

# Surface Deformation Potentials on Meshes for Computer Graphics and Visualization

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät  
der Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Dipl.-Inf. Patrick Degener

aus

Köln

Bonn, Januar 2010

---

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn

Dekan: Prof. Dr. U.-G. Meißner

1. Referent: Prof. Dr. Reinhard Klein
2. Referent: Prof. Dr. Leif Kobbelt
3. Referent: Prof. Dr. Mario Botsch

Tag der Promotion: 18. Juli 2011

Erscheinungsjahr 2011

Diese Dissertation ist auf dem Hochschulschriftenserver der ULB Bonn

[http://hss.ulb.uni-bonn.de/diss\\_online](http://hss.ulb.uni-bonn.de/diss_online)

elektronisch publiziert.



---

*Für Larissa.*

---

# CONTENTS

---

<b>Abstract</b>	<b>xi</b>
<b>Zusammenfassung</b>	<b>xiii</b>
<b>Acknowledgements</b>	<b>xv</b>
<b>I Introduction</b>	<b>1</b>
<b>1 Introduction</b>	<b>3</b>
1.1 Generalized Surface Deformations in Computer Graphics . . . . .	6
1.2 Main Contributions and Structure . . . . .	8
<b>2 Basics on Surfaces and Meshes</b>	<b>11</b>
2.1 Surfaces . . . . .	11
2.1.1 Metric Tensors . . . . .	12
2.1.2 The Shape Operator . . . . .	16
2.1.3 Functions on Surfaces . . . . .	17
2.2 Triangle Meshes . . . . .	18
2.2.1 Functions on Triangle Meshes . . . . .	20
2.2.2 Discrete Parameterizations . . . . .	21
<b>3 Basics on Deformations and Elasticity</b>	<b>23</b>
3.1 Solid Body Deformations . . . . .	23
3.1.1 Stress, Strain and Hooke's law . . . . .	24
3.1.2 Isotropic Materials . . . . .	26
3.2 Deformations of Surfaces . . . . .	27
3.2.1 Elastic Shells . . . . .	28
3.2.2 Membrane Potential and Parameterization . . . . .	31

<b>II</b>	<b>Deformation Potentials for Surface Parameterization</b>	<b>35</b>
<b>4</b>	<b>Potentials for Surface Parameterizations</b>	<b>39</b>
4.1	Introduction . . . . .	39
4.1.1	Parameterization of Triangle Meshes . . . . .	40
4.2	Previous and Related Work on Parameterization . . . . .	41
4.2.1	Important Properties of Parameterization Potentials . . . . .	43
4.2.2	Potentials based on Metric Deformation . . . . .	45
<b>5</b>	<b>Geometric Deformations Potentials</b>	<b>57</b>
5.1	Area . . . . .	58
5.2	Angle . . . . .	60
5.3	Length . . . . .	61
5.4	An adaptable Potential . . . . .	62
5.5	Properties . . . . .	65
5.5.1	Invariance with Respect to Rigid Transformation . . . . .	65
5.5.2	Infinity on the Domain Boundary . . . . .	66
5.5.3	Smoothness and Derivatives . . . . .	66
<b>6</b>	<b>Computing Parameterizations</b>	<b>69</b>
6.1	Optimization . . . . .	69
6.1.1	Simple Relaxation Optimization . . . . .	70
6.1.2	Hierarchical Optimization . . . . .	71
6.2	Experimental Evaluation . . . . .	72
6.3	Discussion . . . . .	78
<b>7</b>	<b>Material Specific Texture Maps and Patterns for Fabric Covered Surfaces</b>	<b>81</b>
7.1	Material Specific Texture Maps . . . . .	83
7.1.1	Discretization and Numerical Optimization . . . . .	88
7.2	Application: Pattern Generation for Upholstery . . . . .	90
7.2.1	Relations to Other Approaches To Pattern Generation . . . . .	90
7.2.2	Seams . . . . .	91
7.3	Results . . . . .	97
7.4	Discussion . . . . .	102
<b>8</b>	<b>Computing Parameterizations for Inconsistent Meshes and Point Clouds</b>	<b>105</b>
8.1	Related Work . . . . .	107
8.2	General Setup . . . . .	109
8.3	Generating the Proxy Surface . . . . .	110
8.4	Mapping between Proxy and Surface . . . . .	111

## CONTENTS

---

8.4.1	Electric Field of Points and Triangles . . . . .	113
8.4.2	Projection for Point Sets . . . . .	116
8.4.3	Projection for Triangulations . . . . .	116
8.5	Evaluation and Results . . . . .	118
8.5.1	Conclusion . . . . .	122
<b>III Interactive Deformation Potentials</b>		<b>123</b>
<b>9</b>	<b>Introduction</b>	<b>125</b>
<b>10</b>	<b>Previous Work on Deformable Models for Shape Editing</b>	<b>127</b>
10.1	General Approaches to Shape Editing . . . . .	127
10.2	Shape Editing with Deformation Potentials . . . . .	129
10.2.1	Differential Representations . . . . .	130
10.2.2	Non-Linear Approaches . . . . .	133
<b>11</b>	<b>A Non-Linear Potential for Interactive Editing</b>	<b>135</b>
11.1	Quaternion-based Coordinates . . . . .	137
11.1.1	From Fundamental Forms to Local Frames . . . . .	137
11.1.2	Encoding Local Frames . . . . .	138
11.1.3	Encoding Local Geometry . . . . .	139
11.2	Reconstruction Potentials . . . . .	140
11.2.1	Reconstructing from Quaternion-based Coordinates . . . . .	140
11.2.2	Deformation Potentials for Quaternion-based Coordinates	141
11.2.3	Derivatives . . . . .	144
11.2.4	Properties . . . . .	147
11.3	Efficient Non-linear Optimization . . . . .	150
11.3.1	Linear Steps . . . . .	151
11.3.2	Non-linear Step . . . . .	152
11.3.3	Parameterizing Deformation . . . . .	153
11.3.4	Quaternion Distance Measures . . . . .	157
11.3.5	Implementation . . . . .	161
<b>12</b>	<b>Results and Discussion</b>	<b>163</b>
12.1	Experimental Results . . . . .	163
12.2	Discussion . . . . .	168
12.3	More recent work . . . . .	170

<b>13 Extensions and Applications</b>	<b>171</b>
13.1 Application: Modeling Upholstery for Early Product Visualization	171
13.1.1 Basic Reconstruction and Editing	173
13.2 Force Field Constraints	176
13.3 Fitting Template Surfaces to Point Measurements	177
13.4 Geometric Continuity	180
<b>IV Visibility Driven Deformation for Panorama Maps</b>	<b>183</b>
<b>14 Introduction</b>	<b>185</b>
14.1 Panorama Maps	187
14.2 Contributions	189
14.3 Previous Work	190
<b>15 Potentials for Panorama Maps</b>	<b>193</b>
15.1 Variational Approach	193
15.1.1 Why Surface Deformation ? — Techniques in Manual Design	193
15.1.2 Problem Statement	194
15.1.3 Variational Formulation	195
15.2 Deformation Potential	197
15.2.1 Membrane Strain Potential	199
15.2.2 Bending Potential	201
15.3 Visibility Potentials	201
15.3.1 Screen Size Potential	202
15.3.2 Occlusion Potential	204
<b>16 Generating Panoramic Maps Automatically</b>	<b>211</b>
16.1 Combining Shape and Visibility Potentials	211
16.2 Discretization and Optimization	212
16.3 Approximation Scheme for the Occlusion Potential	213
16.4 Details on Discretizations and Derivatives	215
16.4.1 Discretization of $E_{\text{tangential}}$	215
16.4.2 Derivatives of $E_{\text{bending}}$	216
16.4.3 Discretization of $E_{\text{screen}}$	217
16.4.4 Discretization of $E_{\text{occ}}$	217
16.5 Results and Discussion	218
16.5.1 Experimental Results	218
16.5.2 Discussion	224
16.5.3 Conclusion	228

## CONTENTS

---

<b>V Conclusion and Closure</b>	<b>233</b>
<b>List of Publication</b>	<b>241</b>
<b>Index</b>	<b>244</b>
<b>Bibliography</b>	<b>247</b>





## ABSTRACT

---

Shape deformation models have been used in computer graphics primarily to describe the dynamics of physical deformations like cloth draping, collisions of elastic bodies, fracture, or animation of hair. Less frequent is their application to problems not directly related to a physical process. In this thesis we apply deformations to three problems in computer graphics that do not correspond to physical deformations. To this end, we generalize the physical model by modifying the energy potential. Originally, the energy potential amounts to the physical work needed to deform a body from its rest state into a given configuration and relates material strain to internal restoring forces that act to restore the original shape. For each of the three problems considered, this potential is adapted to reflect an application specific notion of shape. Under the influence of further constraints, our generalized deformation results in shapes that balance preservation of certain shape properties and application specific objectives similar to physical equilibrium states.

The applications discussed in this thesis are surface parameterization, interactive shape editing and automatic design of panorama maps. For surface parameterization, we interpret parameterizations over a planar domain as deformations from a flat initial configuration onto a given surface. In this setting, we review existing parameterization methods by analyzing properties of their potential functions and derive potentials accounting for distortion of geometric properties.

Interactive shape editing allows an untrained user to modify complex surfaces, be simply grabbing and moving parts of interest. A deformation model interactively extrapolates the transformation from those parts to the rest of the surface. This thesis proposes a differential shape representation for triangle meshes leading to a potential that can be optimized interactively with a simple, tailored algorithm. Although the potential is not physically accurate, it results in intuitive deformation behavior and can be parameterized to account for different material properties.

Panorama maps are blends between landscape illustrations and geographic maps that are traditionally painted by an artist to convey geographic survey knowledge on public places like ski resorts or national parks. While panorama maps are not drawn to scale, the shown landscape remains recognizable and the observer can easily recover details necessary for self location and orientation. At the same

time, important features as trails or ski slopes appear not occluded and well visible. This thesis proposes the first automatic panorama generation method. Its basis is again a surface deformation, that establishes the necessary compromise between shape preservation and feature visibility.

## ZUSAMMENFASSUNG

---

Deformationsmodelle werden in der Computergrafik bislang hauptsächlich eingesetzt, um die Dynamik physikalischer Deformationsprozesse zu modellieren. Gängige Beispiele sind Bekleidungssimulationen, Kollisionen elastischer Körper oder Animation von Haaren und Frisuren. Deutlich seltener ist ihre Anwendung auf Probleme, die nicht direkt physikalischen Prozessen entsprechen. In der vorliegenden Arbeit werden Deformationsmodelle auf drei Probleme der Computergrafik angewandt, die nicht unmittelbar einem physikalischen Deformationsprozess entsprechen. Zu diesem Zweck wird das physikalische Modell durch eine passende Änderung der potentiellen Energie verallgemeinert. Die potentielle Energie entspricht normalerweise der physikalischen Arbeit, die aufgewendet werden muss, um einen Körper aus dem Ruhezustand in eine bestimmte Konfiguration zu verformen. Darüber hinaus setzt sie die aktuelle Verformung in Beziehung zu internen Spannungskräften, die wirken um die ursprüngliche Form wiederherzustellen.

In dieser Arbeit passen wir für jedes der drei betrachteten Problemfelder die potentielle Energie jeweils so an, dass sie eine anwendungsspezifische Definition von Form widerspiegelt. Unter dem Einfluss weiterer Randbedingungen führt die so verallgemeinerte Deformation zu einer Fläche, die eine Balance zwischen der Erhaltung gewisser Formeigenschaften und Zielvorgaben der Anwendung findet. Diese Balance entspricht dem Equilibrium einer physikalischen Deformation.

Die drei in dieser Arbeit diskutierten Anwendungen sind Oberflächenparameterisierung, interaktives Bearbeiten von Flächen und das vollautomatische Erzeugen von Panoramakarten im Stile von Heinrich Berann. Zur Oberflächenparameterisierung interpretieren wir Parameterisierungen über einem flachen Parametergebiet als Deformationen, die ein ursprünglich ebenes Flächenstück in eine gegebene Oberfläche verformen. Innerhalb dieses Szenarios vergleichen wir dann existierende Methoden zur planaren Parameterisierung, indem wir die resultierenden potentiellen Energien analysieren, und leiten weitere Potentiale her, die die Störung geometrischer Eigenschaften wie Fläche und Winkel erfassen.

Verfahren zur interaktiven Flächenbearbeitung ermöglichen schnelle und intuitive Änderungen an einer komplexen Oberfläche. Dazu wählt der Benutzer Teile der Fläche und bewegt diese durch den Raum. Ein Deformationsmodell extra-

poliert interaktiv die Transformation der gewählten Teile auf die restliche Fläche. Diese Arbeit stellt eine neue differentielle Flächenrepräsentation für diskrete Flächen vor, die zu einem einfach und interaktiv zu optimierendem Potential führt. Obwohl das vorgeschlagene Potential nicht physikalisch korrekt ist, sind die resultierenden Deformationen intuitiv. Mittels eines Parameters lassen sich außerdem bestimmte Materialeigenschaften einstellen.

Panoramakarten im Stile von Heinrich Berann sind eine Verschmelzung von Landschaftsillustration und geographischer Karte. Traditionell werden sie so von Hand gezeichnet, dass bestimmte Merkmale wie beispielsweise Skipisten oder Wanderwege in einem Gebiet unverdeckt und gut sichtbar bleiben, was große Kunstfertigkeit verlangt. Obwohl diese Art der Darstellung nicht maßstabsgetreu ist, sind Abweichungen auf den ersten Blick meistens nicht zu erkennen. Dadurch kann der Betrachter markante Details schnell wiederfinden und sich so innerhalb des Gebietes orientieren. Diese Arbeit stellt das erste, vollautomatische Verfahren zur Erzeugung von Panoramakarten vor. Grundlage ist wiederum eine verallgemeinerte Oberflächendeformation, die sowohl auf Formhaltung als auch auf die Sichtbarkeit vorgegebener geographischer Merkmale abzielt.

## ACKNOWLEDGEMENTS

---

The following thesis would not have been possible without the excellent collaborative environment provided by the computer graphics working group at the university of Bonn that i had the pleasure to be part of. At the heart of this environment is certainly my advisor Prof. Dr. Reinhard Klein who i would like to thank for the inspiration that was the seed of this work and for his continuous motivation that guided my through all stages of this thesis. He also was the one that pulled me into research and he always encouraged me to follow my own scientific curiosity.

But the seed would not have sprouted without the many discussions and collaborations with my colleagues in the working group. My thanks are due to my coauthors Ruwen Schnabel, Christopher Schwartz, Roland Wahl, Sebastian Möser, Markus Schlattmann and Gerhard Bendels. Working with them was not only highly productive but also a pleasure. Thanks also go to my room mates Alexander Gress, Roland Wahl, Raoul Wessel and Fabian Aitenau for many fruitful discussions. In addition to those, i also found very good friends in Gero Müller, Marcin and Dominik Novotni, Jan Meseth and Tomas Lay Herrera. Thanks to all of you and certainly also to the rest of the group for giving me a great time.

Certainly, I would like to express my gratitude to Prof. Dr. Leif Kobbelt and Prof. Dr. Mario Botsch for serving as external reviewers. In addition, I want to thank Prof. Dr. Mario Botsch for the test cases from [SB09] and the comparison with Primo. Many thanks also go to the Volkswagen AG and RSS GmbH for providing data sets for experimental evaluation.<sup>1</sup>

As representatives for many friends in Bonn that contributed by cheering me up, listening or simply spending time, i would like to thank Timm Springer, Dirk Peters, Jens Schröder and Sven Esmark.

Finally, I would like to thank my family: My mother and father for their love, support, and all the freedom they gave and still give me; My brothers Lukas, Fabian and Michel simply for being around; My daughter Anna for enriching my life in ways i never expected; And my wife Larissa who I am very glad to know at my side. To her i dedicate this thesis.

---

<sup>1</sup>The results shown in Chapters 7 and 13 are based on sewing patterns, upholstery frame data and 3D scans kindly provided by the Volkswagen AG. RSS GmbH kindly provided aerial images and elevation data for the Sölden ski resort and the Zugspitze area show in Part IV.



**Part I**  
**Introduction**





# CHAPTER 1

---

## INTRODUCTION

---

Since the beginnings of mankind, humans have deformed objects found in their environment to put them into a shape more suitable for a particular function. Early examples can be found in bows or wicker made from tree branches, bamboo or willow switches. But the principle of “constructive deformation” that increases fitness of a shape for a certain purpose has been applied throughout the ages e.g. to form tools and weapons from iron, to shape ship planks, or in upholstery. Long since it has found its way into industrial production processes of all kinds of metals and plastics, car bodies, cans or carton boxes.

Physically speaking, to trigger a deformation external forces have to be applied to a body. At the same time, internal material forces act to preserve the original shape configuration. In the interplay of these forces, the actual deformation is found in an equilibrium configuration that minimizes an energy potential. While this happens almost instantaneously in the real world, the physical laws have to be carefully modeled and simulated in the virtual world. This is tedious and poses high computational demands that easily overstrain the powers even of supercomputers. But it also has a significant advantage: In the virtual setting we are no more limited by the actual laws of physics. In fact, we can vary forces and potential to find shapes with properties desired by certain applications. In the simulation, the constructive deformation approach becomes even more powerful.

This thesis follows the basic idea of such generalized deformations and applies it to three problems in computer graphics. Although inspired by physical deformation, each of these generalized deformations is tuned to a specific application in computer graphics. In all cases we concentrate on the final result of the deformation process, i.e. the static equilibrium configuration and neglect the dynamics. It is thus the potential that determines the properties of this final shape and that lies at the center of our interest. For each problem, an energy potential is chosen that aims at restoring the original shape similar to the internal restoring forces in a deformed solid body. From application to application, the actual choice of the potential, however, varies reflecting an application specific notions of shape and further optimality criteria.

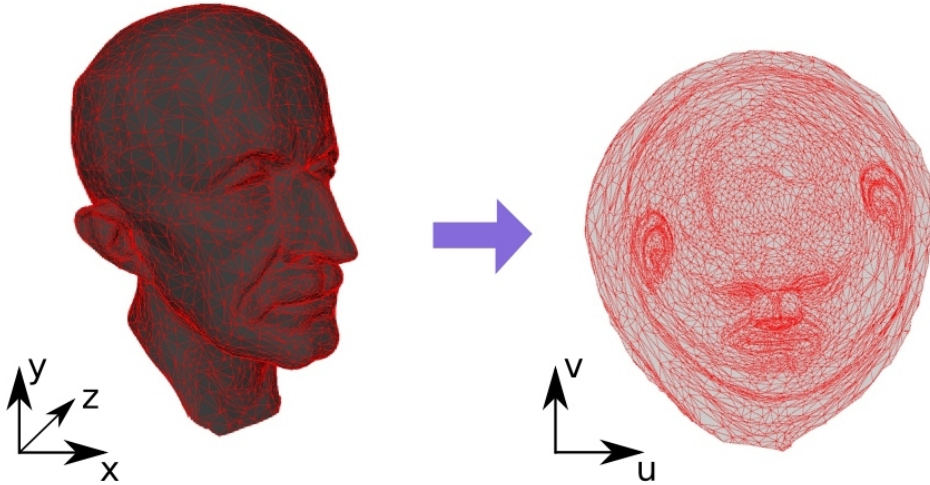


Figure 1.1: A surface parameterization example: A surface shown on the left is mapped by a parameterization to a planar uv-domain.

In contrast to physical deformation of solids, in computer graphics shapes are rather represented by their surface only, as the surface largely dominates their appearance. Consequently, we focus on deformation of surfaces. Natural examples of surface deformations can be found in thin-walled structures as leaves, paper sheets, egg shells or tin cans. In physics, their deformation behavior is mathematically described by *thin shell models* that already have been applied to computer graphics in the context of animation of cloth and other thin walled structures. The potentials proposed in this thesis borrow from thin shell models but abandon the strict physical accuracy in favor of application specific requirements. Nevertheless, the potentials remain similar to the thin shell model's potential in that they preserve some notion of shape.

The first problem approached in Part II is *surface parameterization*, which refers to the task of unfolding a curved surface in three space into a planar parameter space (see Figure 1.1). Mapping from surfaces to the familiar plane, parameterization help to cope with the inherent complexity of surfaces and are one of the most essential tools in computer graphics, used for texture mapping, remeshing, filtering, compression and much more. But even long before their use in computer graphics, parameterizations have found applications in mathematics or geography. The most familiar incarnations of parameterizations are geographic maps, that map a sphere onto the plane. In this thesis, we interpret parameterizations as surface deformations that deform a planar patch into a curved surface. Just as with geographic maps, parameterizations are best if they faithfully reproduce



Figure 1.2: Original shape and result of an interactive editing session. The result was obtained by fixing parts of the dragons back feet (shown in green) and pulling parts of its nose (shown in orange) upward. The surface deformation interpolates the transformation of the nose over the surface but preserves shape details as much as possible.

important geometric relations like distances or areas. For surface parameterization, the potential governing the deformation should thus aim at restoring these geometric properties during flattening. In Part II we interpret existing parameterization methods in context of surface deformation, analyze their energy potentials and, based on our observations, derive a set of novel potentials.

