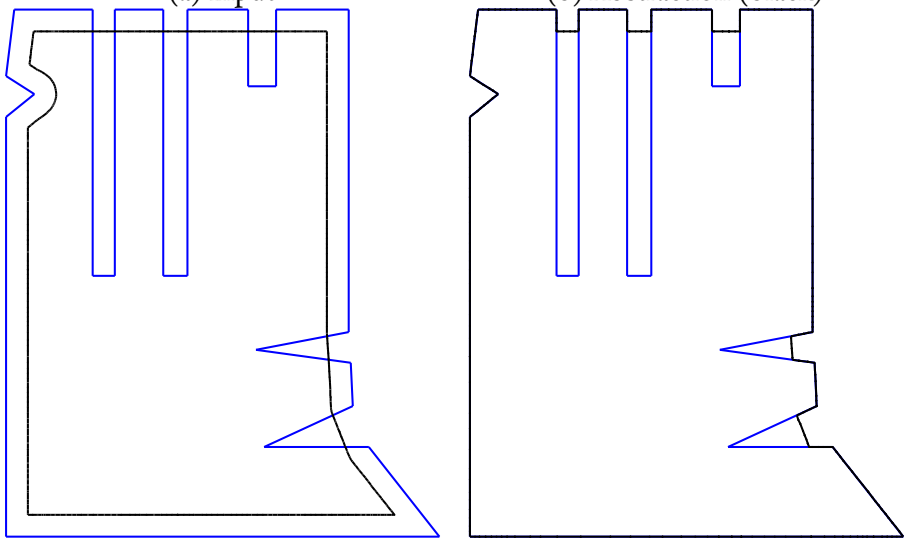
We provide a demonstration of our two-stage processing on a basic example in Figure 7.6.1: we start by abstracting our input (a), closing deep and narrow intrusions (b). In contrast to alpha shapes [Edelsbrunner and Mücke 1994], shallow intrusions are not rounded off and are reconstructed with high fidelity (b, upper left corner). We then shrink interior and grow exterior medial balls, resulting in an offset abstraction (c). With a set union operation with the input, we then achieve engraving (d) of abstracted detail. All sharp corners and low curvature concavities are preserved (d, lower right).

For the Eiffel tower example in Figure 7.6.2, we used edge sampling pools E_p of 20′000 uniform samples. For extraction of the approximate medial axes for initializing adaptive distances for the subsampling of both, our input and abstraction, we used a uniform disk radius of 0.002 times the diagonal of the axis aligned bound box of the tower, then adaptively subsample with r_{min} set to 0.0004 times the di-



(a) Input

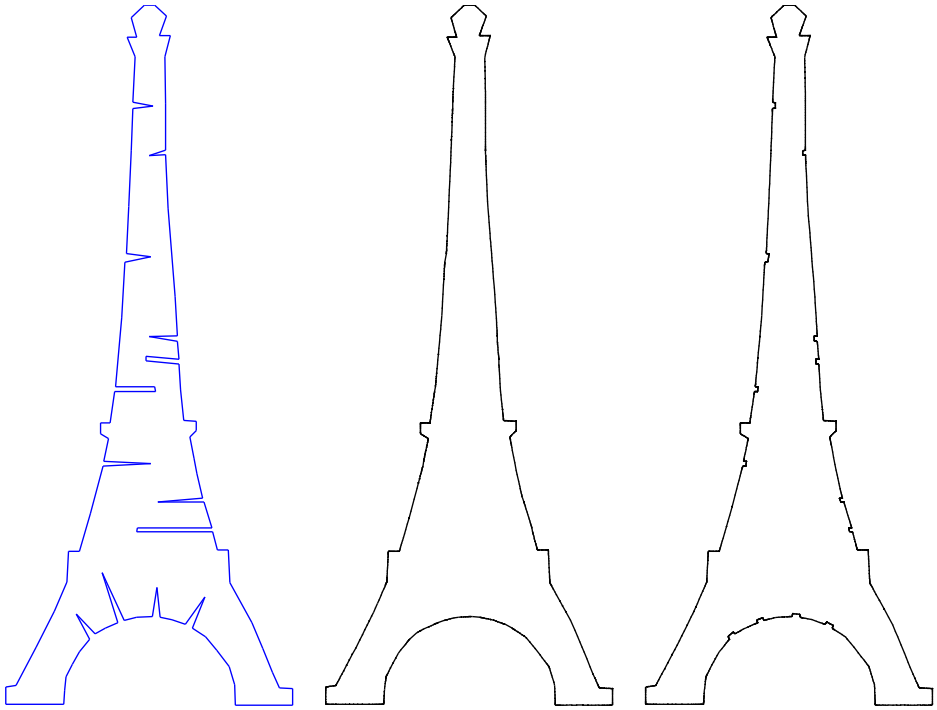
(b) Abstraction (black)



(c) Offset abstraction (black)

(d) Engraved output (black)

Figure 7.6.1: Abstraction and Engraving (2D)



(a) Input (b) Abstraction (c) Engraving

Figure 7.6.2: Abstracting and Engraving Man-Made Shapes (2D)

agonal and an ε of 0.25. For our input, 3′111 Poisson-disk samples were generated, for our union of of input and abstraction, 3′641 and 2′241, respectively. The second sampling of the input is larger because we sample adaptively near intersection points with the abstraction. Because the abstraction contains less concavities, hence, regions with high curvature, we need significantly less samples to represent it with unions of medial balls.

7.7 Discussion and Future Work

We have demonstrated a method capable of abstracting and engraving features too fine and thin, for small-scale, additive manufacturing. While our 3D medial graph analysis needs further refinements, our output models are all "watertight" and self-intersection free – as desired.

Besides a thorough validation of our geometric operators, we plan to extend our processing with local thickening and shrinking operations. As opposed to offsetting, such operators require non-uniform adjustments to ball radii. This is, however, challenging because medial balls may touch at several locations, far apart from each other. When growing or shrinking the surface in one region, we may effect a completely different surface patch. Hence, we would need to carefully sort these conflicts, then copy medial balls, restricting effected regions to those in focus and leaving others unchanged.