The second application approached in Part III is *interactive shape editing* that allows a user to interactively edit surfaces represented as large triangle meshes as they are typically obtained from acquisition devices like laser range scanners. The user can grab and move parts of the surface while remaining unconstrained surface parts deform interactively according to a deformation model that preserves shape detail as much as possible. An example result of an interactive editing session is shown in Figure 1.2. Such editing approaches are highly intuitive giving the user the impression of pushing or pulling a thin object made of an elastic material like rubber. For interactive shape editing, potentials must meet two requirements: First, they must support an efficient optimization to enable interactive frame rates on large models and, second, they must be well-defined for triangle meshes as this is the predominant representation for scanned objects. In Part III we propose a

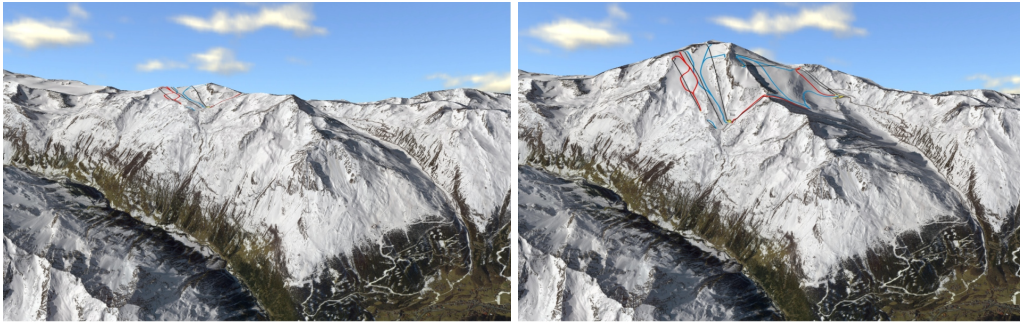


Figure 1.3: Automatic generation of panorama maps: On the left a traditional perspective image of a ski resort is shown. Slopes and lifts are highlighted but hardly visible due to occlusion and size. A panorama map of the same area as shown on the right avoids the occlusion of these features and also scales them to improve visibility.

potential for interactive shape editing which is physically plausible and can imitates the behavior of different materials. Yet, it can efficiently be optimized by a simple optimization algorithm.

In the third application tackled in this thesis, the deformation not only aims at restoring the original shape, but also at enhancing the visibility of important features. We call these kind of deformation *visibility driven* and put it to work in the design of panorama maps. *Panorama maps* are blends between landscape painting and geographic map that show an area of interest (e.g. a ski resort) from a familiar, earth bound perspective. These maps are traditionally drawn by an experienced and skillful artist, who ensures that important features (like ski slopes) are visible and not occluded in the final image. In Part IV, we show how such maps can be generated automatically by surface deformation with a suitable chosen potential. For this application, the potential must again regulate shape deviation to ensure that the panorama resembles the actual landscape. But at the same time, it also has to enforce the visibility of important features.

## 1.1 Generalized Surface Deformations in Computer Graphics

Besides applications in the three above mentioned areas, we would like to point to some other problems in computer graphics that have been approached with similar ideas. We exclude from this discussion physical simulations for animations, that do not generalize the physical deformations but aim at most accurate reproduction

of the physical process. For a survey of those methods we refer to [NMK<sup>+</sup>06].

*Surface matching* refers to the task of finding a map between two similar surfaces, that maps corresponding features onto each other. A popular approach to surface matching consists in searching a deformation that aligns both surfaces. Besides space deformations, there are a number of methods [KS04, SAPH04, LDRS05] that employ surface deformations in this context. These methods try to establish suitable maps by minimizing potentials that measure both feature alignment and intrinsic shape deformation. While feature alignment is specific to shape matching, potentials for intrinsic shape deformation have long been searched after for surface parameterization. While in this thesis, the surface matching problem is not explicitly discussed, the above approaches reuse potentials originally developed for surface parameterizations that are discussed in Part II. Surface deformations have also been used to find features correspondences (a preprocessing step of surface matching) as e.g. recently in [ZSC<sup>+</sup>08].

One of the hardest problems in geometry processing is *surface reconstruction* from point measurements. Here, the surface of a shape must be reconstructed from a set of partial, noisy and possibly incomplete point clouds. Each of these clouds originate e.g. from a laser range scan taken from a different position. Surface reconstruction is one of the most important problems in computer graphics and many different approaches exist. Closely related to surface deformations are methods based on deformable models like snakes or balloning methods (see [MM98, MT00, Dua01, SLS<sup>+</sup>06] and references herein). Methods in this class evolve an initial approximating surface to fit the measured points. At the same time, the deformation of the evolving surface is restricted by a deformable model to prevent overfitting and to give reasonable completions in regions where measurements are missing.

While classical surface reconstruction assumes a fixed, unarticulated shape, the problem becomes even harder if the scanned object is allowed to move or change over time. Surface reconstruction then consists in reassembling partial scans taken at different positions and points in time into a consistent, articulated shape. Without further assumptions this problem is ill-posed, as the shape potentially can change arbitrarily between each pair of measurements. 4D surface reconstruction methods (see [LRS<sup>+</sup>09] for a survey) thus need a shape regularization that limits the allowed deformation between two time frames. This regularization is usually realized by a solid or surface deformation model. A recent example is the approach of Wand et al. [WAO<sup>+</sup>09] who use a physically motivated potential to limit the deformation while fitting a gradually constructed urshape to point measurements. Fitting the urshape to point measurements does not necessarily correspond to a physical deformation process and can thus be understood as a generalized surface deformation.

Symmetry detection on large surfaces has recently gained much interest in



computer graphics [PSG<sup>+</sup>06, LE06, MGP07, PMW<sup>+</sup>08]. It allows to identify identical or similar parts within a given surface, such as window sills on the facade of a large building. The identification of such symmetries and repetitions allows very efficient compression, shape analysis and easy editing. While existing methods are limited to detection of features coupled by rigid or simple transformations, in recent work of Berner et al. [BBW<sup>+</sup>09] use a space deformation model based on the well-known thin plate potential to match features on a deformed surface. Even though a space and not a surface deformation is used, their approach can be regarded as an application of generalized deformations.

We also like to point to relations of the generalized deformation concept with shape morphing and, more general, *shape spaces* [DM97] as they have become an increasingly active field of research over the last years. In shape spaces shapes are arranged such that their geodesic distance with respect to an appropriate shape metric reflects similarity or dissimilarity e.g. as perceived by humans. The structure of shape spaces then allows to interpolate shapes, to compute mean shapes or even to perform statistical analysis of shape sets. Given a shape space and a metric, minimizing shape distance has a similar effect than minimizing a generalized deformation potential in the sense that it tends to restore a reference shape and, vice versa, distance in shape space can be defined via a deformation potential. In contrast to shape spaces, we focus in this thesis on more application specific requirements on the potentials as e.g. support for efficient minimization while the implied distance metric is of minor concern.

In addition to computer graphics problems, the above sketched idea of generalized deformations is also related to *shape optimization* [HM03], an engineering field concerned with shapes that are optimal for a certain purpose. Typical examples are mechanical parts with minimal weight that sustain given loads or airplane wings that minimize air flow turbulence. In shape optimization, the deformation is driven by a general objective function that, in contrast to physical deformation and the potentials developed here, does not aim at restoring an original configuration. Traditionally, it targets on design problems in mechanics while this thesis focuses on computer graphics applications.

## 1.2 Main Contributions and Structure

This thesis follows the main idea of generalized surface deformation and applies it to the three problems sketched in the introduction. It summarizes and extends work that has been already published at several international computer graphics conferences [DMK03, DK07, PDK07, SDK09, DK09] (a further publication [DTK] is currently in preparation) and which is also listed in Section V. The Parts II-IV each correspond to one of these applications while the remainder of this

first part gives the necessary background on surfaces, deformations and elasticity. Within the three application areas we make the following main contributions:

- Part **II** starts with an analysis of planar surface parameterization methods and the derivation of a novel method accounting for both angle and area distortion. These contributions have been published previously as parts of the Diploma thesis [Deg03], but have been updated and reinterpreted in context of surface deformations for this thesis. The parameterization method has moreover been complemented by an projection approach, that allows to compute texture maps for surfaces with triangulation artifacts and topological inconsistencies as they frequently occur in day to day modeling practice and for surfaces given as point sets. For surfaces covered with fabrics as e.g. upholstery, we suggest *material specific texture maps* that are obtained by optimizing a potential from elasticity theory. In contrast to standard texture maps, these maps do not aim at minimal shape deformation, but rather on a plausible and realistic deformation specific to a given fabric material. As an application, we show how parameterization methods can be used to infer sewing patterns for upholstery in industrial design.
- In Part **III**, the thesis contributes a novel differential representation for discrete surfaces targeted at interactive shape editing. Residuals in this representation give rise to a potential that can be very efficiently minimized. We describe an efficient optimization algorithm that allows editing of large models at interactive frame rates. We show how the method can be parameterized to reflect material properties and, finally, we extend it to rapidly visualize the shape of upholstered furniture in industrial design.
- In context of panorama maps, Part **IV** derives an appropriate potential measuring visibility and appearance compliance. Based on this potential, we propose the first, fully automatic panorama generation method. The user must only specify an rough initial viewpoint and a set of features. The panorama map is then computed automatically by a surface deformation.





---

## BASICS ON SURFACES AND MESHES

---

### 2.1 Surfaces

The predominant representation of surfaces in computer graphics is the polygonal mesh (that will be defined in Section 2.2) and all algorithms developed in this thesis were designed to work on this representation. Nevertheless, it will sometimes be convenient for the discussion to consider the case of a smooth surfaces. In this section we therefore summarize the most important concepts from differential geometry. A more detailed introduction to differential geometry can be found in [Bär01].

We define a two dimensional *regular surface* as a set  $S \subset \mathbb{R}^3$  that satisfies the following property: For each  $\mathbf{p} \in S$  there exists an open neighborhood  $V \subset \mathbb{R}^3$  and a map  $\mathbf{x} : \omega \rightarrow V \cap S$  from an open subset  $\omega \subset \mathbb{R}^2$  with the following properties:

- $\mathbf{x}$  is differentiable.
- $\mathbf{x}$  is a homeomorphism. As it is continuous this means that it has an continuous inverse  $\mathbf{x}^{-1}$ . More precisely, there exists a continuous function  $\mathbf{X} : V \rightarrow \mathbb{R}^2$  such that  $\mathbf{x}^{-1} = \mathbf{X}|_{V \cap S}$ .
- the Jacobian  $\nabla \mathbf{x}$  is of full rank at each point  $\mathbf{u} \in \omega$ .

The function  $\mathbf{x}$  is called a *parameterization* of the neighborhood  $V \cap S$ .

We denote coordinates of  $\omega$  by  $\theta^1, \theta^2$ . Here and in the following, we precede a lower index  $\alpha$  by a colon to indicate partial derivatives with respect to  $\theta^\alpha$ , e.g. we write  $\mathbf{x}_{,\alpha}$  to denote the partial derivative  $\partial \mathbf{x} / \partial \theta^\alpha$ . Moreover, we use Einstein summation conventions which means that we implicitly sum over all possible values if an index appears repeatedly in a term. For example, the dot product of two points  $\mathbf{p}, \mathbf{q} \in \mathbb{R}^3$  is written as

$$\mathbf{p} \cdot \mathbf{q} = p_i q_i$$

To indicate the range of possible values, sub- and superscript Greek indices will always take values 1, 2 while latin indices take values 1, 2, 3. ( An exception are indices  $v$  and  $w$  into a set of vertex indices  $V$  of a triangle mesh to be defined in the next section. These will take values in  $V$ ).

Taking derivatives of the parameterization  $\mathbf{x}$  in coordinate directions  $\theta^\alpha$  at each point  $\mathbf{x}(\theta^1, \theta^2)$  of  $S$  yields two vectors

$$\mathbf{a}_\alpha := \mathbf{x}_{,\alpha}$$

for  $\alpha = 1, 2$ , that are tangent to  $S$ . As the Jacobian  $\nabla \mathbf{x}$  is of full rank, these two tangent vectors are linearly independent and span a plane  $T_{\mathbf{p}}S$  at each point  $\mathbf{p} = \mathbf{x}(\theta^1, \theta^2)$ . This plane runs tangential to the surface  $S$  at  $\mathbf{p}$  is thus called *tangent plane*.

We can extend the system  $(\mathbf{a}_1, \mathbf{a}_2)$  to a basis frame by adding a third vector

$$\mathbf{a}_3 := \mathbf{n} := \frac{\mathbf{a}_1 \times \mathbf{a}_2}{\|\mathbf{a}_1 \times \mathbf{a}_2\|}$$

which is normal and orthogonal to  $T_{\mathbf{p}}S$  and thus to the surface  $S$  in  $\mathbf{p}$ . Every vector that is orthogonal to  $T_{\mathbf{p}}S$  is called a *surface normal* at  $\mathbf{p}$ .

While the above definition of regular surfaces requires only existence of parameterizations for local neighborhoods  $V \cap S$ , we will often assume that a single parameterization  $\mathbf{x} : \omega \rightarrow S$  covers the entire surface  $S$ . The existence of such a parameterization implies the existence of a smooth normal field  $\mathbf{a}_3$  on  $S$ . Surfaces for which a smooth normal field can be found are called *orientable*.  $S$  must thus be orientable to support a single parameterization. Besides orientability, the spaces  $S$  and  $\omega$  must have a compatible topology and in particular the genus, the number of boundaries and connected components must match. As  $\omega$  is planar, the surface  $S$  must have genus zero to support a single parameterization, which informally means that there are no tunnels in the surface.

### 2.1.1 Metric Tensors

Using a parameterization  $\mathbf{x}$ , shapes in  $\omega$  can be mapped onto  $S$  and vice versa. In general, the shapes undergo a deformation in this mapping. This inherent deformation is captured in the (*covariant*) *metric tensor* (which is also called *first fundamental form*) defined as

$$g_{\alpha\beta} := \mathbf{a}_\alpha \cdot \mathbf{a}_\beta = x_{i,\alpha} x_{i,\beta}$$

To see this, we consider two smooth curves

$$\begin{aligned} \mathbf{c} &: [0, 1] \rightarrow \omega, t \mapsto \mathbf{c}(t) \\ \mathbf{d} &: [0, 1] \rightarrow \omega, t \mapsto \mathbf{d}(t) \end{aligned}$$

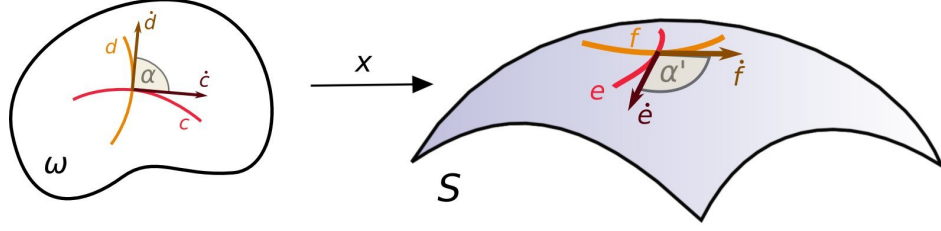


Figure 2.1: Deformation of shape angles by the parameterization. We consider the intersection angle  $\alpha$  between two curves  $c$  and  $d$  and relate it to the intersection angle of the images  $e = x \circ c$  and  $f = x \circ d$  of the two curves on the surface.

in  $\omega$ . Mapping these curves using  $x$  we obtain two curves  $e = x \circ c$  and  $f = x \circ d$  that take values on the surface  $S$  (see Figure 2.1). If  $c$  and  $d$  intersect in a point  $\mathbf{u} := c(t_0) = d(t_1)$  then  $e$  and  $f$  intersect in  $x(\mathbf{u})$ . At each point  $c(t)$  of a curve, a tangent vector is given by the derivative with respect to  $t$  which we denote by a dot, i.e.  $\dot{c} = \partial c / \partial t$ . The intersection angle  $\alpha$  between  $c$  and  $d$  is easily computed from the dot product  $\dot{c}_\gamma \dot{d}_\gamma$  of the tangents  $\dot{c}$  and  $\dot{d}$  to these curves as

$$\alpha = \arccos\left(\frac{\dot{c}_\gamma \dot{d}_\gamma}{\|\dot{c}\| \|\dot{d}\|}\right)$$

After mapping both curves to the surface, the intersection angle  $\alpha'$  of the images  $e$  and  $f$  at  $x(\mathbf{u})$  is computed analogously as

$$\alpha' = \arccos\left(\frac{\dot{e}_k \dot{f}_k}{\|\dot{e}\| \|\dot{f}\|}\right)$$

Now, the dot product and norms in this expression evaluate to

$$\dot{e} \cdot \dot{f} = \dot{e}_k \dot{f}_k = x_{k,\alpha} \dot{c}_\alpha x_{k,\beta} \dot{d}_\beta = g_{\alpha\beta} \dot{c}_\alpha \dot{d}_\beta = \dot{c}^t g \dot{d} \quad (2.1)$$

$$\|\dot{e}\|^2 = \dot{e}_k \dot{e}_k = x_{k,\alpha} \dot{c}_\alpha x_{k,\beta} \dot{c}_\beta = g_{\alpha\beta} \dot{c}_\alpha \dot{c}_\beta = \dot{c}^t g \dot{c} \quad (2.2)$$

The intersection angle  $\alpha'$  of the images can thus be computed from the original planar curves and the metric tensor only. The metric tensor thus describes angular shape deformation imposed by the parameterization  $x$ .

But also changes in length are described by the metric tensor: In general, the length of a smooth curve  $e$  is given as

$$l(e) = \int_0^1 \|\dot{e}\| dt$$

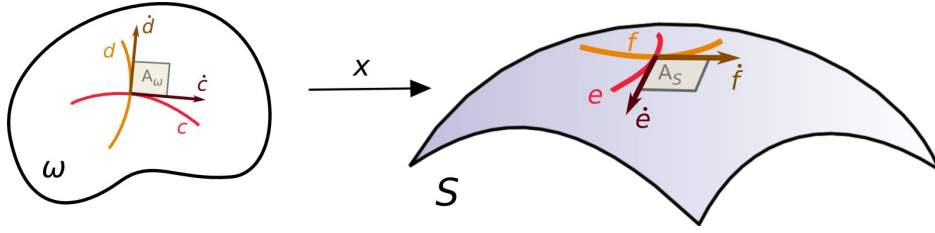


Figure 2.2: Deformation of shape area by the parameterization. This time we consider the area of a small area element spanned  $\dot{\mathbf{c}}$  and  $\dot{\mathbf{d}}$  and relate it to the surface area element spanned by images  $\dot{\mathbf{e}}$  and  $\dot{\mathbf{f}}$ .

Applying Equation 2.2 we get

$$l(\mathbf{e}) = \int_0^1 (\dot{e}_k \dot{e}_k)^{\frac{1}{2}} dt = \int_0^1 (g_{\alpha\beta} \dot{c}_\alpha \dot{c}_\beta)^{\frac{1}{2}} dt$$

and again the length of the image curve  $\mathbf{e}$  can be computed from the original planar curve and the metric tensor. Finally, we would like find the change in area that the parameterization imposes on a planar shape. To this extent, we consider the parallelogram spanned by the scaled tangent vectors  $\epsilon \dot{\mathbf{c}}$  and  $\epsilon \dot{\mathbf{d}}$  at  $\mathbf{u}$  (see Figure 2.2). Its area is given by

$$A_\omega = \epsilon^2 \det \mathbf{B}$$

where  $\mathbf{B} = (\dot{\mathbf{c}} \ \dot{\mathbf{d}})$  denotes the matrix whose columns equal the tangents of the two curves. The corresponding area of the parallelogram spanned by  $\epsilon \dot{\mathbf{e}}$  and  $\epsilon \dot{\mathbf{f}}$  can be computed using the cross product as

$$A_S = \|\epsilon \dot{\mathbf{e}} \times \epsilon \dot{\mathbf{f}}\| = \epsilon^2 (\|\dot{\mathbf{e}}\|^2 \|\dot{\mathbf{f}}\|^2 - (\dot{\mathbf{e}} \cdot \dot{\mathbf{f}})^2)^{\frac{1}{2}}$$

and by Equations 2.1-2.2

$$A_S = \epsilon^2 (\dot{\mathbf{c}}^t \mathbf{g} \dot{\mathbf{c}} \dot{\mathbf{d}}^t \mathbf{g} \dot{\mathbf{d}} - (\dot{\mathbf{c}}^t \mathbf{g} \dot{\mathbf{d}})^2)^{\frac{1}{2}} = \epsilon^2 \det(\mathbf{B}^t \mathbf{g} \mathbf{B})^{\frac{1}{2}}$$

The change in area is thus given as

$$A_S/A_\omega = (\det \mathbf{g})^{\frac{1}{2}}$$

This relation is independent of the scaling factor  $\epsilon$  and thus also holds in the limit  $\epsilon \rightarrow 0$  for an infinitesimal area element  $d\omega$  at point  $\mathbf{u}$  and its image  $dS$  on the surface. The local change in area imposed by the parameterization can thus be again described in terms of the metric tensor as

$$dS = (\det \mathbf{g})^{\frac{1}{2}} d\omega \quad (2.3)$$

For later reference we also define the *contravariant metric tensor* as the inverse of covariant tensor  $\mathbf{g}$ . Its components are denoted by  $g^{\alpha\beta}$  and are uniquely defined by

$$g_{\alpha\beta}g^{\beta\gamma} = \delta_{\alpha}^{\gamma}$$

where  $\delta_{\alpha\beta} = \delta_{\alpha}^{\beta} = \delta^{\alpha\beta}$  is the Kronecker delta that evaluates to one for  $\alpha = \beta$  and is zero otherwise.