We believe that our MAT-based processing has several advantages over methods that rely on other representations. When working directly on surface meshes, we lack volume-awareness. Hence, operations such as offsetting, or closing are extremely challenging, often leading to self-intersecting faces, or non-manifoldness. Implicit surfaces together with a polygonizer such as marching cubes, on the other hand, inherently deal with topological changes and allow us to generate "watertight" output. However, it is unclear how we would implement abstraction, local thickening and shrinking on an implicit representation because it does not come with an inherent medial axis.

The only true wisdom is in knowing you know nothing.

Socrates

8

Conclusion and Future Directions

In this thesis we have explored computational aspects of 3D manufacturing an object's elastic deformation behavior, articulation, and geometry. After a brief summary in Section 8.1, we discuss future directions in Section 8.2.

8.1 Summary

We built two custom stereo vision systems for the non-invasive acquisition of an object's elastic deformation properties, one with a hand-held, one with a robotic contact probe. Our hand-held system is well-suited for the acquisition of force-displacement samples of human soft tissue, while our automated system is tailored for repeatable high precision acquisition of elastic materials.

We devised a data-driven representation and modeling technique that allows us to digitize an object's elastic deformation behavior, thereby simplifying the construction of convincing deformable models by avoiding complex selection and tuning of physical material parameters. Yet, our method retains the richness of non-linear heterogeneous behavior.

We proposed a complete process for the physical reproduction and design of materials with desired deformation behavior. We acquire deformation properties of several printable base material and represent them using a non-linear stress-strain relationship in a finite element model. We then express a desired behavior with stacked layers of base materials using a combinatorial optimization, pruning poor solutions from the search space with a branch-and-bound strategy.

We introduced a technique that facilitates the conversion of virtual articulated models into a fabricatable format. Given a skinned mesh, we estimate posable toy models, consisting of a set of jointed, rigid pieces that we can print assembled. We start by extracting a set of potential joint locations, then maximize minimal cross-sectional areas of hinges and ball-and-sockets, while avoiding non-functional

joints through inter-joint non-penetration and other fabrication constraints.

Lastly, we proposed a technique for the automated fabrication of detailed static geometry. Inspired by souvenir manufacturing, we abstract non-fabricatable features, then engrave them. To this end, we extend a medial axis transform [Amenta et al. 2001a] with a closing operator capable of filling narrow concavities while keeping wide ones untouched. Our output is manifoled, closed, and intersection free.

8.2 Future Directions

We use data-driven physics simulation paired with geometric processing as the fundamental building blocks when solving our computational aspects of 3D manufacturing. For physical reproduction, we acquire properties from real-world objects, simulate them using the finite element method, then estimate models, fabricatable using AM. For the automation of digital content, we estimate physical objects that best approximate the static and dynamic properties of virtual models, while guaranteeing that the output geometry fulfills the requirements for printing.

So far we have explored aspects of manufacturing an object's elastic deformable properties. However, we lack reproduction cycles for many other properties that characterize an object's behavior under motion such as, e.g., plasticity and viscosity. We believe that our data-driven simulation-based approach that first digitizes properties of fabricatable base materials, then expresses a desired behavior within this basis, can be adopted for reproduction of many other properties.

The vast majority of digital 3D content is not directly manufacturable due to the

lack of methods for the automated conversion to fabriatable formats. We estimate piecewise rigid approximate models from the most widely used format in character animation. However, a complete piecewise continuous reproduction with a deformable skin, and automated actuation is left as future work. Our MAT-based geometry processing is volume-aware and is guaranteed to output a "watertight" model. Hence, our framework is well-suited for many other geometry problems in the context of manufacturing.

Lastly, for complete physical reproduction of digital and physical objects, we seek methods for the concurrent acquisition, simulation, and fabrication of several static and dynamic properties. So far our community has looked at many aspects in isolation. However, to reach our goal of building Gershenfeld's personal fabricator [2005], we need to start integrating techniques in a single framework, formulating combined material optimization for competing appearance, shape, and interaction properties.

A

Jacobian for Parameter Fitting

During fitting of material parameters $\mathbf{p} = \{\lambda_e, a_e\}$ through minimization of Equation 4.10, we need to compute the Jacobian of the deformed vertex positions w.r.t. the parameters, i.e., $\mathbf{J} = \frac{\partial \mathbf{x}}{\partial \mathbf{p}}$, in each iteration of the Levenberg-Marquardt algorithm.

Given external forces $\mathbf{F}$ and initial positions $\mathbf{x}_0$, the deformed positions under

the linear co-rotational elastostatic problem [Müller and Gross 2004] are

$$\mathbf{x} = \mathbf{K}^{-1}\left(\mathbf{F} + \mathbf{K}'\mathbf{x}_0\right), \tag{A.1}$$
$$\text{with } \mathbf{K} = \sum_e \left[\mathbf{R}_e\mathbf{K}_e\mathbf{R}_e^T\right]_e \text{ and } \mathbf{K}' = \sum_e \left[\mathbf{R}_e\mathbf{K}_e\right]_e.$$

Here $\left[\ldots\right]_e$ denotes the assembly of the submatrix of the e-th element into the complete stiffness matrix. The Jacobian w.r.t. each parameter $p_i \in \{\lambda_e, a_e\}$ can then be computed as

$$\mathbf{J}_i = \frac{\partial \mathbf{K}^{-1}}{\partial p_i}\left(\mathbf{F} + \mathbf{K}'\mathbf{x}_0\right) + \mathbf{K}^{-1}\frac{\partial \mathbf{K}'}{\partial p_i}\mathbf{x}_0, \tag{A.2}$$
$$\text{with } \frac{\partial \mathbf{K}^{-1}}{\partial p_i} = -\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial p_i}\mathbf{K}^{-1}.$$

Note that we do not compute the inverse of $\mathbf{K}$. Instead, we compute a sparse Cholesky factorization, and then use this factorization many times for solving the linear systems above. Recall the expression for the (unwarped) per-element stiffness matrix in Equation 4.8. The remaining terms are defined as:

$$\frac{\partial \mathbf{K}}{\partial \lambda_e} = \left[V_e\mathbf{R}_e\mathbf{B}_e^T\mathbf{G}\mathbf{B}_e\mathbf{R}_e^T\right]_e, \qquad \frac{\partial \mathbf{K}'}{\partial \lambda_e} = \left[V_e\mathbf{R}_e\mathbf{B}_e^T\mathbf{G}\mathbf{B}_e\right]_e,$$
$$\frac{\partial \mathbf{K}}{\partial a_e} = \left[V_e\mathbf{R}_e\mathbf{B}_e^T\mathbf{H}\mathbf{B}_e\mathbf{R}_e^T\right]_e, \qquad \frac{\partial \mathbf{K}'}{\partial a_e} = \left[V_e\mathbf{R}_e\mathbf{B}_e^T\mathbf{H}\mathbf{B}_e\right]_e. \tag{A.3}$$

In case some nodes are constrained not to deform (e.g., when the bottom of the captured objects is fixed), their known positions move to the right-hand side in Equation A.1, and the Jacobians must be slightly modified.

B

Voronoi, Power Diagrams, and their Duals

In this section, we review Voronoi and power diagrams and their duals in $\mathbb{R}^d$ and summarize how we can easily compute them using a convex hull algorithm in $d+1$ dimensions. While we keep the description of the algorithm general, we fall back to the 3D-case ($d = 3$) when discussing robustness aspects.

B.1 Voronoi and Power Diagrams

Voronoi diagrams are named after mathematician Georgy Voronoy who first described them in their general form [1908]: Given a set $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ of n distinct points (called *sites*) in a bounded, convex region $\Omega \subset \mathbb{R}^d$, the corresponding Voronoi diagram $\mathcal{VD}$ divides Ω into n convex regions (called *Voronoi cells*), each consisting of all points closest to a particular site

$$\mathcal{VD}(\mathbf{p}_i) = \left\{\mathbf{q} \in \Omega : \forall \mathbf{p}_j \neq \mathbf{p}_i : d(\mathbf{q}, \mathbf{p}_i) < d(\mathbf{q}, \mathbf{q}_j)\right\} \tag{B.1}$$

where $d(\mathbf{a}, \mathbf{b})$ denotes the Euclidean distance between $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$.

Voronoi diagrams under more general distance metrics and for more general objects than points have been developed, among which *power diagrams* $\mathcal{PD}$ are most similar to the original diagrams: given a sphere $s \subset \mathbb{R}^d$ with center $\mathbf{p}$ and radius r, the power p of a point $\mathbf{q}$ w.r.t. s is given by $d^2(\mathbf{q}, \mathbf{p}) - r^2$. The power metric allows a simple geometric interpretation for $\mathbf{q}$ outside of s: p is the squared length of the line segment from $\mathbf{q}$ and a point tangent to s. For points at arbitrary position, the power of $\mathbf{q}$ is positive if $\mathbf{q}$ is outside, zero if on, and negative if inside s. Formally, given a set $S = \{s_1, \ldots, s_n\}$ of n distinct spheres, the power diagram is defined by

$$\mathcal{PD}(s_i) = \left\{\mathbf{q} \in \Omega : \forall s_j \neq s_i : p(\mathbf{q}, s_i) < p(\mathbf{q}, s_j)\right\} \tag{B.2}$$

Note that the power cell corresponding to a sphere s can be empty if s is contained in the union of the balls bounded by the remaining spheres. This condition,

however, is not sufficient [Aurenhammer 1987].

B.2 Dual Correspondence to Convex Hulls

As first described by Brown [1979] and later refined by Edelsbrunner and colleagues [1986], a Voronoi diagram in $\mathbb{R}^d$ can be computed using a transformation to the $d+1$ dimensional space and a convex hull algorithm therein. All relevant properties of Voronoi diagrams naturally carry over to power diagrams as described by Aurenhammer in [1987], which allows a natural extension of Brown's construction to power diagrams also. Because power diagrams fall together with Voronoi diagrams if all radii are equal, we only discuss Aurenhammer's construction here. Setting all radii to zero, we directly get the construction for Voronoi diagrams.

"Lifting" from $\mathbb{R}^d$ to $\mathbb{R}^{d+1}$: The construction relies on the bijective mapping Π of the power of a sphere $s = (\mathbf{p}, r)$ in $\mathbb{R}^d$ to a hyperplane in $\mathbb{R}^{d+1}$

$$\Pi(s) : x_{d+1} = 2\mathbf{p}\mathbf{x} - \mathbf{p}^2 + r^2 \tag{B.3}$$

where $\mathbf{x} = (x_1, \ldots, x_d)$, and stems from the observation that the power distance of a point $\mathbf{q}$ w.r.t. s is given by the difference of the segments between $\mathbf{q}$ and its vertical projections $\mathbf{q}'$ and $\mathbf{q}''$ onto the paraboloid $x_{d+1} = \mathbf{x}^2$ and $\Pi(s)$, respectively. From this observation, it follows that the vertical projection of the intersection $\Pi(s_i) \cap \Pi(s_j)$ onto $\mathbb{R}^d$ separates the points closest to s_i from those closest to s_j under p. Hence, we are interested in the vertical projection of the boundary of the

intersections of all upper halfspaces delimited by these hyperplanes.

Duality in $\mathbb{R}^{d+1}$: Using geometric duality, we map each hyperplane h in its general form $x_{d+1} = \mathbf{a}\mathbf{x} + a_{d+1}$ with $\mathbf{a} = (a_1, \ldots, a_d)$ to a point in dual space

$$\Delta(h) = \left(\frac{1}{2}\mathbf{a}, -a_{d+1}\right), \tag{B.4}$$

thereby recasting our upper halfspace intersection as a lower convex hull problem in dual space. Δ is known as the *polarity* function and the resulting point as *pole*. Concatenating our "lifting" and duality transforms, we directly get the dual set S^* of poles

$$\Delta(\Pi(s)) = (\mathbf{p}, \mathbf{p}^2 - r^2) = (\mathbf{p}, h_\mathbf{p}) \tag{B.5}$$

on which we compute the convex hull $\mathcal{CH}$. The boundary of $\mathcal{CH}$ naturally splits into a lower (part facing the $\mathbb{R}^d$ hyperplane) and an upper part and the vertical projection of the lower boundary (ignore the $d+1$-th component of each point), falls together with the regular triangulation $\mathcal{RT}$ dual to $\mathcal{PD}$.