To conclude this section, we look at two important types of parameterizations. In many cases a parameterization is desired that perfectly preserves shapes. Such a parameterization is called *isometric* and from Equations 2.1-2.2 we easily see that it must satisfy

$$g_{\alpha\beta} = \delta_{\alpha\beta}$$

in every point  $\mathbf{u} \in \omega$ . If an isometric parameterization exists for a surface  $S$ , it cannot be distinguished from a planar domain by measuring angles, distances, areas or any property that can be expressed in terms of these. Such properties that are invariant under isometric parameterizations (or more generally isometric maps between surfaces) are called *intrinsic*. Intrinsic properties depend on the surface metric only and not on the position or configuration of the surface in the surrounding space. Intuitively, a property is intrinsic if it can be measured by small, shortsighted inhabitants of the surface. Geodesics length, angles and surface area are certainly intrinsic properties while mean curvature or minimal curvature direction (defined in the next section) are not.

Interestingly, the ‘Theorema egregium’ by Gauss states that Gaussian curvature (defined in the next section) is also an intrinsic property and thus preserved by isometric parameterizations. As the parameter domain  $\omega$  is flat, Gaussian curvature vanishes everywhere. Isometric parameterizations exist therefore only if the Gaussian curvature in each point of  $S$  also equals zero. This is only the case for a few special surfaces, as e.g. cylinder or plane.

While isometric parameterization do not exist for general surfaces, the Uniformization Theorem [Wei] guarantees (under the topological assumptions of the last section) the existence of a conformal parameterization  $\mathbf{x} : \omega \rightarrow S$  that covers the whole surface  $S$ . A parameterization is said to be *conformal* if in every point  $\mathbf{u} \in \omega$  the metric tensor equals the scaled identity, i.e.

$$g_{\alpha\beta}(\mathbf{u}) = \lambda(\mathbf{u})\delta_{\alpha\beta}$$

with a smooth real valued function  $\lambda(\mathbf{u})$  that is called *conformal factor*. Conformal parameterizations have the important property that they perfectly preserve angles. This follows from Equations 2.1-2.2 that simplify for a conformal map to

$$\dot{e}_k \dot{f}_k = \lambda \delta_{\alpha\beta} \dot{c}_{\alpha} \dot{d}_{\beta} = \lambda \dot{c}_{\gamma} \dot{d}_{\gamma} \quad \text{and} \quad \|\dot{\mathbf{e}}\|^2 = \lambda \delta_{\alpha\beta} \dot{c}_{\alpha} \dot{c}_{\beta} = \lambda \|\dot{\mathbf{c}}\|^2$$

and we have  $\alpha = \alpha'$  by the above definitions of the intersection angles.

### 2.1.2 The Shape Operator

The metric tensor only gives access to intrinsic shape properties. To compute non-intrinsic properties like curvatures, it needs to be complemented by the *second fundamental form*. If  $\mathbf{n} : \omega \rightarrow S$  is the smooth unit surface normal field, it is given as

$$h_{\alpha\beta} = -\mathbf{n} \cdot \mathbf{a}_{\alpha,\beta} = -\mathbf{n} \cdot \mathbf{x}_{,\alpha,\beta} = -n_k x_{k,\alpha,\beta}$$

As the normal is orthogonal to  $\mathbf{x}_{,\alpha}$  we have

$$0 = \frac{\partial}{\partial \theta^\beta} (n_k x_{k,\alpha}) = n_{k,\beta} x_{k,\alpha} + n_k x_{k,\alpha,\beta}$$

and thus

$$h_{\alpha\beta} = \mathbf{n}_{,\beta} \cdot \mathbf{x}_{,\alpha} = \mathbf{n}_{,\beta} \cdot \mathbf{a}_\alpha \quad (2.4)$$

Written in this form, it becomes apparent that second fundamental form captures changes in the normal field projected to the tangent plane  $T_p S$ . In fact, it is sufficient to consider only this projection: Because of  $\|\mathbf{n}\| = 1$  we have

$$0 = \frac{\partial}{\partial \theta^\beta} (n_k n_k) = 2n_k n_{k,\beta} = 2\mathbf{n}_{,\beta} \cdot \mathbf{a}_3$$

i.e. the normal component of  $\mathbf{n}_{,\beta}$  vanishes. For two arbitrary directions  $\mathbf{s}, \mathbf{t} \in \mathbb{R}^2$  at a point  $\mathbf{u} \in \omega$  the expression  $s^\alpha h_{\alpha\beta} t^\beta$  evaluates to the normal change in direction  $\mathbf{a}_\alpha s^\alpha$  projected onto  $\mathbf{a}_\beta t^\beta$ . While changes in the surface normal intuitively correspond to surface curvature, the second fundamental form is “contaminated” by the shape deformation caused by the parameterization. To subtract the influence of the parameterization, we divide by the metric tensor which yields the *shape operator* as

$$w_\alpha^\beta := h_{\alpha\gamma} g^{\gamma\beta}$$

The difference between shape operator and second fundamental form becomes obvious when one considers a spherical surface  $S$ . One verifies, that on a sphere the shape operator equals identity in each point which correspond to our intuition of constant curvature in all directions, while the second fundamental form evaluates to the metric tensor in this case and thus varies across the surface.

For a smooth parameterization the second order derivatives  $\mathbf{x}_{,\alpha,\beta}$  and  $\mathbf{x}_{,\beta,\alpha}$  are equal and so the second fundamental form as well as the shape operator are symmetric. The shape operator can therefore be diagonalized and its eigenvectors correspond to the *principal curvature directions*, i.e. the directions of minimal and maximal curvature on the surface while the associated eigenvalues give the curvatures in these directions. For a unit sphere, the shape operator equals identity and thus the curvature is constant over all directions.

Finally, the eigenvalues  $\kappa_1$  and  $\kappa_2$  associated with the principal curvature directions are called *principal curvatures*. They can be averaged in two different ways: The *mean curvature* is simply defined as the mean  $\kappa_{mean} = \frac{1}{2}(\kappa_1 + \kappa_2)$ . In contrast, the *Gaussian curvature* is defined as the product of the principle curvature  $\kappa = \kappa_1\kappa_2$ . While the shape operator itself, principle curvatures and mean curvature are non-intrinsic properties, the famous ‘Theorema egregium’ by Gauss states that Gaussian curvature is surprisingly an intrinsic property, i.e. it does not change under isometries and can be expressed in terms of the metric tensor alone.

### 2.1.3 Functions on Surfaces

At some point it will be necessary to consider functions defined on a regular surface  $S$ . Just like functions defined on the Euclidean space  $\mathbb{R}^n$ , we can take derivatives or integrate these functions. The concept of smoothness, differential and integral can be easily generalized using parameterizations. We consider a function  $\mathbf{F} : S \rightarrow \mathbb{R}^n$ . We say  $\mathbf{F}$  is *smooth* in a point  $\mathbf{p} \in S$ , if there exists a open neighborhood  $V \subset S$  around  $\mathbf{p}$  and parameterization  $\mathbf{x} : \omega \rightarrow V$  such that  $\mathbf{F} \circ \mathbf{x}$  is smooth. It can be shown, that this definition is independent of the choice of  $\mathbf{x}$ . We call  $\mathbf{F}$  *smooth*, if it is smooth in every point  $\mathbf{p} \in S$ .

Slightly more general is the case of a function  $\tilde{\mathbf{x}} : \bar{S} \rightarrow S$  that maps between two surfaces  $\bar{S}$  and  $S$  as shown in Figure 3.2. However, as  $S \subset \mathbb{R}^3$  it can be considered as functions taking values in  $\mathbb{R}^3$  and the above definition of smoothness applies as well. The *differential*  $D\tilde{\mathbf{x}}$  of such functions at a point  $\bar{\mathbf{p}} = \tilde{\mathbf{x}}(\theta^1, \theta^2)$  is defined as the linear map

$$D_{\bar{\mathbf{p}}}\tilde{\mathbf{x}} : T_{\bar{\mathbf{p}}}\bar{S} \rightarrow T_{\tilde{\mathbf{x}}(\bar{\mathbf{p}})}S$$

between the tangent spaces specified as follows: For a tangent vector  $\mathbf{v} \in T_{\bar{\mathbf{p}}}\bar{S}$  we consider an arbitrary curve  $\mathbf{c} : (-1, 1) \rightarrow \bar{S}$  with  $\mathbf{c}(0) = \bar{\mathbf{p}}$  and  $\dot{\mathbf{c}} = \mathbf{v}$  and map it to the curve  $\mathbf{e} := \tilde{\mathbf{x}} \circ \mathbf{c}$  on  $S$ . The differential is then set to the tangent at  $\tilde{\mathbf{x}} \circ \mathbf{c}$ , i.e.

$$D_{\bar{\mathbf{p}}}\tilde{\mathbf{x}}(\mathbf{v}) := \dot{\mathbf{e}}|_{t=0}$$

It can be shown, that the differential  $D_{\bar{\mathbf{p}}}\tilde{\mathbf{x}}$  is well defined in this way and that it is in fact a linear map.

We now turn back to the case of a smooth real valued function  $\mathbf{F} : S \rightarrow \mathbb{R}$  on the surface  $S$ . In this case the differential becomes a linear function

$$D_{\mathbf{p}}\mathbf{F} : T_{\mathbf{p}}S \rightarrow \mathbb{R}$$

as  $T_{\mathbf{F}(\mathbf{p})}\mathbb{R}$  and  $\mathbb{R}$  are isomorphic. According to the Riesz representation theorem there exists a uniquely determined tangent vector  $\nabla_S\mathbf{F} \in T_{\mathbf{p}}S$ , such that

$$\nabla_S\mathbf{F} \cdot \mathbf{v} = D_{\mathbf{p}}\mathbf{F}(\mathbf{v})$$

for all  $\mathbf{v} \in T_{\mathbf{p}}S$ . This tangent vector  $\nabla_S \mathbf{F}$  is called the *surface gradient* of  $\mathbf{F}$ . Just as the gradient of functions defined on Euclidean spaces, it points into the direction in which the function  $\mathbf{F}$  increases most.

To integrate a function  $\mathbf{F} : S \rightarrow \mathbb{R}$  over the surface  $S$ , we make again use of the parameterization  $\mathbf{x} : \omega \rightarrow S$  and set

$$\int_S F dS := \int_{\omega} F \circ \mathbf{x} (\det \mathbf{g})^{\frac{1}{2}} d\omega$$

where the factor  $(\det \mathbf{g})^{\frac{1}{2}}$  accounts for the change in area imposed by the parameterization  $\mathbf{x}$  as given by Equation 2.3. This definition can be shown to be invariant under changes of the parameterization. While for the sake of simplicity this definition uses a single parameterization of the entire surface  $S$ , the surface integral can be defined in a similar way using local parameterizations (see e.g. [dC76]).

We finally state a generalized integral transformation theorem for surfaces that directly follows from this definition. Again, we consider a map  $\tilde{\mathbf{x}}$  between surface  $\tilde{S}$  and  $S$  as well as a real valued function  $\mathbf{F} : S \rightarrow \mathbb{R}$  on  $S$ . Both surfaces are parameterized over  $\omega$  by  $\tilde{\mathbf{x}}$  and  $\mathbf{x}$  respectively such that  $\tilde{\mathbf{x}} = \mathbf{x} \circ \tilde{\mathbf{x}}^{-1}$  (see Figure 3.2) and we set  $\tilde{\mathbf{F}} = \mathbf{F} \circ \tilde{\mathbf{x}}$ . In this situation we have

$$\begin{aligned} \int_S F dS &= \int_{\omega} F \circ \mathbf{x} (\det \mathbf{g})^{\frac{1}{2}} d\omega \\ &= \int_{\omega} \tilde{\mathbf{F}} \circ \tilde{\mathbf{x}} (\det \mathbf{g})^{\frac{1}{2}} d\omega = \int_{\tilde{S}} \tilde{\mathbf{F}} (\det \mathbf{g} \tilde{\mathbf{g}}^{-1})^{\frac{1}{2}} d\tilde{S} \end{aligned}$$

i.e. the area elements are related by

$$dS = (\det \mathbf{g} \tilde{\mathbf{g}}^{-1})^{\frac{1}{2}} d\tilde{S} \quad (2.5)$$

## 2.2 Triangle Meshes

While large parts of the theory in this thesis are illustrated and valid in the smooth setting of a regular surface, for computer graphics applications it is primarily the discrete case of a piecewise affine surface  $S$  that is of importance. In computer graphics, piecewise affine surfaces occur almost naturally as the output of many different geometry acquisition devices and techniques. Moreover, most rendering algorithms also expect surfaces in this representation. As an alternative to smooth regular surfaces, we thus consider the case of a surface  $S$  given as a *triangle mesh*. To formalize this, we define an abstract mesh connectivity as a tuple  $\mathcal{M} = (V, \mathcal{T})$  where  $V$  is a set of vertex indices and  $\mathcal{T} \subset V^3$  is a set of triangular faces.



With each vertex  $v \in V$  we associate a vertex coordinate  $\mathbf{x}^v \in \mathbb{R}^3$ . Using these vertex coordinates, each abstract face  $T \in \mathcal{T}$  with  $T = (v_1, v_2, v_3)$  defines a triangle  $T_S \subset \mathbb{R}^3$  in the Euclidean space as

$$T_S := \{ \lambda_1 \mathbf{x}^{v_1} + \lambda_2 \mathbf{x}^{v_2} + \lambda_3 \mathbf{x}^{v_3} \mid \lambda_1 + \lambda_2 + \lambda_3 = 1, \lambda_i > 0 \}.$$

i.e.  $T_S$  is obtained by barycentric interpolation of the positions  $\mathbf{x}^v$  of the triangle's vertices. We will write  $|T_S|$  to denote the area of the triangle  $T_S$ . The fact that  $S$  is represented by the triangle mesh  $\mathcal{M}$  and vertex positions  $(\mathbf{x}^v)_{v \in V}$  can now be formalized as

$$S = \bigcup_{T \in \mathcal{T}} T_S.$$

An illustration is given in Figure 2.3.

Often a second surface  $\bar{S}$  is considered that is also given as a triangle mesh, with the same connectivity  $\mathcal{M}$  but different vertex coordinates. Its vertex coordinates are denoted by  $(\bar{\mathbf{x}}^v)_{v \in V}$  so that

$$\bar{S} = \bigcup_{T \in \mathcal{T}} T_{\bar{S}}.$$

where  $T_{\bar{S}}$  is defined analogous to  $T_S$ .

Some shorthand notation will be handy to describe the mesh connectivity. A vertex  $v \in V$  appearing in a tuple  $T \in \mathcal{T}$  is said to be *incident* with  $T$  and we write  $v \in T$ . If two vertices  $v, w \in V$  appear in consecutive order in some tuple  $T \in \mathcal{T}$  we write  $(v, w) \in \mathcal{M}$  to denote this fact. If either  $(v, w) \in \mathcal{M}$  or  $(w, v) \in \mathcal{M}$  the vertices are said to be *adjacent*. The adjacency pattern of vertices in the mesh is compactly represented in the *adjacency matrix* denoted by  $\mathcal{M}_{vw}$  that is defined as

$$\mathcal{M}_{vw} = \begin{cases} 1 & \text{if } (v, w) \in \mathcal{M} \\ 0 & \text{otherwise} \end{cases}$$

For any subset  $H \subset V$  of the vertices, a symbol  $H_v$  is defined as

$$H_v = \begin{cases} 1 & \text{if } v \in H \\ 0 & \text{otherwise} \end{cases}$$

Regular surfaces are a special case of so called two dimensional manifolds. In particular, their definition requires the existence of a homeomorphism around each point  $\mathbf{p} \in S$  that maps the surface to the plane. Intuitively this means, that the surface locally resembles a bended plane. In analogy, a mesh  $\mathcal{M}$  is said to be *manifold* if at each edge  $(v, w)$  at most two faces meet. In this case, the two triangles can be unfolded into the plane. This becomes impossible if more

than two faces are adjacent. Throughout this thesis we assume that meshes are manifold if not stated otherwise.

The notion of orientability of regular surfaces can also be generalized to the discrete case: A mesh is called *oriented*, if for any two triangles  $T^1, T^2 \in \mathcal{T}$  sharing a common edge  $\{v, w\}$ , the two vertices  $v$  and  $w$  appear in opposite order in the ordered 3-tuples  $T^1$  and  $T^2$ . It is thus called *orientable* if it can be oriented only by permutation of tuples  $T \in \mathcal{T}$ . This discrete notion of oriented meshes does in fact relate to smooth normal fields on regular surfaces: For each triangle  $T = (v_1, v_2, v_3)$  a normal for  $T_S$  can be defined by taking the cross product of two edges, i.e.

$$\mathbf{n}_T = (\mathbf{x}^{v_2} - \mathbf{x}^{v_1}) \times (\mathbf{x}^{v_3} - \mathbf{x}^{v_1}) / \|(\mathbf{x}^{v_2} - \mathbf{x}^{v_1}) \times (\mathbf{x}^{v_3} - \mathbf{x}^{v_1})\|$$

The resulting normal field is consistently oriented if and only if  $\mathcal{M}$  is oriented.

## 2.2.1 Functions on Triangle Meshes

Section 2.1.3 introduced differential and integral for real valued functions  $F$  defined on regular surfaces  $S$ . In the discrete case, where the surface is represented by a triangle mesh, it is common in computer graphics to require  $F|_{T_S}$  to be affine within each triangle  $T_S$ . This is advantageous as such functions allow for compact storage and efficient evaluation. In particular, if  $F$  is continuous in addition, it is uniquely determined by its values  $(F(\mathbf{x}^v))_{v \in V}$  on the vertices of  $\mathcal{M}$ .

We apply the same restriction to functions  $\tilde{\mathbf{x}} : \bar{S} \rightarrow S$  between surfaces. If  $\tilde{\mathbf{x}}$  continuous, it is then uniquely determined by its values  $\mathbf{x}^v := \tilde{\mathbf{x}}(\bar{\mathbf{x}}^v)$  on the vertices  $\bar{\mathbf{x}}^v \in \bar{S}$ . In this situation  $S = \tilde{\mathbf{x}}(\bar{S})$  is again represented by the same mesh but with vertex coordinates  $(\mathbf{x}^v)_{v \in V}$  and we have  $T_S = \tilde{\mathbf{x}}(T_{\bar{S}})$ . The discrete setup for a map between surfaces is illustrated in Figure 2.3. In contrast to regular surfaces, representing the surface  $S$  by a triangular mesh does not require the existence of a parameterization  $\mathbf{x} : \omega \rightarrow S$ . Nevertheless, when generalizing potentials defined on regular surfaces to triangle meshes, it will be handy to have a discrete counterpart for the parameterization. In the case of a triangular mesh, we also assume that the inverse parameterization  $\mathbf{x}^{-1} : S \rightarrow \omega$  is piecewise affine on the triangles  $T_S$  and set  $\mathbf{u}^v := \mathbf{x}^{-1}(\mathbf{x}^v)$ . Having a piecewise affine parameterization for a triangle mesh is in fact very helpful for many geometry related tasks in computer graphics and Part II will describe computational methods to find these.

As the image of  $S$  under the piecewise affine function  $\mathbf{x}^{-1}$ , the domain  $\omega = \mathbf{x}^{-1}(S)$  is again given as triangle mesh with connectivity  $\mathcal{M}$  and vertex coordi-

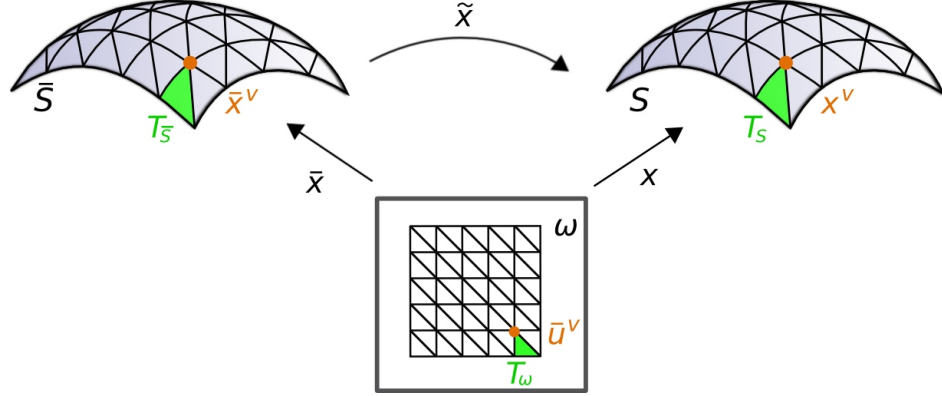


Figure 2.3: Notation for a map  $\tilde{\mathbf{x}}$  between two surfaces in the discrete case. Both surfaces are given as triangle meshes with the same connectivity  $\mathcal{M}$  but with different vertex coordinates. On  $S$  the vertex coordinates are denoted by  $(\mathbf{x}^v)_{v \in V}$  and on  $\bar{S}$  by  $(\bar{\mathbf{x}}^v)_{v \in V}$ .

nates  $(\mathbf{u}^v)_{v \in V}$  (see Figure 2.3). In particular

$$\omega = \bigcup_{T \in \mathcal{T}} T_\omega$$

where the triangle  $T_\omega$  is defined analogously to  $T_S$ .

### 2.2.2 Discrete Parameterizations

With the piecewise affine parameterization  $\mathbf{x}$  for a surface  $S$  given as triangle mesh, the discrete setup described so far parallels the smooth case of a regular surface. It is therefore possible to take over concepts discussed in context of regular surfaces like the metric tensor to the discrete case. In fact, as  $\mathbf{x}|_{T_\omega}$  is smooth in the interior of  $T_\omega$ , the gradient, metric tensors, and the area element as defined in Section 2.1 exist. Moreover, as these quantities are expressed in terms of the first order derivatives of  $\mathbf{x}$ , they become constant within each triangle.