To get the power diagram from $\mathcal{RT}$, we map the result back to primal space. While the combinatorial part is straightforward (vertices map to cells, faces to edges, edges to faces, and cells to vertices), the vertex positions need a careful treatment. To this end, we fall back onto the 3D case ($d = 3$) considered here and give geometrically robust formulas for both the Voronoi and power diagram case.

Circum- and Orthospheres [Shewchuk 1997; Schewchuk 2009]: Given four dual 3D points with their heights $(\mathbf{a}, h_\mathbf{a})$, $(\mathbf{b}, h_\mathbf{b})$, $(\mathbf{c}, h_\mathbf{c})$, and $(\mathbf{d}, h_\mathbf{d})$, delimiting a

tetrahedron, the corresponding orthosphere center $\mathbf{o}$ and squared radius $r_{\mathbf{o}}^2$ are

$$\frac{(h_a - h_d)\mathbf{u} \times \mathbf{v} + (h_b - h_d)\mathbf{v} \times \mathbf{t} + (h_c - h_d)\mathbf{t} \times \mathbf{u}}{12V} \tag{B.6}$$

and

$$(\mathbf{od})^2 + (h_{\mathbf{d}} - \mathbf{d}^2) \tag{B.7}$$

where $\mathbf{u} = \mathbf{b} - \mathbf{d}$, $\mathbf{v} = \mathbf{c} - \mathbf{d}$, $\mathbf{t} = \mathbf{a} - \mathbf{d}$, and V denotes the volume of the tetrahedron.

In the Voronoi case, where all radii and, hence, all heights are zero, we compute the circumsphere center $\mathbf{c}$ and radius $r_{\mathbf{c}}$ using

$$\mathbf{d} + \frac{\mathbf{t}^2(\mathbf{u} \times \mathbf{v}) + \mathbf{u}^2(\mathbf{v} \times \mathbf{t}) + \mathbf{v}^2(\mathbf{t} \times \mathbf{u})}{12V} \tag{B.8}$$

and

$$\frac{||\mathbf{t}^2(\mathbf{u} \times \mathbf{v}) + \mathbf{u}^2(\mathbf{v} \times \mathbf{t}) + \mathbf{v}^2(\mathbf{t} \times \mathbf{u})||}{12V}. \tag{B.9}$$

All of the above constructions can be made robust using Shewchuk's adaptive predicates [1997]. For convex hull computations, we suggest using Clarkson's dynamic algorithm [1993].

References

CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

J-P. ABBAT. Articulated doll joint. U.S. Patent 5257873, 1993.

MARC ALEXA AND WOJCIECH MATUSIK. Reliefs as images. *ACM Trans. Graph.*, 29:60:1–60:7, July 2010.

BRETT ALLEN, BRIAN CURLESS, AND ZORAN POPOVIĆ. Articulated body deformation from range scan data. *ACM Trans. Graph.*, 21(3):612–619, 2002.

MALONE E. MOON F.C. LIPSON H. ALONSO, M.P. Reprinting the telegraph: Replicating the vail register using multi-materials 3d printing. In *Solid Freeform Fabrication Symposium (SFF'09)*, 2009.

NINA AMENTA AND MARSHALL BERN. Surface reconstruction by voronoi filtering. *Discrete and Computational Geometry*, 22:481–504, 1998.

NINA AMENTA, MARSHALL BERN, AND MANOLIS KAMVYSSELIS. A new voronoi-based surface reconstruction algorithm. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 415–421, New York, NY, USA, 1998. ACM.

NINA AMENTA, SUNGHEE CHOI, AND RAVI KRISHNA KOLLURI. The power crust. In *Proceedings of the sixth ACM symposium on Solid modeling and applications*, SMA '01, pages 249–266, New York, NY, USA, 2001a. ACM.

NINA AMENTA, SUNGHEE CHOI, AND RAVI KRISHNA KOLLURI. The power crust, unions of balls, and the medial axis transform. *Computational Geometry*, 19(2-3):127 – 153, 2001b.

DOMINIQUE ATTALI, JEAN-DANIEL BOISSONNAT, AND HERBERT EDELSBRUNNER. Stability and computation of medial axes - a state-of-the-art report. In

Torsten Müller, Bernd Hamann, and RobertD. Russell, editors, *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, Mathematics and Visualization, pages 109–125. Springer Berlin Heidelberg, 2009.

Marco Attene, Marcel Campen, and Leif Kobbelt. Polygon mesh repairing: An application perspective. *ACM Comput. Surv.*, 45(2):15:1–15:33, March 2013.

F Aurenhammer. Power diagrams: properties, algorithms and applications. *SIAM J. Comput.*, 16(1):78–96, 1987.

Ilya Baran and Jovan Popović. Automatic rigging and animation of 3D characters. *ACM Trans. Graph.*, 26(3):72:1–72:8, July 2007.

Ilya Baran, Philipp Keller, Derek Bradley, Stelian Coros, Wojciech Jarosz, Derek Nowrouzezahrai, and Markus Gross. Manufacturing layered attenuators for multiple prescribed shadow images. *Comp. Graph. Forum*, 31(2pt3):603–610, May 2012.

Klaus Jürgen Bathe. *Finite Element Procedures*. Prentice Hall, 1995.

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

Ferdinand Beer, E. Russell Johnston, John T. DeWolf, and David F. Mazurek. *Mechanics of Materials*. McGraw-Hill, 6 edition, 2011.

Martin Philip Bendsoe and Ole Sigmund. *Topology Optimization*. Springer Berlin, 2003.

P. Bergeron and P. Lachapelle. Controlling facial expression and body movements in the computer generated short "Tony de Peltrie". Siggraph Course Notes, 1985.

Amit Bermano, Ilya Baran, Marc Alexa, and Wojciech Matusk. Shadowpix: Multiple images from self shadowing. *Comp. Graph. Forum*, 31(2pt3): 593–602, May 2012.

Bernd Bickel, Mario Botsch, Roland Angst, Wojciech Matusik, Miguel Otaduy, Hanspeter Pfister, and Markus Gross. Multi-scale capture of facial geometry and motion. *ACM Trans. Graph.*, 26(3):33, 2007.

Bernd Bickel, Manuel Lang, Mario Botsch, Miguel A. Otaduy, and Markus Gross. Pose-space animation and transfer of facial details. In *Proc. of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 57–66, 2008.

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 Trans. Graph.*, 31(4):118:1–118:10, July 2012.

V. Blanz, C. Basso, T. Poggio, and T. Vetter. Reanimating faces in images and video. *Computer Graphics Forum*, 22(3):641–650, September 2003.

Harry Blum. A Transformation for Extracting New Descriptors of Shape. In Weiant Wathen-Dunn, editor, *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, Cambridge, 1967.

M. Botsch and O. Sorkine. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 14 (1):213–230, 2008.

Jean-Yves Bouguet. Camera calibration toolbox for matlab, http://www.vision.caltech.edu/bouguetj/calib_doc, 2006.

Kevin Q. Brown. Voronoi diagrams from convex hulls. *Information Processing Letters*, 9(5):223 – 228, 1979.

Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proc. of ACM SIGGRAPH*, pages 425–432. ACM, 2001.

S. Burion, F. Conti, A. Petrovskaya, C. Baur, and O. Khatib. Identifying physical properties of deformable objects by using particle filters. In *Proc. of the International Conference on Robotics and Automation*, pages 1112–1117, 2008.

Jiri Bursa and Miroslav Zemanek. Evaluation of biaxial tension tests of soft tissues. *Studies in health technology and informatics*, 133:45, 2008.

Jacques Calì, Dan A. Calian, Cristina Amati, Rebecca Kleinberger, Anthony Steed, Jan Kautz, and Tim Weyrich. 3d-printing of non-assembly, articulated models. *ACM Trans. Graph.*, 31(6):130:1–130:8, November 2012.

S. Capell, M. Burkhart, B. Curless, T. Duchamp, and Z. Popović. Physically based rigging for deformable characters. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 301–310, 2005.

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 *Proc. of ACM SIGGRAPH*, pages 67–76, 2001.

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 SIGGRAPH 2013, Anaheim, CA)*, 32(4):135:1–135:10, 2013a.

Desai Chen, Pitchaya Sitthi-amorn, Justin T. Lan, and Wojciech Matusik. Fabrication-aware design with intersecting planar pieces. *Computer Graphics Forum (Proceedings of Eurographics 2013)*, 32(2), 2013b. to appear.

James H. Clark. Hierarchical geometric models for visible surface algorithms. *Commun. ACM*, 1976.

Kenneth L. Clarkson, Kurt Mehlhorn, and Raimund Seidel. Four results on randomized incremental constructions. *Computational Geometry*, 3(4):185 – 212, 1993.

Jonathan Cohen, Amitabh Varshney, Dinesh Manocha, Greg Turk, Hans Weber, Pankaj Agarwal, Frederick Brooks, and William Wright. Simplification envelopes. *ACM Trans. Graph.*, 1996.

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 ACM SIGGRAPH)*, 32(4):to appear, 2013.

Massimiliano Corsini, Paolo Cignoni, and Roberto Scopigno. Efficient and flexible sampling with blue noise properties of triangular meshes. *Visualization and Computer Graphics, IEEE Transactions on*, 18(6):914–924, 2012.

P. DiLorenzo, V. Zordan, and B. Sanders. Laughing Out Loud: Control for modeling anatomically inspired laughter using audio. *ACM Trans. Graph. (Proc. of ACM SIGGRAPH Asia)*, 27(5), 2008.

Yue Dong, Jiaping Wang, Fabio Pellacini, Xin Tong, and Baining Guo. Fabricating spatially-varying subsurface scattering. *ACM Trans. Graph.*, 29: 62:1–62:10, July 2010.

Herbert Edelsbrunner and Ernst P. Mücke. Three-dimensional alpha shapes. *ACM Trans. Graph.*, 13(1):43–72, January 1994.

Herbert Edelsbrunner and Raimund Seidel. Voronoi diagrams and arrangements. *Discrete & Computational Geometry*, 1(1):25–44, 1986.

R. Ferre. Form of articulated structures for dolls or puppet bodies. U.S. Patent 6033284, 2000.

N. Galoppo, M. A. Otaduy, W. Moss, J. Sewall, S. Curtis, and M. C. Lin. Controlling deformable material with dynamic morph targets. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 2009.

Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. *ACM Comput. Graph.*, 1997.

Neil Gershenfeld. *Fab: The Coming Revolution on Your Desktop–from Personal Computers to Personal Fabrication*. Basic Books, 2005.

M. J. Grey. Construction system. U.S. Patent 5897417, 1999.

E. W. Hart. Theory of the tensile test. *Acta Metallurgica*, 15:351–355, 1967.

Miloš Hašan, Martin Fuchs, Wojciech Matusik, Hanspeter Pfister, and Szymon Rusinkiewicz. Physical reproduction of materials with specified subsurface scattering. *ACM Trans. Graph.*, 29:61:1–61:10, July 2010.

Chris Hecker, Bernd Raabe, Ryan W. Enslow, John DeWeese, Jordan Maynard, and Kees van Prooijen. Real-time motion retargeting to highly varied user-created morphologies. *ACM Trans. Graph.*, 27(3):27:1–27:11, August 2008.

Kristian Hildebrand, Bernd Bickel, and Marc Alexa. crdbrd: Shape fabrication by sliding planar slices. *Comp. Graph. Forum*, 31(2pt3):583–592, May 2012.

Jonathan Hiller and Hod Lipson. Design and analysis of digital materials for physical 3d voxel printing. *Rapid Prototyping Journal*, 15:137–149, 2009.

Derek Hoiem, Alexei A. Efros, and Martial Hebert. Automatic photo pop-up. *ACM Trans. Graph.*, 24(3):577–584, July 2005.

Michael Holroyd, Ilya Baran, Jason Lawrence, and Wojciech Matusik. Computing and fabricating multilayer models. *ACM Trans. Graph.*, 30(6): 187:1–187:8, December 2011.

Hugues Hoppe. Progressive meshes. *ACM Trans. Graph.*, 1996.

Berthold K.P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. of the Optical Society of America*, 4(4):629–642, 1987.