To find expressions for these quantities, it is convenient to define a set of piecewise affine, continuous basis functions  $\Psi_v : \omega \rightarrow \mathbb{R}$  within each triangle  $T_\omega$ , uniquely determined by

$$\Psi_v(\mathbf{u}^w) = \delta_{vw}.$$

In fact, one easily verifies that the family  $(\Psi_v)_{v \in V}$  is a basis for the space of continuous piecewise affine functions on the triangulation  $\mathcal{M}$  of  $\omega$ . In this basis, the parameterization  $\mathbf{x}$  is given as

$$\mathbf{x} = \Psi_v \mathbf{x}^v \tag{2.6}$$

and analogous expressions hold for  $\bar{\mathbf{x}}$ . Partial derivatives  $\mathbf{x}_{,\alpha}$  are immediately found as

$$\mathbf{x}_{,\alpha} = \Psi_{v,\alpha} \mathbf{x}^v \quad (2.7)$$

and by definition the metric tensor  $\mathbf{g}$  is then given as

$$g_{\alpha\beta} = \Psi_{v,\alpha} \Psi_{w,\beta} x_k^v x_k^w.$$

It remains to find an expression for the partial derivatives  $\Psi_{v,\alpha}$  of the basis functions. To this extent we consider a single triangle  $T = (v^1, v^2, v^3)$ . The restriction of the basis function  $\Psi_{v^i}|_{T_\omega}$  is found as the affine map

$$\Psi_{v^i}|_{T_\omega}(\mathbf{u}) = \frac{(\mathbf{u}^{v^{i+2}} - \mathbf{u}^{v^{i+1}}) \mathbf{R}^{\pi/2} (\mathbf{u} - \mathbf{u}^{v^{i+1}})}{(\mathbf{u}^{v^{i+2}} - \mathbf{u}^{v^{i+1}}) \mathbf{R}^{\pi/2} (\mathbf{u}^{v^i} - \mathbf{u}^{v^{i+1}})} \quad (\text{i not summed})$$

where  $\mathbf{R}^{\pi/2}$  denotes rotation by  $\pi/2$  in the plane and superindices  $i$  are understood modulo 3. Its partial derivatives are given by

$$\Psi_{v^i,\alpha}|_{T_\omega} = \frac{(u_\beta^{v^{i+2}} - u_\beta^{v^{i+1}}) R_{\beta\alpha}^{\pi/2}}{A_{T_\omega}} \quad (\text{i not summed})$$

with

$$A_{T_\omega} := (u_\beta^{v^{i+2}} - u_\beta^{v^{i+1}}) R_{\beta\gamma}^{\pi/2} (u_\gamma^{v^i} - u_\gamma^{v^{i+1}}) \quad (\text{i not summed})$$

Within the triangle  $T_\omega$  all but three basis functions  $\Psi_{v^i}$  for  $i = 1, 2, 3$  vanish and so do their partial derivatives. The restriction of the metric tensor to  $T_\omega$  is thus given as

$$g_{\alpha\beta}|_{T_\omega} = \frac{(u_\gamma^{v^{i+2}} - u_\gamma^{v^{i+1}})(u_\delta^{v^{j+2}} - u_\delta^{v^{j+1}}) R_{\gamma\alpha}^{\pi/2} R_{\delta\beta}^{\pi/2} x_k^{v^i} x_k^{v^j}}{(A_{T_\omega})^2} \quad (\text{i,j summed})$$

Finally, we turn to integrals of functions on surfaces given as triangle meshes. Throughout this thesis we consider potentials on surfaces that take the form

$$E = \int_S e(\mathbf{g}) dS$$

where  $e : S \rightarrow \mathbb{R}$  is a real valued density function defined on the surface and that can be expressed in terms of  $\mathbf{g}$  or at least in terms of  $\Psi_{v^i,\alpha}$ . In both cases, the density  $e(\mathbf{g})|_{T_S}$  is constant on each triangle and we can write

$$E = \int_\omega e(\mathbf{g})(\det \mathbf{g})^{\frac{1}{2}} d\omega = \sum_{T \in \mathcal{T}} e(\mathbf{g}|_{T_\omega})(\det \mathbf{g}|_{T_\omega})^{\frac{1}{2}} |T_\omega| = \sum_{T \in \mathcal{T}} e(\mathbf{g}|_{T_\omega}) |T_S|$$

In the discrete case such potential therefore become a simple area weighted sum over all triangles.

---

## BASICS ON DEFORMATIONS AND ELASTICITY

---

While this thesis considers generalizations of surface deformations, the starting point and initial motivation are certainly physical deformation processes as they surround us in everyday life when we come in touch with soft objects as rubber bands, seats, pillows, cloth, clay or skin. As a motivation and to understand, how such deformations can be modeled mathematically, this chapter not only gives essential basics on mathematical deformations but also looks at how these are described in physics, where deformations have been studied since the early experiments of Robert Hooke in the 17th century. While physics describes materials of all kinds and flavors, we pick the rather simple linear elastic model for illustration. Primarily interested in static equilibrium configurations rather than dynamics, we focus on the elastic potential that characterizes such states. This chapter starts with the theory for elastic solids and then turns to deformation of surfaces.

### 3.1 Solid Body Deformations

We review here some fundamental concepts from the theory of elasticity and introduce the linearized elastic model. It is based on three fundamental assumptions: the continuum hypothesis, the assumptions that local displacements are small and the assumption of a linear stress/strain relation (Hooke's law). The discussion concentrates on aspects most relevant for this thesis and we refer the interested reader to [GZ02] for more details.

Consider an elastic solid body  $\bar{B} \subset \mathbb{R}^3$  which is deformed under the influence of external forces into a shape  $B \subset \mathbb{R}^3$ . We assume that both shapes are parameterized over a common domain  $\Omega \subset \mathbb{R}^3$  by smooth maps  $\bar{\mathbf{r}}$  and  $\mathbf{r}$  respectively in terms of a system  $\theta^1, \theta^2, \theta^3$  as illustrated in Figure 3.1. The deformation is then given as the map  $\tilde{\mathbf{r}} = \mathbf{r} \circ \bar{\mathbf{r}}^{-1}$ . At each point  $\bar{\mathbf{r}}(\theta^1, \theta^2, \theta^3)$  of  $\bar{B}$  taking derivatives in coordinate directions  $\theta^i$  yields a local frame given by

$$\bar{\mathbf{a}}_i = \bar{\mathbf{r}}_{,i} .$$

The analogous expression holds for the deformed body  $B$ . To ensure that the

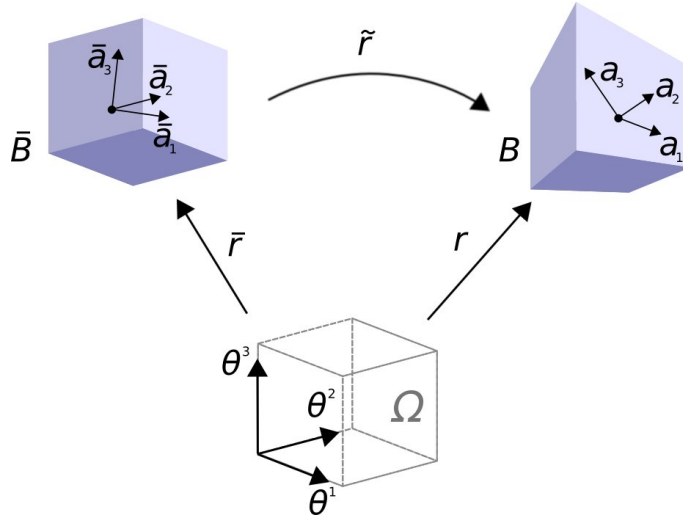


Figure 3.1: Geometry of a solid body in undeformed and deformed configurations.

system  $\bar{\mathbf{a}}_i$  is a well-defined coordinate frame, we require that the maps  $\bar{\mathbf{r}}$  and  $\mathbf{r}$  are *regular*, i.e. their Jacobians are of full rank. The just defined frame vectors  $\bar{\mathbf{a}}_i$  are also called *covariant base vectors*. For later, we define a dual coordinate frame of *contravariant base vectors*  $\bar{\mathbf{a}}^i$  as the uniquely defined set of vectors satisfying

$$\bar{\mathbf{a}}_i \cdot \bar{\mathbf{a}}^j = \delta_i^j$$

where  $\delta_i^j = \delta_{ij} = \delta^{ij}$  denotes the Kronecker's delta function.

Considering  $\bar{B}$  and  $B$  as three dimensional regular surfaces, we can now generalize the covariant metric tensor for both parameterizations as

$$\bar{G}_{ij} = \bar{\mathbf{a}}_i \cdot \bar{\mathbf{a}}_j \text{ and } G_{ij} = \mathbf{a}_i \cdot \mathbf{a}_j .$$

and likewise the contravariant metric tensor  $\bar{G}^{ij}$  and  $G^{ij}$  respectively as the matrix uniquely determined by

$$\bar{G}_{ik} \bar{G}^{kj} = G_{ik} G^{kj} = \delta_i^j$$

Similar to the metric tensor  $\mathbf{g}$  on surfaces defined in Section 2.1.1, the tensor  $\mathbf{G}$  can be used to find local shape deformation (in area, angle, length, and volume) caused by the parameterizations.

### 3.1.1 Stress, Strain and Hooke's law

Relative displacements between infinitesimal particles in the body are represented by *strain*, a local deformation measure that captures how much the deformation

$\tilde{\mathbf{r}}$  differs from a rigid body transformation. Mathematically, it is defined as the difference in squared length of a differential line element  $d\tilde{\mathbf{r}}$  inside the body  $\tilde{B}$  and the corresponding line segment  $d\mathbf{r}$  inside  $B$ . As the elongation of the line element  $d\tilde{\mathbf{r}}$  can vary with its direction, strains in all directions are compactly represented by a tensor — the *Green-Lagrange strain tensor* — which is given as

$$\gamma_{ij} = \frac{1}{2}(G_{ij} - \bar{G}_{ij}) . \quad (3.1)$$

Denoting the preimage of the differential line element  $d\mathbf{r}$  under  $\mathbf{r}$  in parameter space by  $d\theta$  we have

$$d\tilde{r}^i = \bar{r}_{i,j}d\theta^j \quad \text{and} \quad dr^i = r_{i,j}d\theta^j .$$

The strain in direction  $d\theta$  can now be conveniently expressed using the Green-Lagrange strain tensor:

$$2\gamma_{ij}d\theta^i d\theta^j = (G_{ij} - \bar{G}_{ij})d\theta^i d\theta^j \quad (3.2)$$

$$= G_{ij}d\theta^i d\theta^j - \bar{G}_{ij}d\theta^i d\theta^j \quad (3.3)$$

$$= r_{k,i}r_{k,j}d\theta^i d\theta^j - \bar{r}_{k,i}\bar{r}_{k,j}d\theta^i d\theta^j \quad (3.4)$$

$$= dr^k dr^k - d\tilde{r}^k d\tilde{r}^k \quad (3.5)$$

$$= \|d\mathbf{r}\|^2 - \|d\tilde{\mathbf{r}}\|^2 \quad (3.6)$$

If an elastic solid body is deformed, restoring forces act in each point of the body to restore its original undeformed configuration. A simple example is the one dimensional case of a spring. In this case the restoring force is given by Hooke's law

$$F = -k \cdot x \quad (3.7)$$

which relates the restoring force  $F$  and the current strain (i.e. elongation or compression) of the spring  $x$ , where  $k$  is the elastic constant of the spring. We see from the above expression, that the magnitude of the restoring force  $F$  is proportional to the strain  $x$ . Moreover, it points in the opposite direction of the strain thus seeking to restore the system to its equilibrium.

In order to compress or extend a spring by a certain length  $x$  it is necessary to apply an external force of equal magnitude as the restoring force  $F$  but opposite direction over the distance  $x$ . The expression for the corresponding work is obtained by integrating this force along the direction of  $x$  which yields

$$\mathbf{E} = \frac{1}{2}kx^2 .$$

This work is stored in the spring as the *elastic potential energy* and the restoring force  $F$  coincides with its negative gradient, i.e. at each time it aims to minimize the elastic potential energy.

In the three dimensional case, there are besides compression and elongation along an axis several other ways to deform a solid, like bending, shearing or twisting. Each of these ways contributes its own amount of elastic potential energy. In general, the elastic potential energy is thus a function of the three dimensional strain tensor  $\gamma_{ij}$ . For elastic materials, the elastic energy's density function is given by

$$e = \frac{1}{2} E^{ijkl} \gamma_{ij} \gamma_{lk}$$

where  $E^{ijkl}$  denotes the elasticity tensor that characterizes the material. The elastic energy itself is then obtained by integrating this density function over the undeformed body  $\bar{B}$ . Even though, as a tensor of order four  $E^{ijkl}$  has 81 components, it can be shown, that it must satisfy the following symmetry condition

$$E^{ijkl} = E^{klij} = E^{jikl} = E^{ijlk}$$

which leaves only 36 degrees of freedom. It turns out, that for energetic reasons the number of independent entries is further reduced to 21. In analogy to the one dimensional case, the restoring forces can be obtained from the potential elastic energy by taking the derivatives with respect to the strains  $\gamma_{ij}$  which yields

$$\tau^{ij} := \frac{\partial e}{\partial \gamma_{ij}} = E^{ijkl} \gamma_{lk} . \quad (3.8)$$

The above equation is the tensor expression of Hooke's law. The expression  $\tau^{ij}$  is called *stress* tensor from which the restoring forces in all directions can be derived. To see this, consider an arbitrary surface  $S$  inside the deformed body  $B$  that separates it into two disjunct parts  $B^-$  and  $B^+$ . On the surface we fix an infinitesimal surface element  $dS$  around some point  $\mathbf{O}$  with unit normal  $\mathbf{n}$ . We can think of the elastic body  $B^+$  exerting forces on the elastic body  $B^-$  across the surface  $S$ . The forces acting on  $dS$  are equivalent to a resulting force vector  $\mathbf{t}$  at  $\mathbf{O}$ . Denoting the components of  $\mathbf{n}$  with respect to the local frame  $\mathbf{a}^i$  by  $n_i$  and those of  $\mathbf{t}$  with respect to  $\mathbf{a}_i$  by  $t^i$  respectively, it can be shown, that this resulting force is then given by

$$t^i = \tau^{ij} n_j . \quad (3.9)$$

### 3.1.2 Isotropic Materials

Mathematically, the most simple materials are *isotropic elastic materials*. For such materials the elasticity tensor has no preferred direction, i.e. a force will give the same displacement (relative to its direction) independent of the direction in which the force is applied. Many materials, for example metallic alloys like



steel or aluminum, exhibit such behavior when strained within certain limits. For isotropic materials the elasticity tensor can be written in a particular simple form

$$E^{ijkl} = \frac{E}{1 + \nu} \left( \frac{\nu}{1 - 2\nu} \bar{G}^{ij} \bar{G}^{kl} + \frac{1}{2} (\bar{G}^{ik} \bar{G}^{jl} + \bar{G}^{il} \bar{G}^{jk}) \right)$$

with now only two independent material parameters  $E$  and  $\nu$ . The first parameter  $E$  is called *Young's modulus* and describes the material's unidirectional stiffness. It is closely related to the elastic spring constant  $k$  in Equation 3.7: In fact, a bar made from an isotropic elastic material can be considered as a spring whose elastic constant  $k$  is proportional to Young's modulus  $E$ . The second parameter  $\nu$  is called *Poisson's ratio*. Intuitively, it describes the strain (i.e. extension or contraction) of a piece of material relative to a unconstrained direction, if a force acts on it in an orthogonal direction. As illustrated in Figure 11.3, materials with high Poisson's ratio like rubber but also steel shrink in orthogonal unconstrained direction when a uniaxial force is applied. In contrast, materials with a Poisson's ratio near zero (e.g. cork) do not change in size along unconstrained direction. Finally, there exists a class of so called auxetic materials with a negative Poisson's ratio that stretch perpendicularly to the applied force.

## 3.2 Deformations of Surfaces

The general deformation setup considered in this thesis is shown in Figure 3.2. It parallels the setup for elastic solids as described in the last section, but essentially replaces solids by surfaces. The deformation itself is given as a map  $\tilde{x}$  defined on the undeformed surface  $\bar{S}$  that maps each point to its corresponding point on the deformed surface  $S$ . Similar to elastic solids, we are looking for an elastic potential for surfaces.

While elastic surfaces of infinitesimal thickness do not exist in the real world, specialized deformation models have been developed for *shells*, flexible thin walled structures with a high ratio of width to thickness. In general shells assume a curved, undeformed initial configuration, into which they relax in the absence of external forces. Examples can be found in structures that are either naturally curved like leaves, egg shells, and fingernails or put into that shape by plastic deformation like cans, carton boxes, car bodies, etc. For completeness we mention two special cases of shells: *Membranes* are shells that resist only material stretch and shear but not bending. A membrane deformed by pure bending will not relax to its initial configuration in the absence of other forces. Natural counterparts for membranes can be found in most textiles so that in computer graphics membrane models are typically used for cloth simulation (see e.g. [HB00, VCM05]).

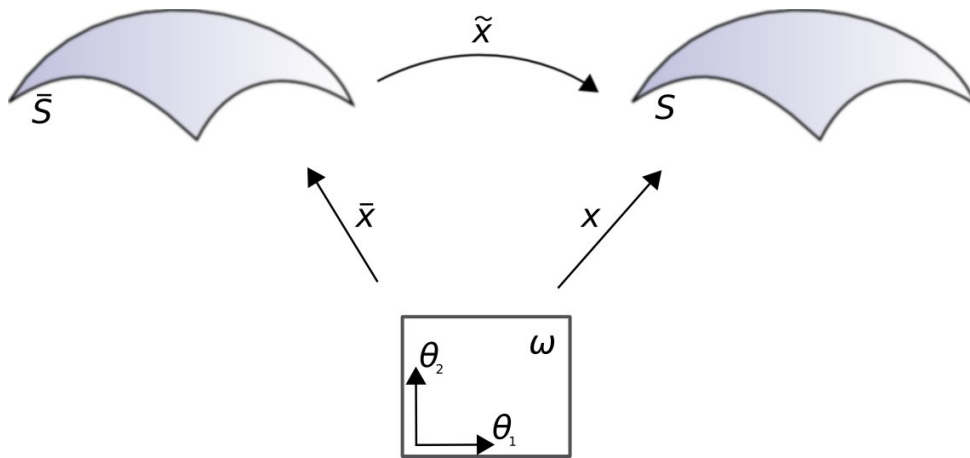


Figure 3.2: The surface deformation scenario: The original surface  $\bar{S}$  is deformed by a map  $\tilde{x}$  into a surface  $S$ . Both surfaces are parameterized over a common domain  $\omega$  by two parameterizations  $\bar{x}$  and  $x$ .

Another popular special case of shells are *thin plates* which relax to a flat rest configuration as e.g. steel plates or leaves of paper.

### 3.2.1 Elastic Shells

In contrast to the solid body model described in the last section, shell models represent shells by their middle surface  $S$  only. In these models the shell's deformation is completely determined by the deformation of its middle surface which is a valid approximation if the shell's thickness is constant and small. One way to obtain a shell model is to embed  $S$  into an appropriate sequence of elastic 3D bodies whose thickness vanishes in the limit. Applying given forces to these bodies results in deformations that can be derived by solving the constitutive equations of continuum mechanics. The deformation of the shell's middle surface is then obtained by taking the limit of solid body deformations.

As real shells always have a finite thickness, physical shell models have no natural counterpart in the strict sense, but only approximate the behaviour of real shells. Nevertheless, they prove very useful in the numerical treatment of thin shells, as the high ratio of width to thickness complicates meshing and hinders the application of finite element methods for solid bodies [GHDS03]. One of the simpler physical shell models is the Kirchhoff Love model for thin shells which we summarize below. Our description of the Kirchhoff Love theory roughly follows [COS00] but additional details can also be found in [GZ02].

Again, we consider the shell as a thin body  $\bar{B}$  and  $B$  that extends around

its middle surface  $\bar{S}$  and  $S$ . In fact, we assume that the shell's body in both configurations is thin enough to be represented by its middle surface only. As explained in Section 2.1 at each point of the surfaces coordinate systems can be defined by taking derivatives of the parameterizations. In context of shells, the normal vectors  $\bar{\mathbf{a}}_3$  and  $\mathbf{a}_3$  are also referred to as *shell director*.

Assuming a sufficiently small shell thickness, the shell  $\bar{B}$  can be parameterized in the following way

$$\bar{\mathbf{r}}(\theta^1, \theta^2, \theta^3) := \bar{\mathbf{x}}(\theta^1, \theta^2) + \theta^3 \bar{\mathbf{a}}_3(\theta^1, \theta^2) \text{ with } -\frac{h}{2} \leq \theta^3 \leq \frac{h}{2}$$

where  $h$  denotes the thickness of the shell. Now, the fundamental assumptions of the Kirchoff Love model are, that the deformation does not change the thickness of the shell and that directions orthogonal to the middle surface  $\bar{S}$  remain orthogonal to the deformed middle surface  $S$ . In particular, this assumption holds for the shell director  $\bar{\mathbf{a}}_3$ . As the shell thickness is small, the deformed shell  $B$  is therefore given by a parameterization similar to the above

$$\mathbf{r}(\theta^1, \theta^2, \theta^3) := \mathbf{x}(\theta^1, \theta^2) + \theta^3 \mathbf{a}_3(\theta^1, \theta^2) \text{ with } -\frac{h}{2} \leq \theta^3 \leq \frac{h}{2}.$$

As the deformation  $\tilde{\mathbf{r}}$  of the shell  $\bar{B}$  is equivalent to the composition of the parameterizations  $\mathbf{r} \circ \bar{\mathbf{r}}^{-1}$ , it is thus completely determined by the deformation of its middle surface  $\bar{S}$ .