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

Charles W. Hull. Apparatus for the production of three-dimensional objects by stereolithography. U.S. Patent 4575330 A, 1986.

Alec Jacobson, Ladislav Kavan, , and Olga Sorkine-Hornung. Robust inside-outside segmentation using generalized winding numbers. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 32(4):to appear, 2013.

Doug L. James and Dinesh K. Pai. ArtDefo: Accurate real time deformable objects. In *Proc. of ACM SIGGRAPH*, pages 65–72. ACM Press/Addison-Wesley Publishing Co., 1999.

Doug L. James and Christopher D. Twigg. Skinning mesh animations. *ACM Trans. Graph.*, 24(3):399–407, August 2005.

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.

M. Kauer, V. Vuskovic, J. Dual, G. Szekely, and M. Bajka. Inverse finite element characterization of soft tissues. *Medical Image Analysis*, 6(3):257–287, 2002.

Peter Kaufmann, Sebastian Martin, Mario Botsch, and Markus Gross. Flexible simulation of deformable models using discontinuous galerkin fem. *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 105–115, 2008.

Ladislav Kavan, Steven Collins, Jiri Zara, and Carol O'Sullivan. Geometric skinning with approximate dual quaternion blending. *ACM Trans. Graph.*, 27(4):105:1–105:23, October 2008.

Lily Kharevych, Patrick Mullen, Houman Owhadi, and Mathieu Desbrun. Numerical coarsening of inhomogeneous elastic materials. *ACM Trans. Graph.*, 28(3):51:1–51:8, July 2009.

Rafal Kicinger, Tomasz Arciszewski, and Kenneth De Jong. Evolutionary computation and structural design: A survey of the state-of-the-art. *Comput. Struct.*, 83(23-24):1943–1978, 2005.

Martin Kilian, Simon Flöry, Zhonggui Chen, Niloy J. Mitra, Alla Sheffer, and Helmut Pottmann. Curved folding. *ACM Trans. Graph.*, 27(3): 75:1–75:9, August 2008.

Tae-Yong Kim and Eugene Vendrovsky. Drivenshape - a data-driven approach to shape deformation. In *Proc. of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 2008.

R. M. Koch, M. H. Gross, F. R. Carls, D. F. von Büren, G. Fankhauser, and Y. Parish. Simulating facial surgery using finite element methods. In *Proc. of ACM SIGGRAPH*, pages 421–428, August 1996.

Paul G. Kry and Dinesh K. Pai. Interaction capture and synthesis. *ACM Trans. Graph. (Proc. of ACM SIGGRAPH)*, 25(3):872–880, 2006.

Paul G. Kry, Doug L. James, and Dinesh K. Pai. EigenSkin: Real Time Large Deformation Character Skinning in Hardware. In *ACM SIGGRAPH SCA*, pages 153–160, July 2002.

Yanxiang Lan, Yue Dong, Fabio Pellacini, and Xin Tong. Bi-scale appearance fabrication. *ACM Trans. Graph.*, 32(4), 2013.

A. H. Land and A. G Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960.

J. Lang, D. K. Pai, and R. J. Woodham. Acquisition of elastic models for interactive simulation. *International Journal of Robotics Research*, 21(8):713–733, 2002.

Manfred Lau, Akira Ohgawara, Jun Mitani, and Takeo Igarashi. Converting 3d furniture models to fabricatable parts and connectors. *ACM Trans. Graph.*, 30(4):85:1–85:6, July 2011.

Sung-Hee Lee and Demetri Terzopoulos. Heads up!: Biomechanical modeling and neuromuscular control of the neck. *ACM Trans. Graph. (Proc. of ACM SIGGRAPH)*, 25(3), 2006.

Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *The Quarterly of Applied Mathematics*, (2):164–168, 1944.

Anat Levin, Daniel Glasner, Ying Xiong, Fredo Durand, Bill Freeman, Wojciech Matusik, and Todd Zickler. Fabricating brdfs at high spatial resolution using wave optics. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 32(4):to appear, 2013.

J. P. Lewis, Matt Cordner, and Nickson Fong. Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *Proc. of ACM SIGGRAPH*, pages 165–172, 2000.

Xian-Ying Li, Chao-Hui Shen, Shi-Sheng Huang, Tao Ju, and Shi-Min Hu. Popup: automatic paper architectures from 3d models. *ACM Trans. Graph.*, 29: 111:1–111:9, July 2010.

Xian-Ying Li, Tao Ju, Yan Gu, and Shi-Min Hu. A geometric study of v-style pop-ups: theories and algorithms. *ACM Trans. Graph.*, 30(4):98:1–98:10, July 2011.

H. Lipson. Homemade: The future of functional rapid prototyping. In *IEEE Spectrum*, pages 24–31, May 2005.

Kui-Yip Lo, Chi-Wing Fu, and Hongwei Li. 3d polyomino puzzle. *ACM Trans. Graph.*, 28(5):157:1–157:8, December 2009.

Erik Lund and J Stegmann. On structural optimization of composite shell structures using a discrete constitutive parameterization. *Wind Energy*, 8: 109–124, 2005.

Linjie Luo, Ilya Baran, Szymon Rusinkiewicz, and Wojciech Matusik. Chopper: partitioning models into 3d-printable parts. *ACM Trans. Graph.*, 31 (6):129:1–129:9, November 2012.

Wan-Chun Ma, Andrew Jones, Jen-Yuan Chiang, Tim Hawkins, Sune Frederiksen, Pieter Peers, Marko Vukovic, Ming Ouhyoung, and Paul Debevec. Facial performance synthesis using deformation-driven polynomial displacement maps. *ACM Trans. Graph. (Proc. of ACM SIGGRAPH Asia)*, 27(5), 2008.

Nadia Magnenat-Thalmann, Prem Kalra, Jean Luc Lévêque, Roland Bazin, Dominique Batisse, and Bernard Queleux. A computational skin model: fold and wrinkle formation. *IEEE Trans. on Information Technology in Biomedicine*, 6(4):317–323, 2002.

Tom Malzbender, Ramin Samadani, Steven Scher, Adam Crume, Douglas Dunn, and James Davis. Printing reflectance functions. *ACM Trans. Graph.*, 31(3):20:1–20:11, June 2012.

W. Matusik, H. Pfister, M. Brand, and L. McMillan. A data-driven reflectance model. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH)*, 22(3):759–770, 2003.

Wojciech Matusik, Boris Ajdin, Jinwei Gu, Jason Lawrence, Hendrik P. A. Lensch, Fabio Pellacini, and Szymon Rusinkiewicz. Printing spatially-varying reflectance. *ACM Trans. Graph.*, 28(5):128:1–128:9, December 2009.

James McCrae, Karan Singh, and Niloy J. Mitra. Slices: a shape-proxy based on planar sections. *ACM Trans. Graph.*, 30(6):168:1–168:12, December 2011.

Ravish Mehra, Qingnan Zhou, Jeremy Long, Alla Sheffer, Amy Gooch, and Niloy J. Mitra. Abstraction of man-made shapes. *ACM Trans. Graph.*, 2009.

Ross J Micheals and Terrance E Boult. On the robustness of absolute orientation. In *Proceeding of the International Association for Science and Technology Development (IASTED) Conference on Robotics and Automation*, 2000.

Balint Miklos, Joachim Giesen, and Mark Pauly. Discrete scale axis representations for 3D geometry. *ACM Trans. Graph.*, 29:101:1–101:10, July 2010.

K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

Jun Mitani and Hiromasa Suzuki. Making papercraft toys from meshes using strip-based approximate unfolding. *ACM Trans. Graph.*, 23:259–263, August 2004.

Niloy J. Mitra and Mark Pauly. Shadow art. *ACM Trans. Graph.*, 28(5): 156:1–156:7, December 2009.

Alex Mohr and Michael Gleicher. Building efficient, accurate character skins from examples. *ACM Trans. Graph.*, 22(3):562–568, July 2003.

Yuki Mori and Takeo Igarashi. Plushie: an interactive design system for plush toys. *ACM Trans. Graph.*, 26(3), July 2007.

Matthias Müller and Markus Gross. Interactive virtual materials. In *GI '04: Proceedings of Graphics Interface 2004*, pages 239–246, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2004. Canadian Human-Computer Communications Society.

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, December 2006.

Diego Nehab, Szymon Rusinkiewicz, James Davis, and Ravi Ramamoorthi. Efficiently combining positions and normals for precise 3D geometry. *ACM Trans. Graph.*, 24:536–543, July 2005.

Matthieu Nesme, Paul G. Kry, Lenka Jeřábková, and François Faure. Preserving topology and elasticity for embedded deformable models. *ACM Trans. Graph.*, 28(3):52:1–52:9, July 2009.

Arnold Neumaier and Andrzej Pownuk. Linear systems with large uncertainties with applications to truss structures. *Reliable Computing*, 13:149–172, 2007.

R. W. Ogden. *Non-Linear Elastic Deformations*. Courier Dover Publications, 1997.

Hidehiko Okabe, Haruki Imaoka, Takako Tomiha, and Haruo Niwaya. Three dimensional apparel cad system. *ACM Trans. Graph.*, 26:105–110, July 1992.

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 ACM SIGGRAPH*, pages 87–96, August 2001.

Daniele Panozzo, Philippe Block, and Olga Sorkine-Hornung. Designing unreinforced masonry models. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 32(4):to appear, 2013.

Marios Papas, Wojciech Jarosz, Wenzel Jakob, Szymon Rusinkiewicz, Wojciech Matusik, and Tim Weyrich. Goal-based caustics. *Computer Graphics Forum (Proceedings of Eurographics 2011)*, 30(2):503–511, June 2011.

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 ACM SIGGRAPH)*, 32(4):to appear, 2013.

Sang Il Park and Jessica K. Hodgins. Capturing and animating skin deformation in human motion. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH)*, 25(3), 2006.

Sang Il Park and Jessica K. Hodgins. Data-driven modeling of skin and muscle deformation. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH)*, 27 (3), 2008.

Romain Prévost, Emily Whiting, Sylvain Lefebvre, and Olga Sorkine-Hornung. Make It Stand: Balancing shapes for 3D fabrication. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 32(4):to appear, 2013.

R Rebonato and P Jäckel. The most general methodology to create a valid correlation matrix for risk management and option pricing purposes. Technical report, Quantitative Research Centre, NatWest Group, 1999.

Christian Regg, Szymon Rusinkiewicz, Wojciech Matusik, and Markus Gross. Computational highlight holography. *ACM Trans. Graph.*, 29(6): 170:1–170:12, December 2010.

Alec Rivers, Andrew Adams, and Frédo Durand. Sculpting by numbers. *ACM Trans. Graph.*, 31(6):157:1–157:7, November 2012a.

Alec Rivers, Ilan E. Moyer, and Frédo Durand. Position-correcting tools for 2d digital fabrication. *ACM Trans. Graph.*, 31(4):88:1–88:7, July 2012b.

J.R. Rossignac and P. Borrel. Multi-resolution 3d approximations for rendering complex scenes. *Geometric Modeling in Computer Graphics, Springer-Verlag*, 1993.

Jonathan Richard Schewchuk. Lecture notes on geometric robustness. Lecture Notes, Nov 2009.

Michel Schmitt. Some examples of algorithms analysis in computational geometry by means of mathematical morphological techniques. In J.-D. Boissonnat and J.-P. Laumond, editors, *Geometry and Robotics*, volume 391 of *Lecture Notes in Computer Science*, pages 225–246. Springer Berlin Heidelberg, 1989.

D. S. Schnur and N. Zabaras. An inverse method for determining elastic material properties and a material interface. *International Journal for Numerical Methods in Engineering*, 33(10):2039–2057, 1992.

J. L. Schoner, J. Lang, and H.-P. Seidel. Measurement-based interactive simulation of viscoelastic solids. *Computer Graphics Forum (Proc. Eurographics)*, 23 (3):547–556, 2004.

Yuliy Schwartzburg and Mark Pauly. Fabrication-aware design with intersecting planar pieces. *Computer Graphics Forum (Proceedings of Eurographics 2013)*, 32(2), 2013. to appear.

Jonathan Richard Shewchuk. Adaptive Precision Floating-Point Arithmetic and Fast Robust Geometric Predicates. *Discrete & Computational Geometry*, 18 (3):305–363, October 1997.