For the shell case, the first fundamental form  $G_{ij}$  of  $\mathbf{r}$  (and analogously  $\tilde{G}_{ij}$  for  $\tilde{\mathbf{r}}$ ) now takes the following form:

$$\begin{aligned} G_{\alpha\beta} &= \mathbf{r}_{,\alpha} \cdot \mathbf{r}_{,\beta} \\ &= (\mathbf{x}_{,\alpha} + \theta^3 \mathbf{a}_{3,\alpha}) \cdot (\mathbf{x}_{,\beta} + \theta^3 \mathbf{a}_{3,\beta}) \\ &= g_{\alpha\beta} + \theta^3 \mathbf{x}_{,\alpha} \cdot \mathbf{a}_{3,\beta} + \theta^3 \mathbf{x}_{,\beta} \cdot \mathbf{a}_{3,\alpha} + (\theta^3)^2 \mathbf{a}_{3,\alpha} \cdot \mathbf{a}_{3,\beta} \end{aligned}$$

and by Equation 2.4 this can be further simplified using the second fundamental form  $h_{\alpha\beta}$  to

$$G_{\alpha\beta} = g_{\alpha\beta} - 2\theta^3 h_{\alpha\beta} + (\theta^3)^2 \mathbf{a}_{3,\alpha} \cdot \mathbf{a}_{3,\beta}$$

For the five remaining entries of the  $G_{ij}$  we obtain:

$$G_{33} = \mathbf{r}_{,3} \cdot \mathbf{r}_{,3} = \mathbf{a}_3 \cdot \mathbf{a}_3 = 1$$

$$\begin{aligned} G_{\alpha 3} &= G_{3\alpha} = \mathbf{r}_{,\alpha} \cdot \mathbf{r}_{,3} \\ &= (\mathbf{x}_{,\alpha} + \theta^3 \mathbf{a}_{3,\alpha}) \cdot \mathbf{a}_3 \\ &= \mathbf{x}_{,\alpha} \cdot \mathbf{a}_3 + \theta^3 \mathbf{a}_{3,\alpha} \cdot \mathbf{a}_3 = 0 \end{aligned}$$

In the last equality the term  $\mathbf{x}_{,\alpha} \cdot \mathbf{a}_3$  vanishes by definition of  $\mathbf{a}_3$ . The second term vanishes because of  $\mathbf{a}_3 \cdot \mathbf{a}_3 = 1$  and thus:

$$0 = \frac{\partial}{\partial \theta^\alpha} (\mathbf{a}_3 \cdot \mathbf{a}_3) = 2\mathbf{a}_{3,\alpha} \cdot \mathbf{a}_3$$

Plugging these expressions for the first fundamental forms  $\bar{G}$  and  $G$  into Equation 3.1 we find the Green-Lagrange strain tensor as

$$\begin{aligned} \gamma_{\alpha\beta} &= \frac{1}{2} (g_{\alpha\beta} - \bar{g}_{\alpha\beta}) + \theta^3 (h_{\alpha\beta} - \bar{h}_{\alpha\beta}) \\ \gamma_{\alpha 3} &= \gamma_{3\alpha} = \gamma_{33} = 0 \end{aligned}$$

to first order in the shell thickness  $h$  which is a reasonable approximation under our assumption that  $h$  is small. For later reference, we further introduce the following two abbreviations for the first and second terms in the strain tensor:

$$\alpha_{\alpha\beta} := \frac{1}{2} (g_{\alpha\beta} - \bar{g}_{\alpha\beta}) \quad (3.10)$$

$$\beta_{\alpha\beta} := h_{\alpha\beta} - \bar{h}_{\alpha\beta} \quad (3.11)$$

The tensor  $\alpha_{\alpha\beta}$  is also referred to as *membrane strain* as it captures the strain components within or tangential to the surface  $\bar{S}$ . In contrast, the difference of the second fundamental forms  $\beta_{\alpha\beta}$  is called *bending strain* since it measures bending, i.e. changes in the curvature of the shell. From these expression it also follows, that under the above stated assumptions of the Kirchoff-Love model all relevant strain measures within the shell can be deduced from the deformation of the middle surface  $\bar{S}$ , more specifically from the first and second fundamental forms only.

Using the above strain tensor and the general stress/strain relation given Equation 3.8 it is possible derive an expression for the stress tensor  $\tau^{ij}$  in terms of  $g_{ij}$ ,  $h_{ij}$ , and  $\theta^3$ . However, instead of considering the actual stress at each point of  $B$ , the classical shell model introduces another simplification by considering only an equivalent system of forces and moments resulting on the shell's middle surface. Under the assumption of an isotropic material, corresponding tensors  $n^{\alpha\beta}$  and  $m^{\alpha\beta}$  are found (see [GZ02] Chapter 12) by integrating the actual stresses in the direction of the shell director through  $B$  as

$$n^{\alpha\beta} = \frac{Eh}{(1-\nu^2)} H^{\alpha\beta\gamma\delta} \alpha_{\gamma\delta} \quad (3.12)$$

$$m^{\alpha\beta} = \frac{Eh^3}{12(1-\nu^2)} H^{\alpha\beta\gamma\delta} \beta_{\gamma\delta}$$

$$\text{with } H^{\alpha\beta\gamma\delta} = \nu \bar{g}^{\alpha\beta} \bar{g}^{\gamma\delta} + \frac{1}{2}(1-\nu) (\bar{g}^{\alpha\gamma} \bar{g}^{\beta\delta} + \bar{g}^{\alpha\delta} \bar{g}^{\beta\gamma}) .$$

As the tensor  $n^{\alpha\beta}$  depends only on the first fundamental form, it is also referred to as *membrane stress tensor* while  $m^{\alpha\beta}$  is called the *bending stress tensor*. In analogy to the general case, the elastic potential energy for a shell is now obtained by taking the integral with respect to membrane and bending stresses. Its density function is therefore given as

$$e = e_{\text{membrane}} + e_{\text{bending}} \text{ where} \quad (3.13)$$

$$e_{\text{membrane}} := \frac{1}{2} \frac{Eh}{(1-\nu^2)} H^{\alpha\beta\gamma\delta} \alpha_{\alpha\beta} \alpha_{\gamma\delta} \quad (3.14)$$

$$e_{\text{bending}} := \frac{1}{2} \frac{Eh^3}{12(1-\nu^2)} H^{\alpha\beta\gamma\delta} \beta_{\alpha\beta} \beta_{\gamma\delta} \quad (3.15)$$

We like to point out some simple observations that follow directly from the above expression of a shell's elastic potential. First, we remark that  $e$  falls into two separate terms that capture tangential deformations, i.e. membrane strain and surface bending, i.e. bending strain respectively. It therefore makes sense to refer to these terms as *membrane potential* and *bending potential*. While membrane potential is a function of differences in the metric tensor of the middle surface, the bending potential is expressed in terms of differences in the second fundamental form. Moreover, it can be observed that the bending potential is of third order in the shell thickness  $h$  while membrane potential is only of first order. With decreasing shell thickness, a shell's deformation behavior is thus dominated by membrane stresses. For this reason, many discrete models for cloth simulation omit bending stresses completely.

### 3.2.2 Membrane Potential and Parameterization

In Part II we will discuss potentials for surface parameterization. As will be explained there, surface parameterization can be interpreted in context of deformations with a planar undeformed configuration  $\bar{S} \subset \mathbb{R}^2$ . To show similarities with these potentials, we further rewrite the membrane potential for this special case. While in the above form the membrane potential  $e_{\text{membrane}}$  is given as a function of the difference  $g_{ij} - \bar{g}_{ij}$  and  $\bar{g}_{ij}$ , it turns out, that it actually solely depends on quotient of the metric tensors:

$$\begin{aligned} e_{\text{membrane}} &= \frac{1}{8} \frac{Eh}{(1-\nu^2)} H^{\alpha\beta\gamma\delta} (g_{\alpha\beta} - \bar{g}_{\alpha\beta}) (g_{\gamma\delta} - \bar{g}_{\gamma\delta}) \\ &= \frac{Eh}{8(1-\nu^2)} H^{\alpha\beta\gamma\delta} (g_{\alpha\beta} g_{\gamma\delta} - g_{\alpha\beta} \bar{g}_{\gamma\delta} - \bar{g}_{\alpha\beta} g_{\gamma\delta} + \bar{g}_{\alpha\beta} \bar{g}_{\gamma\delta}) . \end{aligned}$$

Defining the quotient of the metric tensors as  $k_{\beta}^{\alpha} = \bar{g}^{\alpha\gamma} g_{\gamma\beta}$  and using the symmetry of the metric tensors forms, we find the following simple relations:

$$\begin{aligned}\bar{g}^{\alpha\beta} g_{\alpha\beta} &= \bar{g}^{\alpha\beta} g_{\beta\alpha} = k_{\alpha}^{\alpha} \\ g^{\alpha\beta} g_{\alpha\beta} &= g^{\alpha\beta} g_{\beta\alpha} = \delta_{\alpha}^{\alpha} = 2 \\ \bar{g}^{\alpha\gamma} g_{\alpha\beta} &= \bar{g}^{\alpha\gamma} g_{\beta\alpha} = \bar{g}^{\gamma\alpha} g_{\alpha\beta} = k_{\beta}^{\gamma}\end{aligned}$$

and thus

$$\begin{aligned}H^{\alpha\beta\gamma\delta} g_{\alpha\beta} g_{\gamma\delta} &= \nu \bar{g}^{\alpha\beta} \bar{g}^{\gamma\delta} g_{\alpha\beta} g_{\gamma\delta} + \frac{1}{2}(1 - \nu) (\bar{g}^{\alpha\gamma} \bar{g}^{\beta\delta} g_{\alpha\beta} g_{\gamma\delta} + \bar{g}^{\alpha\delta} \bar{g}^{\beta\gamma} g_{\alpha\beta} g_{\gamma\delta}) \\ &= \nu (k_{\alpha}^{\alpha})^2 + (1 - \nu) k_{\beta}^{\gamma} k_{\gamma}^{\beta} \\ H^{\alpha\beta\gamma\delta} g_{\alpha\beta} \bar{g}_{\gamma\delta} &= H^{\alpha\beta\gamma\delta} \bar{g}_{\alpha\beta} g_{\gamma\delta} \\ &= \nu \bar{g}^{\alpha\beta} \bar{g}^{\gamma\delta} g_{\alpha\beta} \bar{g}_{\gamma\delta} + \frac{1}{2}(1 - \nu) (\bar{g}^{\alpha\gamma} \bar{g}^{\beta\delta} g_{\alpha\beta} \bar{g}_{\gamma\delta} + \bar{g}^{\alpha\delta} \bar{g}^{\beta\gamma} g_{\alpha\beta} \bar{g}_{\gamma\delta}) \\ &= 2\nu k_{\alpha}^{\alpha} + (1 - \nu) k_{\alpha}^{\alpha} = (\nu + 1) k_{\alpha}^{\alpha} \\ H^{\alpha\beta\gamma\delta} \bar{g}_{\alpha\beta} \bar{g}_{\gamma\delta} &= \nu \bar{g}^{\alpha\beta} \bar{g}^{\gamma\delta} \bar{g}_{\alpha\beta} \bar{g}_{\gamma\delta} + \frac{1}{2}(1 - \nu) (\bar{g}^{\alpha\gamma} \bar{g}^{\beta\delta} \bar{g}_{\alpha\beta} \bar{g}_{\gamma\delta} + \bar{g}^{\alpha\delta} \bar{g}^{\beta\gamma} \bar{g}_{\alpha\beta} \bar{g}_{\gamma\delta}) \\ &= 4\nu + 2(1 - \nu) = 2\nu + 2.\end{aligned}$$

Assembling these expression, the membrane energy for an isotropic material is found as

$$\begin{aligned}e_{\text{membrane}} &= \\ \frac{Eh}{8(1 - \nu^2)} &(\nu (k_{\alpha}^{\alpha})^2 + (1 - \nu) k_{\beta}^{\gamma} k_{\gamma}^{\beta} - 2(\nu + 1) k_{\alpha}^{\alpha} + 2\nu + 2) .\end{aligned}\quad (3.16)$$

If  $\mathbf{k}$  denotes the matrix with entries  $k_{\beta}^{\alpha}$ , we have  $k_{\alpha}^{\alpha} = \text{tr}(\mathbf{k})$  and further  $k_{\beta}^{\gamma} k_{\gamma}^{\beta} = \text{tr}(\mathbf{k}^2)$ . It then becomes obvious, that the membrane potential is only a function of the singular values of the quotient  $\mathbf{k}$  and of the squared quotient  $\mathbf{k}^2$ .

In Part II we consider mappings from a flat domain onto a surface and derived parametric deformation measures. The above described elastic shell model can also be applied to this special case. For a flat undeformed configuration,  $\tilde{\mathbf{x}}$  maps from a planar domain and we can define the metric tensor  $\tilde{\mathbf{g}}$  of  $\tilde{\mathbf{x}}$  analogously to  $\mathbf{g}$  as

$$\tilde{g}_{\alpha\beta} := \tilde{x}_{i,\alpha} \tilde{x}_{i,\beta} = x_{i,\gamma} \bar{x}_{\gamma,\alpha}^{-1} x_{i,\delta} \bar{x}_{\delta,\beta}^{-1} = g_{\gamma\delta} \bar{x}_{\gamma,\alpha}^{-1} \bar{x}_{\delta,\beta}^{-1} = ((\nabla \bar{\mathbf{x}})^t \mathbf{g} \nabla \bar{\mathbf{x}}^{-1})_{\alpha\beta}$$

and we have

$$\text{tr}(\mathbf{k}) = \text{tr}(\tilde{\mathbf{g}}) = \lambda_{\max} + \lambda_{\min} \text{ and } \text{tr}(\mathbf{k}^2) = \text{tr}(\tilde{\mathbf{g}}^2) = \lambda_{\max}^2 + \lambda_{\min}^2 \quad (3.17)$$

where  $\lambda_i$  denotes the eigenvalues of  $\tilde{\mathbf{g}}$ . The elastic energy thus depends only on the eigenvalues of the metric tensor  $\tilde{\mathbf{g}}$ . This property is characteristic for potentials used for surface parameterization. To allow for better comparison, all of these potentials as they are discussed in Part II will be given in this form.





## **Part II**

# **Deformation Potentials for Surface Parameterization**



---

The visual perception of shapes is to a large part dominated by their surface. Consequently, as a science concerned with the appearance of objects, computer graphics has a great need for techniques that help to describe, store, compress, manipulate, and visualize surfaces. One of the most fundamental and oldest techniques in this context are surface parameterizations. Mapping from complicated surfaces in three (or higher) dimensional space into the familiar Euclidean space, parameterizations help to cope with the inherent complexity of surfaces.

Long before their use in computer graphics, parameterizations therefore have found widespread applications in many other fields as e.g. mathematics or geography. The most commonplace incarnation of a parameterization is the geographic map, that allows to treat a complex shaped three dimensional terrain as a flat, two dimensional object.

In computer graphics, while originally used primarily for texture mapping, parameterizations are today one of the most powerful tools in the geometry processing toolchain. Applications range from remeshing (the task of finding a triangulation with certain properties for a given surface), surface reconstruction, and surface painting [LPRM02b] to surface editing [BMBZ02], shape analysis, shape matching and geometry images [GGH02].

Formally, the parameterization problem can be put as follows: Given an orientable genus zero surface patch  $S$  find a homeomorphism  $\tilde{x}$  that maps from a planar set  $\bar{S} \subset \mathbb{R}^2$  to  $S$  (see Figure 3.3). (In case of a regular surface  $S$ ,  $\tilde{x}$  is also required to be differentiable with full rank Jacobian in accordance with the definition of regular surfaces given in Section 2.1.)

Solutions to the parameterization problem as stated above are certainly not unique: Concatenating an arbitrary parameterization  $\tilde{x}$  e.g. with a bijective map of the plane onto itself yields again a parameterization. However, in most computer graphics applications, parameterizations with certain shape preserving properties are desired. If a flat shape in  $\bar{S}$  is mapped to  $S$  via the parameterization, its geometry, i.e. angles, its area or length should be preserved as much as possible. Maps that perfectly preserve all these shape properties are also called *isometric*. The requirement of a shape preserving parameterization is in perfect analogy to geographic projections like e.g. the Mercator projection that is designed to preserve angles. A map that renders shape arbitrarily deformed gives a false impression of the depicted terrain and cannot be used for navigation.

Finding a shape preserving parameterization can also be regarded as a special case of the general surface deformation problem described in Section 3.2 (see Figure 3.3). For parameterization, we simply restrict our attention to planar undeformed surfaces  $\bar{S}$ . Moreover, in contrast to the classical surface deformation problem, where the surface  $\bar{S}$  is fixed and a deformed surface  $S$  is searched for, we solve for parameterization the following inverse problem: Find a flat “ursurface”  $\bar{S}$  and associated deformation  $\tilde{x}$  such that the shape deformation is minimal. The

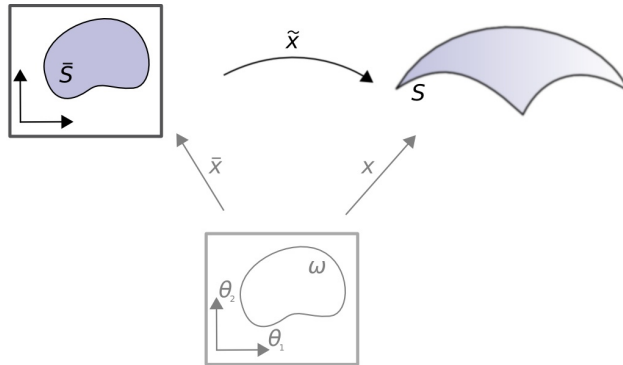


Figure 3.3: Definition of a parameterization as a (smooth) function that maps from a set  $\bar{S} \subset \mathbb{R}^2$  to the surface  $S$ . In gray: Surface parameterization can be regarded as a special case of the general surface deformation setup described in Section 3.2 where the undeformed surface  $\bar{S}$  is flat.

notion of shape deformation in this context is defined via the potential.

Using this interpretation, it is in fact possible to find parameterizations using the elastic shell model (or more precisely the elastic potential) as introduced in Section 3.2.1. However, parameterizations obtained in this way are not well suited for most graphics applications as we will see in Chapter 4. After revising a list of desirable properties, that chapter will therefore derive a novel potential that leads to parameterizations more apt to applications in computer graphics.

In Chapter 7 we will then turn to a special class of objects: Fabric covered surfaces as e.g. seats are either made from or covered with a fabric that is initially supplied in a planar form. This fabric is then put into a desired shape during the production process. The resulting fabric deformation depends not only on the final shape but also on the fabrics materials. We argue that for objects of this class texture maps are desirable which imitate the physical, material specific deformation pattern rather than minimizing geometric shape deformation. To compute such *material specific texture maps*, we revisit the elastic shell model and derive a parameterization algorithm. The resulting potentials and techniques are also put to work in an alternative application: We will see how they can be used to compute sewing patterns for upholstery in industrial design.

The parameterization problem as stated above is only well posed if  $S$  is manifold and of genus zero. While the genus can be reduced by cutting the surface into charts in a preprocessing step, manifoldness is more difficult to fix. Moreover, in practice surfaces are often corrupted by triangulation artifacts. Chapter 8 proposes an approach that circumvents these problems and allows to apply standard parameterization methods to triangulations with inconsistencies.

---

## POTENTIALS FOR SURFACE PARAMETERIZATIONS

---

### 4.1 Introduction

When using a geographic map to navigate through a foreign territory we rely on its property to respect the form and extent of shapes in some sense, i.e. the distances, areas or angles. The more a map respects such metric properties the easier it is to use it for navigation purposes. As argued in the introduction to this part of the thesis, many tasks in computer graphics benefit from parameterizations that respect shapes. By mapping a texture image onto a surface for instance we usually want the image to appear on the surface without any visible deformation or distortions like stretch, shear or shrink. An isometric parameterization is thus highly desirable.

Unfortunately, it follows by Gauss's "Theorema egregium" that a perfect shape preserving parameterization does only exist for surfaces with vanishing Gauss curvature, so called *developable surfaces* like plane or cylinder. In general, surfaces do not have this property. This is in particular true for a sphere implying that there is no geographic map of the earth that perfectly preserves all shapes. While the Mercator projection for instance respects all angles, it does not respect areas and distances. Although a perfect shape preserving map does not exist in general, geographic maps and parameterizations should respect shapes as much as possible. The exact meaning of "as much as possible" in this context is clearly subject to the underlying deformation potential. Its choice is critical for every parameterization algorithm as it directly determines the quality and properties of the resulting parameterization.

Due to the significance of the parameterization problem, many different potentials have been proposed over the last years. In Section 4.2 we give a survey of potentials that measure metric deformations. We also analyze there the impact of metric properties like angle, area and length deformation on these potentials and show up relations. In Chapter 5 then a novel potential for surface parameterization is proposed, that measures metric deformations in terms of two geometric prop-

erties, namely angle and area deformation. The relative importance of angle and area preservation can be specified via a parameter. On surfaces for which no isometric parameterization exists, the inherent trade off between angle and area can thus be controlled by the user. Besides its clear relation to geometric properties, our potential has two important desirable properties: First, it can be optimized in a way that guarantees the admissibility of the parameterization, i.e. its bijectivity. We propose to this extent a customized optimization algorithm whose details are described in Chapter 6. Second, in contrast to many other potentials, our potential can be optimized in the absence of further boundary constraints. The shape of the domain  $\bar{S}$  is thus obtained naturally from the optimization of the potential and its boundaries evolve freely to minimize the overall shape deformation. Finally, we also present results of an evaluation of our method using a number of different data sets and compare them to results obtained by prior methods in Chapter 6.

#### 4.1.1 Parameterization of Triangle Meshes

In the smooth case, the surface  $S$  is regular and it must be orientable to support a single parameterization covering the entire surface as outlined in Section 2.1. In the discrete case, we make analogous assumptions on the triangle mesh  $\mathcal{M}$ , i.e. we assume a manifold and oriented mesh. Moreover, we assume the existence of a fixed auxiliary parameterizations  $\mathbf{x} : \omega \rightarrow S$ . This auxiliary parameterization is convenient to simplify the following discussion of parameterization potentials and algorithms. However, it serves only in the theoretical discussion and the final parameterization method described in Chapter 6 does not rely on its existence.

According to the triangle mesh structure of  $S$  and we assume that both the inverse of the auxiliary parameterization  $\mathbf{x}$  and the sought-after parameterization  $\tilde{\mathbf{x}}$  are affine maps within each triangle  $T_S$ . In computer graphics parameterizations  $\tilde{\mathbf{x}}$  were originally used for texture mapping. The coordinates  $\bar{\mathbf{x}}^v \in \bar{S}$  are thus often referred to as *texture coordinates*. It will sometimes be useful to consider the *inverse parameterization*  $\tilde{\mathbf{x}}^{-1}$  that maps the surface  $S$  into the domain  $\bar{S}$ .

As outlined in Section 2.1 a surface  $S$  supports a single parameterization covering the whole surface only if  $S$  is orientable and of genus zero with at least one boundary loop. Some parameterization methods work only for surfaces with a single connected boundary loop. However, this is a purely technical limitation of these methods and not necessary for the existence of a parameterization. Thus we do not assume the surface  $S$  to have a single boundary loop. In general a surface  $S$  might not fulfill these topological requirements, either because its non orientable or has a higher genus. To parameterize such surfaces, automatic cutting and charting algorithms have been developed that can be applied in a preprocessing step. These methods output several connected patches of genus zero that can then be parameterized separately. Cutting down the surface into several patches (or charts)

also reduces the Gaussian curvature and increases boundary length, which in turn can in some cases drastically reduce shape deformation. On the other hand, a cut is a discontinuity in the parameterization. Depending on the application, this can also cause problems as e.g. visible cracks in texture mapping. In general, an application specific tradeoff has to be made between the number of cuts and the quality of the parameterization of the patches. However, cutting and charting algorithms are in general beyond the scope of this thesis ( although we will come back to them in Chapter 8). For simplicity, we assume in the following that the surface  $S$  is connected and of genus zero.

## 4.2 Previous and Related Work on Parameterization

As one of the most important tools in the geometry processing tool box, parameterization has been intensively studied in the last decades. While the number of approaches is vast, recently excellent surveys have been compiled by Floater and Hormann [FH05] and Hormann et al. [HLS07]. Instead of discussing each method in detail, we will introduce in this section the fundamental approaches and refer for details to these surveys. After giving an overview of approaches in this section, the discussion in Section 4.2.2 will lie special emphasis on the employed potentials.

In computer graphics, the parameterization problem was first addressed by Bennis *et al.* [BVI91] who flatten a given surface by progressively laying out surface geodesics in the plane. This concept of progressive surface unfolding was subsequently used in many other approach like e.g. [SCOGL02], but also in computer aided design for reverse engineering (see [AS04] and [YZS07] for a survey). Progressive unfolding works well for nearly developable surfaces. However, on general surfaces, the number and structure of patches or cracks is hard to control.

Based on earlier work by Tutte [Tut63, Tut60], the approach of Floater [Flo97] was the first to guarantee a admissible parameterization, i.e. bijectivity of the map. The texture coordinate for each vertex is chosen as a strictly convex combination of the texture coordinates of its neighbors. It can be shown that these maps are bijective if the boundary of the texture mesh is fixed on a convex shape. As shown in [Hor01], the method of Floater fits into the so called *edge-spring model*. In this model, parameterizations are computed as configurations of minimal energy of a mass-spring model, where each vertex is associated with a mass and (half)edge with springs of zero rest length. By choosing spring constants appropriately several potentials can be minimized in this framework, among them the important Dirichlet energy that is discussed in Section 4.2.2. All potentials that can be de-

scribed by the edge-spring model require that texture coordinates at boundary vertices have to be fixed, since otherwise the spring-mass system would simply collapse into a single point.

After the seminal work of Floater, a large body of work concentrated on parameterizations that are not only admissible, but also minimize some kind of shape deformation. Besides geometric properties, there exists also a class of approaches [SGSH02, BTB02, SWB98, HC00] that optimize the parameterization for optimal sampling of a given surface signal. Approaches of both kinds vary most of all in the choice of the potential and the techniques for its minimization. The method proposed in this part of the thesis also falls into the first category. We will have a closer look on these approaches and the employed potentials in Section 4.2.2.

Much research has also been done on automatic cutting or charting algorithms [SSGH01a, SH02, LPRM02a, JKs05], that cut surfaces of higher genus into charts that are topologically equivalent to a disk. The charts can then be parameterized in an independent second step. As the structure of cuts has a high influence on the resulting deformation, separating cutting and parameterization is clearly suboptimal.

As an alternative, two fundamental approaches have been proposed: Methods following the first approach try to parameterize the surface not over a disk but over a domain of matching topology. This approach is consequently called *non-planar parameterization*. For closed genus zero surfaces such a domain is the unit sphere and specialized methods have been developed (see [GGs03]). For higher genus surfaces a domain with matching topology has to be found e.g. as a simplified version of the original surface [EDD<sup>+</sup>95, LSS<sup>+</sup>98, KLP03] or as an enclosing combination of several unit cubes ( so called *polycube* ) [THCM04]. The second principle approach is called *global parameterization*. Methods of this class [GY03, SSP08] solve for an optimal parameterization and an abstract base domain at the same time placing discontinuities or singular points where they most effectively reduce shape deformation. Targeted at remeshing, some methods [RLL<sup>+</sup>06, TACSD06, KNP07, BZK09] also enforce placements that coincide with certain surface characteristics as features, curvature lines or umbilical points. We would like to note, that even though both non-planar and global parameterization overcomes the restriction of planar parameterization, such methods are limited to certain application where the higher complexity of the base domain is not a problem. While, for example, it is easy to pattern a surface with regular tiles using a planar parameterization, this task is not trivial for topologically more complex domains.

Similarly to non-planar parameterization, *cross parameterization* methods (see [SPR06] for a survey) also try to find a parameterization over a non-planar domain. In contrast to non planar parameterization, this domain is not derived from the surface itself, but given as an additional input. By establishing a homeomorphism



between two given surfaces, cross parameterizations allow to transfer surface signals like high frequency detail, color, normal maps etc. from one surface onto another. Other applications are morphing or shape matching.

Finally, efforts have recently been made to increase the speed and robustness of parameterization algorithms. Based on the parameterization potentials proposed in [SS01], Sheffer et al. proposed several improvements to the non-linear optimization which yields drastic speedups. Similar approaches in this line of work are the iterative method by Dong and Garland [DG07] and the alternating global/local optimization by Liu et al. [LZX<sup>+</sup>08] which both can be used to optimize a variety of parameterization potentials.

More recent than our work is the approach of Clarenz et al. [CLR04a]. They identified three basic desirable properties of parameterization potentials and derived from these a class of potentials that can be regarded as a generalization of the combined potential  $E_{combined}$  proposed in Section 5.4. For this class of potentials they were able to prove the existence of a minimizing parameterization even in the smooth case. We will relate our work to their results in Section 6.3.

### 4.2.1 Important Properties of Parameterization Potentials

The primary distinguishing feature of planar parameterization methods is certainly their underlying potential and most characteristics of the parameterizations computed by a particular method can be directly deduced from it. While the following section gives an overview of the most important potentials used for parameterization, we want to describe in this section some important properties of potentials and their implications for the resulting parameterization.

#### Bijectivity and Flips

As explained in section 4.1.1 in the discrete case a continuous piecewise affine parameterization is uniquely determined by its set of texture coordinates  $(\bar{x}^v)_{v \in V}$ .

However, not each such set determines an admissible parameterization  $\tilde{\mathbf{x}}$ . In fact, it is not difficult to find texture coordinates  $(\bar{x}^v)_{v \in V}$ , for which the inverse parameterization  $\tilde{\mathbf{x}}^{-1}$  is not bijective. For example, if the texture coordinates of the vertices of a triangle  $T \in \mathcal{T}$  are colinear, the texture triangle  $T_{\tilde{\mathbf{x}}} = \tilde{\mathbf{x}}^{-1}(T_S)$  degenerates to a line or a point. But even if no triangle is degenerated, two triangles  $T_{\tilde{\mathbf{x}}}^1$  and  $T_{\tilde{\mathbf{x}}}^2$  can intersect each other and thus violate the bijectivity as depicted in Figure 4.1. There exist two types of bijectivity violations:

1. The boundary of the texture mesh  $((\bar{x}^v)_{v \in V}, \mathcal{T})$  intersects itself. This rarely happens in practice and can be handled by cutting the surface patch along

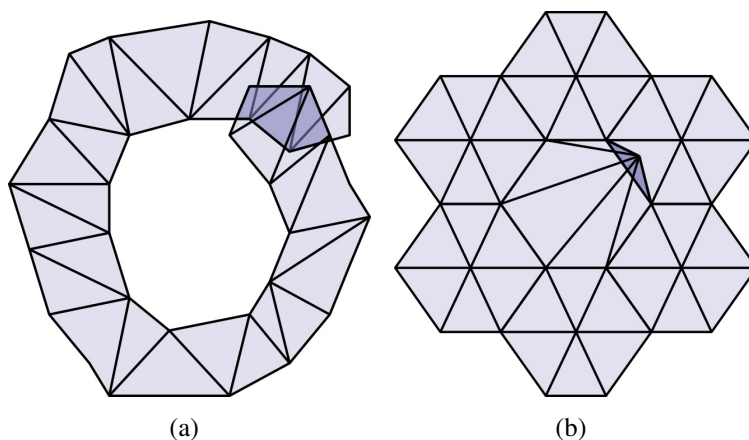


Figure 4.1: **Invalid texture coordinates** cause the texture mesh to fold over and thus do not define bijective parameterizations. There are two possible causes: (a) boundary intersections or (b) triangle flips

the borders of the intersection in a post-processing step as described in [Sds01].

2. A triangle flip denotes two adjacent triangles  $T_{\bar{S}}^1$  and  $T_{\bar{S}}^2$  that have opposite orientations, i.e. running through the vertices of each triangle in the order specified in  $\mathcal{T}$  (Remember that we assume the mesh  $\mathcal{M}$  to be oriented), one triangle is clockwise oriented and the other counterclockwise. While a single, isolated flip is usually easy to resolve, triangle flips in general cannot easily be handled in a post-processing step.

It is therefore reasonable, to require that a parameterization potential should only attain its minimum for texture coordinates  $(\bar{x}^v)_{v \in V}$  that consistently orient all the triangles  $T_{\bar{S}}$  in the parameterization domain  $\bar{S}$ , i.e. either clockwise or counterclockwise. While this does not single out boundary intersections (which are easy to handle), it prevents triangle flips.

### Quadratic Potentials

Some of the potentials discussed in the following are quadratic in the texture coordinates while others are of higher order. Minimizing a quadratic potentials leads to a simple linear equation system in the texture coordinates  $(\bar{x}^v)_{v \in V}$ . Hence, we refer to parameterization methods based on quadratic potentials as *linear methods*.

As the Hessian of quadratic parameterization potentials is usually sparse (with a sparsity pattern that coincides with the vertex adjacency matrix of the mesh), sparse direct solvers like sparse *LU* or Cholesky decomposition can be applied

which have shown to be very efficient for such matrices [BBK05]. Furthermore, it is often easy to show the existence and the uniqueness of a solution or to provide space and time complexity bounds for these methods.

On the other hand the expressiveness of quadratic potentials is clearly limited. For example, as triangle area by itself is a quadratic expression in the texture coordinates, preservation of area cannot be enforced using a quadratic function (a formal proof was given in [Deg03]). Unfortunately, the question for the existence and uniqueness of a minimum becomes more difficult for functionals of higher order. In addition, nonlinear potentials are likely non convex and optimizers are prone to be stuck in local minima. For a successful optimization a reasonable initialization is therefore crucial. To this extent, solutions computed by a linear methods or hierarchical minimization can be used.

### Boundary Conditions

For some potentials appropriate conditions must be imposed on vertices at the boundary of the mesh, in order for the optimization to converge to an admissible parameterization. For example, the Dirichlet energy, that will be described in Section 4.2.2 favors parameterizations over a domain  $\bar{S}$  with small area. In the absence of boundary constraints, minimization of this energy tends to collapse all texture coordinates into a single point. In order to prevent this collapse, the texture coordinates of boundary vertices must be fixed prior to the minimization.

As the optimal shape for the boundary is not known in advance, one common approach is to map boundary vertices to a circle or some regular n-gon in  $\bar{S}$ . The distances of adjacent vertices are chosen to be proportional to the distances along the boundary curve of the surface. Certainly this choice of the boundary is somewhat arbitrary, and fixed boundary vertices restrict the minimization to solutions that may not be optimal. Potentials, that do not need such boundary conditions are therefore in general preferable.

However, fixing boundary conditions often enables other advantages as a quadratic formulation or guaranteed admissibility. Moreover, in some applications a fixed boundary shape is even desired. The generation of geometry images [GGH02], for example, requires the texture mesh to be rectangular.

## 4.2.2 Potentials based on Metric Deformation

In this section we review the most important potentials used for planar parameterization and describe their properties. Common to all potentials is their “intrinsic” formulation, i.e. they are expressed in terms of the metric tensor. As explained in the introduction to this chapter, an isometric parameterization that preserves all intrinsic properties is desirable for many applications. While this only exists for

a few surfaces, it is nevertheless desirable to have a parameterization that is as close to an isometry as possible. All parameterization potentials discussed below are therefore functions of the metric tensor  $\tilde{g}$ , that quantify, how much the metric tensor deviates from identity.

### Conformality (Angles)

While isometric parameterizations do not exist for general surfaces, conformal maps as discussed in Section 2.1.1 still have many desirable properties like angle preservation and the uniformization theorem guarantees their existence in the smooth case. They are sometimes even said to be locally shape preserving because the conformal factor  $\lambda$  varies smoothly over the surface. If we restrict our attention to a sufficiently small neighborhood around a point on the surface, the conformal factor is approximately constant and the parameterization within this neighborhood nearly isometric (up to a constant scaling factor).

However, the uniformization theorem requires a smooth regular surface to guarantee the existence of a conformal parameterization. It is easy to see, that conformal parameterizations do not, in general exist for triangulated surfaces: In the plane  $\bar{S}$ , the tip angles of triangles meeting at each vertex sum to  $2\pi$ . As conformal maps preserve angles they can only exist for triangulated surfaces  $S$  that show the same property. This, however, holds only for flat triangulations.

To generalize conformal parameterizations to the discrete case of a triangle surface, many different notions of a *discrete conformal* parameterization have been suggested, based on one (or more) of the various properties of smooth conformal maps.

**Dirichlet Energy:** One of the first attempts in computer graphics to generalize conformal maps was proposed by Eck et al. [EDD<sup>+</sup>95] and based on the concept of *harmonic maps*. Harmonic maps (see [EL88, EL78] for details on harmonic maps and [Hor01] for harmonic maps in the context of parameterization) minimize the so called *Dirichlet Energy*. For a map  $\mathbf{y} : U \subset \mathbb{R}^2 \rightarrow \mathbb{R}^2$  it is defined as

$$E_{Dirichlet}(\mathbf{y}) := \frac{1}{2} \int_U \|\nabla \mathbf{y}\|^2 dU$$

where  $\|\cdot\|$  denotes the Hilbert-Schmidt matrix norm. Pinkall and Polthier [PP93], who used harmonic maps to compute discrete minimal surfaces, pointed out that the Dirichlet energy can be split into two parts:

$$E_{Dirichlet} = E_{conformal} + \int_{\mathbf{y}(U)} dA \tag{4.1}$$

with

$$E_{conformal} := \int_U |\mathbf{R}^{\pi/2} \mathbf{y}_{,1} - \mathbf{y}_{,2}|^2 \quad (4.2)$$

$$= \int_U ((y_{1,1} - y_{2,2})^2 + (y_{1,2} + y_{2,1})^2) \quad (4.3)$$

where  $\mathbf{R}^{\pi/2}$  denotes a counterclockwise rotation in the plane by  $\pi/2$ . If the map  $\mathbf{y}$  is conformal, we have for the angle  $\angle \mathbf{y}_{,1} \mathbf{y}_{,2} = \pi/2$  and the energy  $E_{conformal}$  vanishes (which legitimates its name). By fixing the area  $\int_{\mathbf{y}(U)} dA$  (e.g. by fixing its boundary), minimizing the Dirichlet energy is equivalent to a minimization of  $E_{conformal}$ .

To generalize the Dirichlet energy to discrete surfaces  $S$  represented as triangle mesh, one identifies the plane defined by each triangle  $T_S$  with  $\mathbb{R}^2$  by fixing an arbitrary orthonormal basis  $(\mathbf{t}_1, \mathbf{t}_2)$  in each triangle plane. We can then compute the Dirichlet energy for a parameterization  $\tilde{\mathbf{x}}$  by substituting  $\tilde{\mathbf{x}}^{-1}$  for  $\mathbf{y}$  and summing over all triangles.

As the partial derivatives of  $\tilde{\mathbf{x}}^{-1}$  are linear in the texture coordinates, the Dirichlet energy is a quadratic energy and can be efficiently minimized using linear solvers. Hormann showed in [Hor01], that harmonic maps fit into the edge-spring model and derived appropriate spring constants. However, in the discrete case, a parameterization minimizing the Dirichlet energy may contain flips. Moreover, the second term in Equation 4.1 favours parameterizations with a small domain area. To avoid the trivial solution that maps  $S$  onto a point, boundary conditions have to be imposed, e.g. by fixing boundary vertices, although Desbrun *et al.* [DMA02] propose a boundary condition that alleviates this problem.

In view of the area deformation potential that we will introduce in Section 5.1, we would like to point out, that Desbrun *et al.* also complemented the conformal energy with a *discrete authalic energy* to reduce area deformation. However, as shown in [Deg03] area deformation is a non linear expression in the texture coordinates, whereas the discrete authalic energy is quadratic. It thus measures not area deformation directly, but only local variations in the area element. While this local variations can be small, the overall global area deformation might still be high.

**Cauchy-Riemann equations:** For a function  $\mathbf{y} : U \subset \mathbb{R}^2 \rightarrow \mathbb{R}^2$  mapping between two planar domains, another way to characterize conformality are the *Cauchy-Riemann equations*:

$$\begin{aligned} y_{1,1} &= y_{2,2} \\ y_{1,2} &= -y_{2,1} \end{aligned}$$

These inspired Levy *et al.* [LPRM02c] to another definition of discrete conformality. Fixing an orthogonal basis in each triangle and setting  $\mathbf{y} = \tilde{\mathbf{x}}^{-1}$  as described

above, they applied these equations to surface parameterization  $\tilde{\mathbf{x}}$ . As, in contrast to the smooth case, no exact solution exists, they minimized the squared residual error and called the resulting parameterizations *Least Squares Conformal Maps (LSCM)*. The squared residual coincides with the energy  $E_{conformal}$  as defined in Equation 4.3. As with the Dirichlet energy, the conformal energy is quadratic in the texture coordinates and can be minimized by solving a sparse linear system. In contrast to the former, the conformal energy is invariant with respect to uniform scaling and thus has no preference for parameterizations with a small domain  $\bar{S}$  (see Equation 4.1) and thus needs not fixed boundary values. However, as the energy is invariant with respect to rigid transformations, at least two vertices have to be fixed to determine position, orientation and scale of the preimage  $\tilde{\mathbf{x}}^{-1}(S)$ . While theoretically any two vertices can be fixed, the choice of constraints significantly influences the robustness of the method (see [SLMB04]). Moreover, minimizing parameterizations may contain flips.