Kaleem Siddiqi and Stephen Pizer. *Medial Representations: Mathematics, Algorithms and Applications*. Springer Publishing Company, Incorporated, 1st edition, 2008.

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

Mélina Skouras, Bernhard Thomaszewski, Bernd Bickel, and Markus Gross. Computational design of rubber balloons. *Comput. Graphics Forum (Proc. Eurographics)*, 2012.

Mélina Skouras, Bernhard Thomaszewski, Stelian Coros, Bernd Bickel, and Markus Gross. Computational design of mechanical characters. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 32(4): to appear, 2013.

Peter-Pike J. Sloan, Charles F. Rose, III, and Michael F. Cohen. Shape by example. In *I3D '01: Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 135–143, New York, NY, USA, 2001. ACM.

Jeffrey Smith, Jessica Hodgins, Irving Oppenheim, and Andrew Witkin. Creating models of truss structures with optimization. *ACM Trans. Graph.*, 21: 295–301, July 2002.

A. Smits, C. Ramault, A. Makris, D. Hemelrijck, A. Clarke, C. Williamson, M. Gower, R. Shaw, R. Mera, E. Lamkanfi, and W. Paepegem. A review of biaxial test methods for composites. In E.E. Gdoutos, editor, *Experimental Analysis of Nano and Engineering Materials and Structures*, pages 933–934. Springer Netherlands, 2007.

Peng Song, Chi-Wing Fu, and Daniel Cohen-Or. Recursive interlocking puzzles. *ACM Trans. Graph.*, 31(6):128:1–128:10, November 2012.

Ondrej Stava, Juraj Vanek, Bedrich Benes, Nathan Carr, and Radomír Měch. Stress relief: improving structural strength of 3d printable objects. *ACM Trans. Graph.*, 31(4):48:1–48:11, July 2012.

Stratasys. Objet500 connex, 2013. [Online; accessed 3-July-2013].

Shinjiro Sueda, Andrew Kaufman, and Dinesh K. Pai. Musculotendon simulation for hand animation. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 27(3), 2008.

Robert W. Sumner, Matthias Zwicker, Craig Gotsman, and Jovan Popović. Mesh-based inverse kinematics. In *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH)*, volume 24, pages 488–495, New York, NY, USA, 2005.

Tomáš Svoboda, Daniel Martinec, and Tomáš Pajdla. A convenient multicamera self-calibration for virtual environments. *PRESENCE: Teleoperators and Virtual Environments*, 14(4):407–422, August 2005.

Robert Endre Tarjan. A note on finding the bridges of a graph. *Inf. Process. Lett.*, 2(6):160–161, 1974.

J. Teran, E. Sifakis, S. Blemker, V. Ng Thow Hing, C. Lau, and R. Fedkiw. Creating and simulating skeletal muscle from the visible human data set. *IEEE TVCG*, 11:317–328, 2005.

D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *Proc. of ACM SIGGRAPH 87*, pages 205–214, 1987.

D. Terzopoulus and K. Waters. Analysis and synthesis of facial image sequences using physical and anatomical models. *IEEE Trans. PAMI*, 14:569–579, June 1993.

Nobuyuki Umetani, Jun Mitani, Takeo Igarashi, and Kenshi Takayama. Designing custommade metallophone with concurrent eigenanalysis. In *New Interfaces for Musical Expression++ (NIME++)*, pages 26–30, 2010.

Kiril Vidimče, Szu-Po Wang, Jonathan Ragan-Kelley, and Wojciech Matusik. Openfab: A programmable pipeline for multi-material fabrication. *ACM Transactions on Graphics*, 32, July 2013.

Georgy Voronoy. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Premier mémoire. Sur quelques propriétés des formes quadratiques positives parfaites. *Journal für die reine und angewandte Mathematik*, 133:97–102, 1908.

Michael D. Vose. A linear algorithm for generating random numbers with a given distribution. *Software Engineering, IEEE Transactions on*, 17(9):972–975, 1991.

F. C. A. Wai. Frictional joint for toys. U.S. Patent 7566256, 2006.

Robert Y. Wang, Kari Pulli, and Jovan Popović. Real-time enveloping with rotational regression. *ACM Trans. Graph.*, 26(3):73:1–73:9, July 2007.

Tim Weyrich, Jia Deng, Connelly Barnes, Szymon Rusinkiewicz, and Adam Finkelstein. Digital bas-relief from 3d scenes. *ACM Trans. Graph.*, 26(3), July 2007.

Tim Weyrich, Pieter Peers, Wojciech Matusik, and Szymon Rusinkiewicz. Fabricating microgeometry for custom surface reflectance. *ACM Trans. Graph.*, 28(3):32:1–32:6, July 2009.

Emily Whiting, John Ochsendorf, and Frédo Durand. Procedural modeling of structurally-sound masonry buildings. *ACM Trans. Graph.*, 28: 112:1–112:9, December 2009.

Emily Whiting, Hijung Shin, Robert Wang, John Ochsendorf, and Frédo Durand. Structural optimization of 3d masonry buildings. *ACM Trans. Graph.*, 31(6):159:1–159:11, November 2012.

Shiqing Xin, Chi-Fu Lai, Chi-Wing Fu, Tien-Tsin Wong, Ying He, and Daniel Cohen-Or. Making burr puzzles from 3d models. *ACM Trans. Graph.*, 30(4):97:1–97:8, July 2011.

Hao Zhang. Discrete combinatorial laplacian operators for digital geometry processing. In *Proc. of SIAM Conference on Geometric Design and Computing*, pages 575–592. Nashboro Press, 2004.

Qingnan Zhou, Julian Panetta, and Denis Zorin. Worst-case structural analysis. *ACM Trans. Graph.*, 32(4):137:1–137:12, July 2013.

Lifeng Zhu, Weiwei Xu, John Snyder, Yang Liu, Guoping Wang, and Baining Guo. Motion-guided mechanical toy modeling. *ACM Trans. Graph.*, 31(6):127:1–127:10, November 2012.

V. Zordan, B. Celly, B. Chiu, and P. C. Dilorenzo. Breathe easy: Model and control of human respiration for computer animation. In *Proc. of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 29–38, 2004.