**MIPS Energy** The *MIPS energy* proposed by Hormann and Greiner [HG00] is based on the following observation: For a piecewise affine parameterization  $\tilde{\mathbf{x}}$ , the metric deformation within each triangle  $T_{\bar{S}}$  is described by the linear map  $\mathbf{A} = \nabla \tilde{\mathbf{x}}|_{T_{\bar{S}}}$  which can be decomposed using its singular value decomposition as

$$\mathbf{A} = \mathbf{U} \begin{pmatrix} \sigma_{max} & 0 \\ 0 & \sigma_{min} \\ 0 & 0 \end{pmatrix} \mathbf{V}^t$$

with two orthogonal matrices  $\mathbf{U}$  and  $\mathbf{V}$ . As orthogonal matrices,  $\mathbf{U}$  and  $\mathbf{V}$  do not cause any shape deformation (only rotation or reflection) the deformation of intrinsic shape is thus captured by the two remaining parameters  $\sigma_{max}$  and  $\sigma_{min}$ . (Please note, that since

$$\tilde{\mathbf{g}}|_{T_{\bar{S}}} = \mathbf{A}^t \mathbf{A} = \mathbf{V} \begin{pmatrix} \sigma_{max}^2 & 0 \\ 0 & \sigma_{min}^2 \end{pmatrix} \mathbf{V}^t$$

the singular values  $\sigma_{max}$  and  $\sigma_{min}$  are the square roots of the eigenvalues of  $\tilde{\mathbf{g}}$ .) Hormann and Greiner observed, that  $\tilde{\mathbf{x}}$  is conformal if and only if

$$\sigma_{max} = \sigma_{min}$$

(this follows easily from the above relation to the eigenvalues of  $\tilde{\mathbf{g}}$ ). To quantize conformality, they proposed the following potential

$$E_{mips} = \frac{\sigma_{max}}{\sigma_{min}} + \frac{\sigma_{min}}{\sigma_{max}}$$

The acronym MIPS is a bit misleading since it stands for Mostly Isometric Parameterizations. In fact, the MIPS energy attains its minimum for  $\sigma_{max} = \sigma_{min}$  and

thus for any conformal parameterization rather than only for isometries, which are characterized by  $\sigma_{max} = \sigma_{min} = 1$ . Similar to the LSCM energy  $E_{conformal}$ , the MIPS energy is invariant with respect to uniform scaling of texture coordinates and thus allows free boundary optimization. In contrast to the former, the MIPS energy is nonlinear energy and its optimization is more involved. However, if the optimization is initialized with an admissible set of texture coordinates, it provably converges to a minimum containing no flips.

**Angle Based Flattening:** Introduced by Sheffer and de Sturler in [SS01], *Angle Based Flattening* is directly formulated in the angles of triangles  $T_{\bar{S}}$ . The employed potential is the summed squared difference of corresponding angles in  $T_{\bar{S}}$  and  $T_S$ . While this potential is quadratic in the angles, non-linear constraints have to be imposed on the minimization, to ensure that the solution in fact corresponds to a planar triangle mesh. These constraints also guarantee the absence of flips. Once a solution is found, triangles are laid out in the plane progressively using the computed angles. The original ABF method is prone to numerical errors in the constrained optimizations which influence the robustness of the progressive triangle layout. This issue was addressed by Sheffer et al. in [SLMB04]. There a customized non-linear solver is proposed that not only enhances robustness but also gives significant speed up. The resulting algorithm coined *ABF++* seems to be the fastest conformal parameterization method that guarantees the absence of flips.

**Circle Packing:** Another characterization of conformal maps is based on circles and goes back to a conjecture of Thurston [Thu85]: As conformal maps are locally isometric, they map infinitesimal circles on circles. Parameterization approaches based on circle packing generalize this property to the discrete setting by replacing the infinitesimal circles by finite circles. To compute a discrete conformal parameterization, the surface  $S$  is first approximated with a circle packing, a set of circles with a prescribed tangency pattern. Once this packing is defined, a circle packing of the plane  $\bar{S}$  with the same tangency pattern is searched for. The approach of Hurdal et al. [HBS<sup>+</sup>99] place circles at vertices in such a way, that two circles are tangent if they are adjacent in  $\mathcal{M}$ . Results from the theory of circle packing then guarantee the existence of a unique circle packing in the plane for any set of radii specified at the boundary vertices. However, their method introduces unnecessary deformation, as planar triangulations are not necessarily mapped onto themselves. Therefore, other notions of adjacency have been suggested (e.g. [BH03, KSS06, YKL<sup>+</sup>08]). Common to all these approaches is a non-linear optimization.

**Metric Scaling and Curvature Flow** For a smooth Riemannian manifold, the Gaussian curvatures  $\kappa$  and  $\bar{\kappa}$  associated with two Riemannian metrics  $g$  and



$\bar{\mathbf{g}} = e^{2u}\mathbf{g}$  are related by the following equation

$$\bar{\kappa} = e^{-2u}(\kappa - \Delta_{\mathbf{g}}u)$$

for which the uniformization theorem guarantees the existence of a solution. The function  $u$  is a smooth metric scaling function defined on the surface and is related to the conformal factor  $\lambda$  by  $\lambda = e^{2u}$ . For the case of flat metric  $\bar{\mathbf{g}}$  with  $\bar{\kappa} = 0$ , the equation simplifies to

$$\kappa = \Delta_{\mathbf{g}}u$$

where  $\Delta_{\mathbf{g}}$  denotes the surface Laplacian.

Recently, another class of conformal parameterization methods has evolved, that solve for a discrete notion of this metric scaling function  $u$  rather than solving for texture coordinates directly. To this extent, Ben-Chen et al. [BCGB08] discretize the above Poisson problem using the popular cotangent discretization of the Laplacian and interpolate the scaling function  $u$  to the edges. However, scaling edges using these values gives only a rough approximation. Ben-Chen et al. use this approximation to optimize the placement of singularities. The actual parameterization is then done using the method from [SLMB04]. Springborn et al. [SSP08] define a discrete notion of Riemannian metric and replace the above Poisson's equation by the Euler-Lagrange equations of an appropriately defined convex energy. A different but related approach is taken by Jin et al. [JKG08], who compute the scaled metric  $\bar{\kappa}$  as stationary point of a discrete version of the Ricci flow. The Ricci flow evolves the metric tensor to a state of constant Gaussian curvature. Concerning planar parameterization, approaches based on metric scaling are, similar to circle packing, rather interesting from a theoretical point of view than having practical advantages for the actual computation. However, they can be used to obtain a quick estimate of the conformal factor which can be valuable for placing singularities or cuts as e.g. done in [BCGB08].

## Geodesic Distances

In Euclidean spaces all geometric quantities as angles and areas can be expressed in terms of distances between points. This observation motivated another class of approaches, that express parametric shape deformation by measuring the changes in distance caused by a parameterization  $\tilde{\mathbf{x}}$ . More precisely, these approaches relate Euclidean distances in  $\bar{S}$  to geodesic distances on the surface  $S$ .

In [ZKK01], Zigelman et al. built up a distance matrix by measuring the geodesic distances between all pairs of vertices on the surface. They then search for a configuration of texture coordinates in  $\bar{S}$ , that exhibit the same pairwise Euclidean distances. As perfect distance preservation is only possible for developable surfaces, the squared difference in distance is summed over all pairs and



minimized using *Multi Dimensional Scaling (MDS)* a statistical method closely related to principal component analysis. The method successfully extrapolates boundary texture coordinates but unfortunately MDS does not allow to specify further constraints as e.g. face orientation. In fact, flips are highly likely to occur if the surface is not developable.

Sheffer and de Sturler propose in [Sds02] a post processing for e.g. conformal parameterization, that reduces length deformation. Their method iteratively smoothes an overlay grid in  $\bar{S}$  to minimized length distortion on edges of the triangle mesh. The original texture coordinates are then warped according to the deformation of the grid. Sheffer and de Sturler proved that their method does not introduce any flips. However, it is not clear how the smoothing influences metric properties of the parameterization, as e.g. conformality.

One of the earliest approaches to parameterization in computer graphics was proposed by Maillot et al. [MYV93]. They suggest to sum squared differences in lengths of edges in  $\bar{S}$  and  $S$  as one part of their potential:

$$E_{edge-length} := \sum_{edges (v,w) \in \mathcal{M}} \frac{(\|\mathbf{x}^v - \mathbf{x}^w\|^2 - \|\bar{\mathbf{x}}^v - \bar{\mathbf{x}}^w\|^2)^2}{\|\mathbf{x}^v - \mathbf{x}^w\|^2} \quad (4.4)$$

The resulting potential is non-linear and does not guarantee the absence of flips. Nevertheless, similar potentials have been used afterwards in several approaches in computer aided design to unfold nearly developable surface (see Section 7.2.1).

### Surface Sampling

While not directly expressed in terms of geometric properties, *surface sampling potentials* also capture certain properties of the metric. These potentials aim at parameterizations that provide an optimal sampling of the surface. While in the Euclidean space  $\bar{S}$  a uniform grid represents a natural sampling, there is no natural sampling on surfaces. To store signals on the surface as e.g. colors, normals, etc. these are therefore usually mapped into  $\bar{S}$  using the inverse parameterization  $\tilde{\mathbf{x}}^{-1}$  and sampled on a uniform grid. The parameterization thus determines the way surface signals are sampled.

The question whether a parameterization provides a good sampling is application specific. If the signal is known and fixed, “good” means that the sample density adapts the signal frequency. Potential for this case are not discussed here (see overview in the introduction to this section), since this quality is rather independent of the surface metric. If, in contrast, the signal is not known in advance, high-frequency content is equally likely to appear anywhere on the surface and it is reasonable to assume a constant distribution of signal frequency over the whole surface and in all directions.

In an attempt to reduce the probability of under sampling, Sander *et al.* [SSGH01b] considered in this situation the stretch imposed on a small line segment in the domain by the parameterization  $\tilde{\mathbf{x}}$ . If its length in the domain is much smaller than the length of its image on the surface, undersampling becomes very likely. Obviously the probability for under sampling increases with stretch and therefore, to reduce undersampling, the stretch should be as small as possible.

Locally, the largest and smallest stretch imposed by the parameterization are captured by the singular values  $\sigma_{max}$  and  $\sigma_{min}$  of the derivative  $\nabla\tilde{\mathbf{x}}$ . As discussed in Section 4.2.2, these values correspond to the roots of the eigenvalues  $\lambda_{max}$  and  $\lambda_{min}$  of the metric tensor. This observation led Sander *et al.* to define two *geometric stretch* energies as follows:

$$E_{stretch-inf}[\tilde{\mathbf{x}}] = \max_{T \in \mathcal{M}} (\lambda_{max}[\tilde{\mathbf{x}}|_{T_S}]) \quad (4.5)$$

and

$$E_{stretch-2}[\tilde{\mathbf{x}}] = \frac{1}{2} \sum_{T \in \mathcal{M}} (\lambda_{min}[\tilde{\mathbf{x}}|_{T_S}] + \lambda_{max}[\tilde{\mathbf{x}}|_{T_S}]) |T_S|$$

While the first energy measures the maximal stretch imposed on a line segment in the domain, the second energy is an area weighted average stretch on the mesh and has the important advantage that it is smooth in the texture coordinates and hence much easier to minimize. Both energies are nonlinear but can be optimized in a way that ensures the absence of flips. Sander *et al.* proposed in [SGSH02] a simple modification that allows natural boundary conditions. Surprisingly, minimizing the geometric stretch energies does not necessarily result in an even distribution of samples on the surface. In figure 4.2, a parameterization is shown for a non-developable surface that was obtained using the method from [SGSH02]. The geometric stretch in the gap marked in blue is much smaller than the average stretch and thus the image of the gap on the surface is heavily oversampled, which is not penalized by the energy  $E_{stretch-2}$  but, on the contrary, honored.

Samplings as provided by the method of Sander *et al.* are well suited for the above sketched application of storing an unspecified surface signal. Other applications, as e.g. remeshing, surface fitting or 3d painting systems, however, require, besides an adequate sampling rate, a mostly uniform and isotropic sampling pattern (see e.g. [KL96, Hor01]). Sorkine *et al.* [SCOGL02] proposed a modification of the  $E_{stretch-inf}$  that penalizes both over- and under sampling

$$E_{uniform} := \max(\sigma_{max}, \frac{1}{\sigma_{min}}). \quad (4.6)$$

Since we have  $\sigma_{max} \geq \sigma_{min} \geq 0$  this potential attains its minimum for  $\sigma_{max} = \sigma_{min}$ , which corresponds to an isometric parameterization. The problem of this

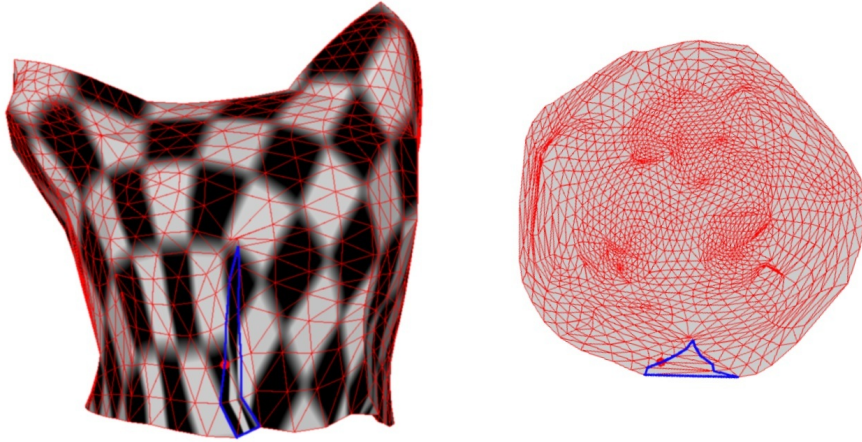


Figure 4.2: Minimizing  $E_{stretch-2}$  does not necessarily result in a uniform sampling: Left: Original surface  $S$  textured with a checker texture to visualize shape deformation. Right: The domain  $\bar{S} = \tilde{\mathbf{x}}^{-1}(S)$ . The stretch in the blue gap is much smaller than the average stretch.

energy is, that it is not differentiable which prevents an efficient minimization. The combined energy that we will propose in Section 5.4 takes for a special choice of parameters a similar form, but is smooth in the texture coordinates.

### Elastic Potential

In the introduction to this chapter we interpreted the parameterization problem as a special case of the general surface deformations. It therefore suggests itself to minimize the elastic potential to compute optimal parameterizations. As for parameterization, we assume in this context a flat original surface  $\bar{S}$  and thus we cannot expect the surface deformation to preserve curvature (except for the special case that the surface  $S$  is also flat). Consequently, it is sufficient to concentrate on the membrane potential and to omit the bending term in Equation 3.13.

The idea of using the elastic shell model for parameterization goes back to Maillot et al. [MYV93]. While in their article Maillot et al. sketched the above idea, they refrained from actually minimizing the elastic potential, which they claimed to be too costly and instead suggested the discrete edge length potential (Equation 4.4).

Unfortunately, the edge-length potential has not much in common with the elastic energy. If a triangle  $T_{\bar{S}}$  collapses, the edge length potential increases but remains bounded. However, it can be shown, that the the membrane potential

tends to infinity.<sup>1</sup> This property of the actual membrane potential is very valuable, as we will see in Section 5.5 that it helps to prevent flips in the optimization.

Using the actual membrane potential for parameterizations seems to be a natural and promising approach at first sight. However, there are some disadvantages: First, for most applications the parameterization should preserve geometric properties as angles, areas and length. The elastic potential has primarily a physical interpretation. It describes the physical work that is necessary to deform a planar domain  $\bar{S}$  into  $S$ . It is not clear how it is affected by distortion of geometric shape properties. It is also not obvious, how the material tensor should be chosen and how its choice affects the preservation of geometric properties.

Another problem are gap artifacts in the parameterization similar to the over-sampling artifacts for the method of Sander et al. described in the previous section. Figure 4.3 shows these artifacts on a results obtained by minimizing the membrane potential. For this result, we chose an isotropic material with a Poisson's ratio of zero as suggested by Maillot et al. which resembles a material like cork. (Please note, that at least the choice of the Poisson's ratio is somewhat arbitrary.) For this choice, the membrane potential's density in Equation 3.16 takes the form:

$$\begin{aligned} e_{membrane} &= \frac{Eh}{8} (k_\beta^\gamma k_\gamma^\beta - 2k_\alpha^\alpha + 2) \\ &= \frac{Eh}{8} (\lambda_{max}^2 + \lambda_{min}^2 - 2(\lambda_{max} + \lambda_{min}) + 2) \\ &= \frac{Eh}{8} ((\lambda_{max} - 1)^2 + (\lambda_{min} - 1)^2) . \end{aligned}$$

Remember, that in the shell model this density is integrated over the undeformed surface  $\bar{S}$ . For parameterization,  $\bar{S}$  is subject to the optimization while we consider  $S$  as fixed. It is therefore convenient to consider the density of the elastic energy with respect to  $S$  which is obtained by dividing by the area element  $d\bar{S} = \sqrt{\det(\tilde{\mathbf{g}}^{-1})}dS = 1/\sqrt{\lambda_{max}\lambda_{min}}dS$  as

$$\hat{e}_{membrane} = \frac{Eh}{8} ((\lambda_{max} - 1)^2 + (\lambda_{min} - 1)^2) / \sqrt{\lambda_{max}\lambda_{min}} .$$

While the artifacts shown in 4.3 resemble those observed for the sampling energy of Sander et al., it becomes apparent by writing the membrane potential in this form, that the elastic membrane potential penalized both over- and undersampling as it tends to infinity for vanishing eigenvalues as well as for large eigenvalues. In [Deg03] we show, that such gap artifacts are related to missing convexity of the

<sup>1</sup>As the area  $|T_{\bar{S}}|$  collapses, the maximal eigenvalue of  $\tilde{\mathbf{g}}$  grows at least linear with  $1/|T_{\bar{S}}|$ . From Equations 3.17 and 3.16 it then follows that the density  $e_{membrane}$  grows with  $1/|T_{\bar{S}}|^2$ . The integral of  $e_{membrane}$  over  $T_{\bar{S}}$  thus tends to infinity.

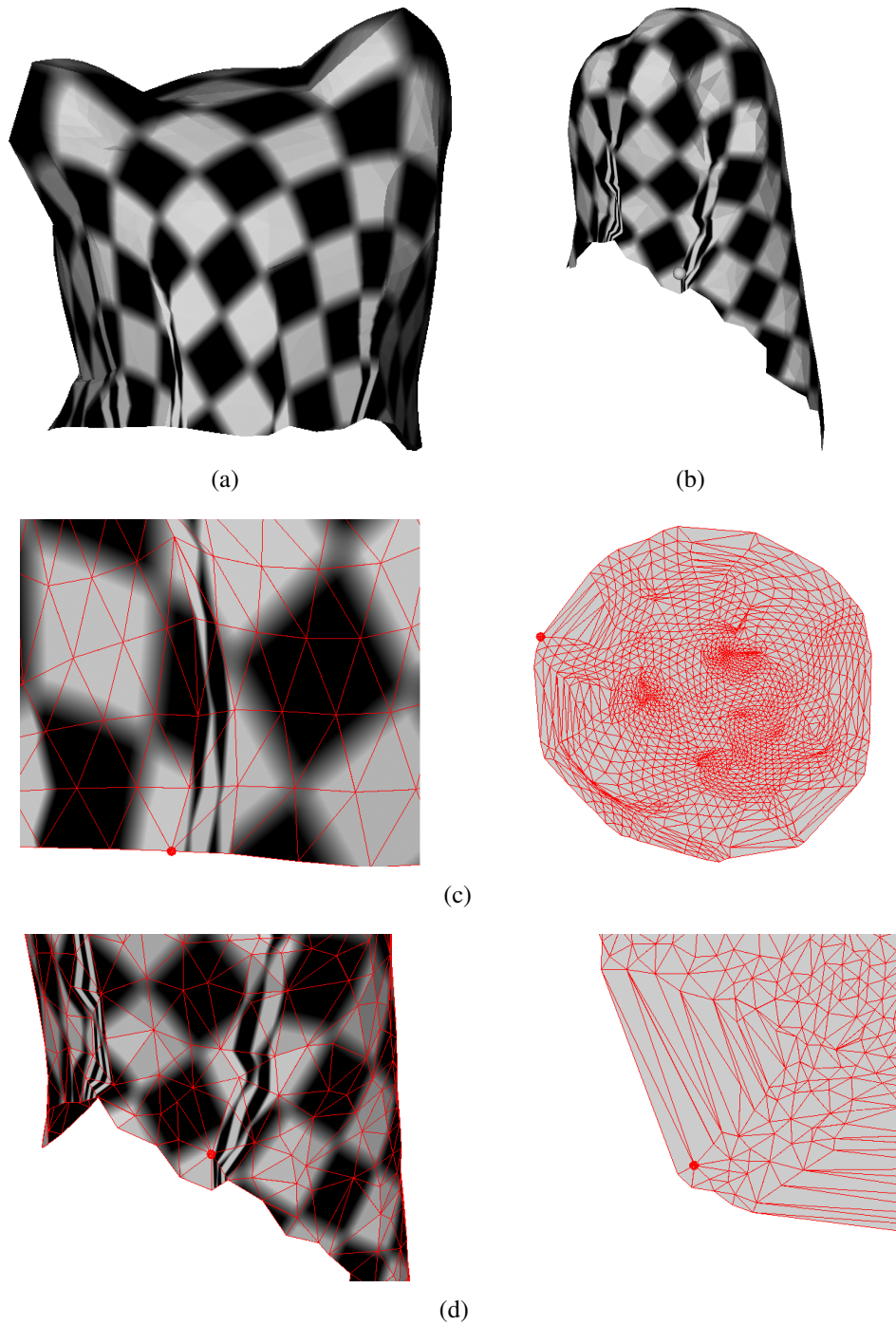


Figure 4.3: Results obtained by minimizing the elastic membrane potential: (a) the back of the cat head model (b) a cylindrical shape (c) a closeup of a gap in cat head (left) and the corresponding texture mesh (right) (d) a closeup of a gap in cylindrical shape (left) and the corresponding texture mesh (right)

potential density in the eigenvalues of  $\tilde{\mathbf{g}}^{-1}$ . This is in particular true for the elastic membrane potential.

We would like to remark, that despite these artifacts and the missing relation to geometric properties, we still see potential for parameterizations based on the shell model in certain application. One of these applications are material aware texture maps as will be discussed in Chapter 7.

---

## GEOMETRIC DEFORMATIONS POTENTIALS

---

Based on a profound mathematical theory, methods for conformal parameterizations have reached today a relatively advanced level. On the contrary only few methods take geometric properties other than angles into account, all of which come with certain drawbacks as was detailed in the previous chapter. The goal of this chapter is to find a potential for parameterizations with the following properties:

- It should relate directly to distortions of geometric shape properties. Besides angle deformation, it should also account for area and length distortion.
- It should not attain its minimum for discrete parameterizations containing flips.
- It should support natural boundary conditions, i.e. its optimization should result in parameterizations with minimal shape deformation in the absence of further constraints on the domain boundary.

For general surfaces, an isometric parameterization does not exist and a compromise between preservation of angles, areas and length must be made. To control this compromise, our approach is to search first for individual measures for distortion of each of these properties. These measures are then combined into a potential. A weighting parameter allows the user to specify the relative importance of angle, area and length preservation.

We start by noting, that the preservation of length, angles and areas are not independent from each other. As the eigenvalues  $\lambda_{\{max,min\}}$  correspond to maximal stretch and shrink, we must have  $\lambda_{max} = \lambda_{min} = 1$  for a parameterization that preserves length. It is thus already isometric and preserves also area and angles. Length preservation is thus the strongest property equivalent to isometry and implies angle and area preservation.

On the other hand, conformal maps preserve angles but not necessarily areas nor length. Likewise, a map that preserves area must not necessarily preserve



angles and length (a simple example is shearing). Area and angle are thus independent of each other in the sense, that a parameterization can preserve one of these properties without preserving the other. However, we will see in the Section 5.3 that taken together, angle and area preservation imply isometry and thus length preservation. We sum up these relations between the three metric properties as the following pseudo-formula:

$$\text{Length Preservation} = \text{Angle Preservation} \dagger \text{Area Preservation}$$

where  $\dagger$  denotes a sum of independent components. Therefore, it seems reasonable to find measures for angle and area distortion (Sections 5.1 and 5.2) and then to express length deformation in terms of these (Section 5.3). Section 5.4 combines both measures in a parameterization potential using a relative angle/area importance weighting parameter. Finally, we show in Section 5.5 that the combined potential has in fact the desired properties.

## 5.1 Area

To support an efficient optimization, it would be highly desirable to find an area distortion measure that is a quadratic function in the texture coordinates. Together with the conformal energy used by Levy *et al.* in [LPRM02c] it would then be possible to derive a linear method that captures angle and area deformations. Unfortunately, it was shown in [Deg03] that such a quadratic area deformation measure does not exist.

To find the change in area of shapes caused by the parameterization  $\tilde{\mathbf{x}}$  we consider an infinitesimal area element  $d\tilde{S}$  that is mapped by  $\tilde{\mathbf{x}}$  onto an area element  $dS$  on the surface  $S$ . According to Equation 2.5, the relation between the area of these infinitesimal shapes is then given as

$$dS = \sqrt{\det(\tilde{\mathbf{g}}^{-1}\mathbf{g})}d\tilde{S} = \sqrt{\det(\tilde{\mathbf{g}})}d\tilde{S}. \quad (5.1)$$

The factor  $\sqrt{\det(\tilde{\mathbf{g}})}$  thus gives the infinitesimal increase in each point and the parameterization is area preserving if and only if for every point  $\tilde{\mathbf{p}} \in \tilde{S}$

$$\sqrt{\det(\tilde{\mathbf{g}}(\tilde{\mathbf{p}}))} = 1$$

holds. To find an appropriate area deformation potential, we now search for its density function  $e_{area} : S \rightarrow \mathbb{R}$  and integrate it over  $S$ :

$$E_{area}[\tilde{\mathbf{x}}] := \int_S e_{area} dS$$



The density  $e_{area}$  itself is chosen as a function of the area increase, i.e.

$$e_{area} = f \circ \sqrt{\det \tilde{\mathbf{g}}} \circ \tilde{\mathbf{x}}^{-1}$$

where  $f : \mathbb{R} \rightarrow \mathbb{R}$  is an objective function that will be defined now: Obviously, the objective function  $f = f(t)$  should be minimal for  $t = 1$ . Furthermore, we require that it has no other local extrema, which could complicate the optimization. As we will see in section 5.5.2 triangle flips can be prevented in the optimization if the potential tends to infinity whenever a texture triangle  $T_{\bar{S}}$  degenerates, that is if its area  $|T_{\bar{S}}|$  tends to zero. For the affine map  $\tilde{\mathbf{x}}|_{T_{\bar{S}}}$  partial derivatives and metric tensor are constant and thus

$$|T_S| = \int_{T_S} dS \quad (5.2)$$

$$= \int_{T_{\bar{S}}} \sqrt{\det(\tilde{\mathbf{g}})} d\bar{S} \quad (5.3)$$

$$= \sqrt{\det(\tilde{\mathbf{g}})} \int_{T_{\bar{S}}} d\bar{S} \quad (5.4)$$

$$= \sqrt{\det(\tilde{\mathbf{g}})} |T_{\bar{S}}| \quad (5.5)$$

If  $T_{\bar{S}}$  tends to zero,  $\sqrt{\det(\tilde{\mathbf{g}})} = |T_S|/|T_{\bar{S}}|$  tends to infinity. The objective function  $f(t)$  should thus tend to infinity as  $t \rightarrow \infty$ .

In order to prevent gap artifacts in the texture mesh similar considerations as for the elastic energy (Section 4.2.2) hint, that the objective function  $f$  should be convex in the eigenvalues of  $\tilde{\mathbf{g}}^{-1}$ . In fact, it was shown in [Deg03] that convexity of  $f(1/t)$  is necessary to prevent gap artifacts.

A simple choice for the objective function  $f$  that satisfies all three requirements is  $f(t) = t + 1/t$  and our sought-after density function becomes

$$e_{area} = \sqrt{\det(\tilde{\mathbf{g}})} + \frac{1}{\sqrt{\det(\tilde{\mathbf{g}})}} \quad (5.6)$$

Later, it will be convenient to express  $e_{area}$  in terms of the singular values  $\sigma_{max}$  and  $\sigma_{min}$  of  $\nabla \tilde{\mathbf{x}}$ . As these are the roots of the eigenvalues  $\lambda_{\{max,min\}}$  of the metric tensor, we have  $\sqrt{\det(\tilde{\mathbf{g}})} = \sqrt{\lambda_{max}\lambda_{min}} = \sigma_{max}\sigma_{min}$  and thus

$$e_{area} = \sigma_{max}\sigma_{min} + \frac{1}{\sigma_{max}\sigma_{min}} \quad (5.7)$$

## 5.2 Angle

As shown in section 4.2.2, the conformality of a parameterization is characterized by

$$\sigma_{max}(\bar{\mathbf{p}}) = \sigma_{min}(\bar{\mathbf{p}}) \quad \Leftrightarrow \quad \frac{\sigma_{max}(\bar{\mathbf{p}})}{\sigma_{min}(\bar{\mathbf{p}})} = 1$$

in each point  $\bar{\mathbf{p}} \in \bar{S}$ . Taking the same approach as for the area potential, we first define a density function  $e_{angle} : S \rightarrow \mathbb{R}$ , which measures the angle deformation of the parameterization in each point on the surface and integrate over  $S$  to find the angle deformation measure:

$$E_{angle}[\tilde{\mathbf{x}}] := \int_S e_{angle} dS$$

Similarly to the derivation of the area deformation function, we choose the angle deformation function as

$$e_{angle} = f\left(\frac{\sigma_{max}}{\sigma_{min}}\right)$$

with an objective function  $f = f(t)$  that has a unique minimum for  $t = 1$ .

In fact, using the same objective function  $f(t) = t + 1/t$  as in the last section yields

$$e_{angle}(x) = \frac{\sigma_{max}}{\sigma_{min}} + \frac{\sigma_{min}}{\sigma_{max}} \quad (5.8)$$

which is nothing but the MIPS energy.<sup>1</sup>

Certainly other conformal potentials can be used to measure angle deformation. However, the MIPS energy can handle boundaries and prevents flips and although there are faster linear methods like LSCM, a quadratic angle potential is not useful in our context. As we are going to combine the angle and area potentials, a nonlinear optimization has to be used in any way.

For later reference we rewrite the density  $e_{angle}$  in terms of determinant  $\det \tilde{\mathbf{g}}$  and trace  $\text{tr} \tilde{\mathbf{g}}$  of the metric tensor:

$$e_{angle}(x) = \frac{\sigma_{max}}{\sigma_{min}} + \frac{\sigma_{min}}{\sigma_{max}} = \frac{\sigma_{max}^2 + \sigma_{min}^2}{\sigma_{max}\sigma_{min}} = \frac{\text{tr} \tilde{\mathbf{g}}}{\sqrt{\det \tilde{\mathbf{g}}}}$$

<sup>1</sup>Actually, integrating  $e_{angle}$  over the triangle surface  $S$  leads to an area weighted sum of the MIPS energy on each triangle. In contrast, the original MIPS energy sums the constant values of the above density on each triangle and omits area weighting. However, the additional area weighting occurs naturally and seems justified, as angle distortion on large surface triangle is visually more noticeable in e.g. texture mapping.

## 5.3 Length

In the introduction to this chapter we stated that a parameterization  $\tilde{\mathbf{x}}$  that is both conformal and area preserving is already an isometry. In fact, we have for such a parameterization

$$\frac{\sigma_{max}}{\sigma_{min}} = 1 \quad \text{and} \quad \sqrt{\det \tilde{\mathbf{g}}} = \sigma_{max}\sigma_{min} = 1 \quad \Leftrightarrow \quad (5.9)$$

$$\frac{\lambda_{max}}{\lambda_{min}} = 1 \quad \text{and} \quad \lambda_{max}\lambda_{min} = 1 \quad \Leftrightarrow \quad (5.10)$$

$$\lambda_{max} = \lambda_{min} = 1 \quad \Leftrightarrow \quad (5.11)$$

$$\tilde{\mathbf{g}} = \text{id} \quad (5.12)$$

that is  $\tilde{\mathbf{x}}$  is an isometry. From this observation, we conjectured in the introduction that deformation imposed by a parameterization can be expressed in terms of angle and area deformation. Multiplying the deformation functions for angle and area, derived in the last sections we get

$$e_{length} := e_{angle} \cdot e_{area} \quad (5.13)$$

$$= f\left(\frac{\sigma_{max}}{\sigma_{min}}\right) \cdot f(\sigma_{max}\sigma_{min}) \quad (5.14)$$

It should be explained now, that this density function does indeed measure the length distortion. First, as  $f(t)$  has only a single minimum for  $t = 1$ ,  $e_{length}$  assumes its global minimum if and only if  $\frac{\sigma_{max}}{\sigma_{min}} = 1$  and  $\sigma_{max}\sigma_{min} = 1$  hold simultaneously, i.e. for an isometric parameterization. While we certainly expect a length deformation measure to be minimal for isometric maps, it is far more interesting, how the deformation function generally captures length distortion and especially on non-developable surfaces. To this extent we rewrite the function as

$$e_{length} = f\left(\frac{\sigma_{max}}{\sigma_{min}}\right) \cdot f(\sigma_{max}\sigma_{min}) \quad (5.15)$$

$$= \left(\frac{\sigma_{max}}{\sigma_{min}} + \frac{\sigma_{min}}{\sigma_{max}}\right) \left(\sigma_{max}\sigma_{min} + \frac{1}{\sigma_{max}\sigma_{min}}\right) \quad (5.16)$$

$$= \sigma_{max}^2 + \frac{1}{\sigma_{max}^2} + \sigma_{min}^2 + \frac{1}{\sigma_{min}^2} \quad (5.17)$$

$$= f(\lambda_{max}) + f(\lambda_{min}). \quad (5.18)$$

In this form, it can be seen that  $e_{length}$  measures how much the eigenvalues  $\lambda_{max}$  and  $\lambda_{min}$  differ from one by the very objective function  $f$  that was used in the definition of  $e_{area}$  and  $e_{angle}$ . As the eigenvalues correspond to largest and smallest change in the length  $e_{length}$  does indeed capture length distortion.

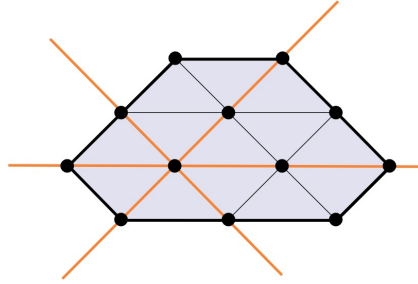


Figure 5.1: The area deformation potential  $e_{angle}$  is invariant with respect to shearing. Moving e.g. all vertices on the orange line along the direction of the line does not alter triangle areas and thus does not change  $e_{angle}$ .

From this density function we define again a length potential by integration over the surface:

$$E_{length} := \int_S e_{length} dS$$

Again, in view of the following discretization, we also give an expression for  $e_{length}$  in terms of determinant  $\det \tilde{\mathbf{g}}$  and trace  $\text{tr } \tilde{\mathbf{g}}$  of the metric tensor :

$$e_{length} = e_{angle} \cdot e_{area} \quad (5.19)$$

$$= \frac{\text{tr } \tilde{\mathbf{g}}}{(\det \tilde{\mathbf{g}})^{\frac{1}{2}}} \left( (\det \tilde{\mathbf{g}})^{\frac{1}{2}} + \frac{1}{(\det \tilde{\mathbf{g}})^{\frac{1}{2}}} \right) \quad (5.20)$$

$$= \text{tr } \tilde{\mathbf{g}} \left( 1 + \frac{1}{\det \tilde{\mathbf{g}}} \right) \quad (5.21)$$

## 5.4 An adaptable Potential

In the last section we multiplied the area and angle deformation functions and obtained a length deformation measure. It seems to be straightforward to generalize this approach by introducing an additional weighting

$$e_{combined} := e_{area}^{\alpha_{area}} \cdot e_{angle}^{\alpha_{angle}}$$

with two factors  $\alpha_{\{area, angle\}} \in [0, 1]$ . These factors determine the relative importance of angle and area preservation, giving the used the opportunity to influence the unavoidable tradeoff between angle and area preservation on non-developable surfaces. By using this density function we are able define a family of distortion measures, that range from pure angle distortion over length distortion to area distortion.

Unfortunately, minimizing the area deformation alone causes problems, which are due to the invariance of the area under shearing. An example is shown in Figure 5.1: Moving all vertices on one of the lines shown in orange only shears adjacent triangles and thus does not alter triangle areas. Due to this invariance, the potential  $E_{angle}$  has no unique minimum and optimizing the area distortion alone does not make much sense in the absence of further constraints. Another practical problem arises, as by shearing a triangle its angles can become arbitrary small. This causes numerical problems in the optimization.

Likewise,  $E_{angle}$  is invariant to rigid transformations of the texture coordinates as well as uniform scaling as these do not alter angles. Therefore, it assumes its minimum not only for a unique parameterization but for an affine subspace. However, minimizing angle distortion alone does not cause problems. It is sufficient to fix the texture coordinates of only two arbitrary vertices to remove the additional degrees of freedom. Unfortunately, this is not true for the area distortion as can be seen in Figure 5.1 where fixing any two vertices does not solve the problem.

Scaling both  $\alpha_{\{area,angle\}}$  only changes the values of the combined energy, but not the optimal parameterization. This leaves in fact only one degree of freedom. As a pure area optimization is not desirable whereas optimization of  $e_{angle}$  makes sense, we actually set

$$\alpha_{area} = \frac{\alpha}{\alpha + 1} \quad \text{and} \quad \alpha_{angle} = \frac{1}{\alpha + 1}$$

where  $\alpha$  is a single trade off parameter that takes values in  $[0, \infty[$ . The angle potential is hence obtained for  $\alpha = 0$ , the length potential for  $\alpha = 1$  and the area potential is only attained in the limit for  $\alpha = \infty$ .

In Figure 5.2 some plots of the combined potential's density as a function of the singular values  $\sigma_{min}$  and  $\sigma_{max}$  are shown: Since we always have  $\sigma_{max} > \sigma_{min}$ , only the values below the diagonal are actually attained. In (a) the pure angle deformation function is plotted. The minimum stretches out along the diagonal  $\frac{\sigma_{max}}{\sigma_{min}} = 1$ , which represents conformality. As  $e_{angle}$  increases with the ratio  $\frac{\sigma_{max}}{\sigma_{min}}$  the contour lines take the shape of lines with different steepness. In (b) the pure area deformation is plotted. The contour lines are given by  $\sigma_{min} = \frac{c}{\sigma_{max}}$  and thus take the shape of hyperbolic functions. The remaining plots show the combined deformation functions for increasing values of  $\alpha$ . Note that for  $\alpha \in ]0, \infty[$  the density function always has a unique global minimum in  $(1, 1)$  as in this point both  $e_{angle}$  and  $e_{area}$  assume their (unique) minimum.

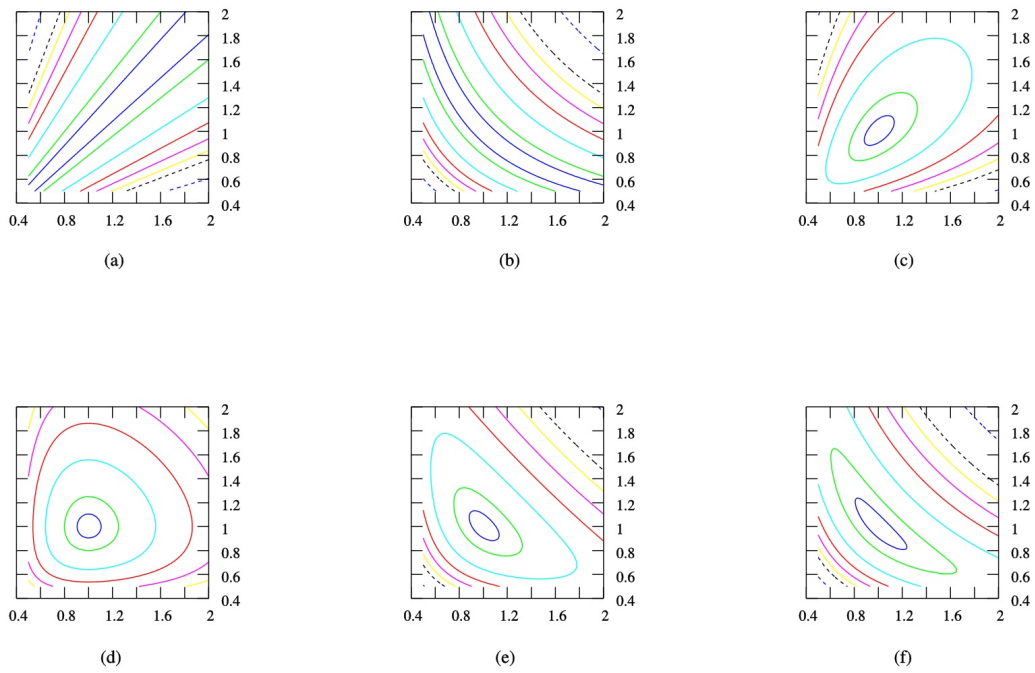


Figure 5.2: Plots of the combined density as a function of the singular values  $\sigma_{max}$  and  $\sigma_{min}$ . (a) pure angle deformation ( $\alpha = 0$ ) (b) pure area deformation ( $\alpha = \infty$ ) (c-f) plots are shown for  $\alpha = 0.25, 1, 4, 16$  respectively.

## 5.5 Properties

In the last section, we already mentioned that  $E_{area}$  is not quadratic in the texture coordinates and so is neither  $E_{length}$  and  $E_{combined}$ . We also remarked that all potentials are minimized by an isometric parameterization if such a parameterization exists. In this section we would like to show a few other basic properties of the potentials  $E_{angle}$  and  $E_{area}$  derived in the last section. The length deformation potential and the combined potential expressed in terms of  $E_{angle}$  and  $E_{area}$  inherit all of these properties.

### 5.5.1 Invariance with Respect to Rigid Transformation

All potentials derived in the last section are invariant under a rigid transformation  $\psi : \mathbb{R}^2 \rightarrow \mathbb{R}^2$  of the texture coordinates  $\bar{\mathbf{x}}^v$ . By rigid transformation we denote affine maps of the form  $\psi(\bar{\mathbf{p}}) = \mathbf{R}\bar{\mathbf{p}} + \mathbf{t}$ , where  $\mathbf{t} \in \mathbb{R}^2$  is a translation vector and  $\mathbf{R}$  is an orthogonal matrix. To see this, we set  $\tilde{\mathbf{x}}' = \tilde{\mathbf{x}} \circ \psi$  and compute its Jacobian

$$\nabla \tilde{\mathbf{x}}'(\psi^{-1}(\bar{\mathbf{p}})) = \nabla \tilde{\mathbf{x}}(\bar{\mathbf{p}}) \nabla \psi(\psi^{-1}(\bar{\mathbf{p}})) = \nabla \tilde{\mathbf{x}}(\bar{\mathbf{p}}) \mathbf{R}.$$

If  $\nabla \tilde{\mathbf{x}} = \mathbf{U}\Sigma\mathbf{V}^t$  is a singular value decomposition for  $\nabla \tilde{\mathbf{x}}$ , then  $\nabla \tilde{\mathbf{x}}' = \mathbf{U}\Sigma(\mathbf{V}^t\mathbf{R})$  is a singular value decomposition for  $\nabla \tilde{\mathbf{x}}'$  as  $\mathbf{R}$  is orthogonal. Thus, the singular values of  $\nabla \tilde{\mathbf{x}}'(\psi^{-1}(\bar{\mathbf{p}}))$  are those of  $\nabla \tilde{\mathbf{x}}(\bar{\mathbf{p}})$ . Since the density  $e_{area}$  is defined solely in terms of the singular values, we have

$$e_{area}[\tilde{\mathbf{x}}'](\mathbf{p}) = e_{area}[\tilde{\mathbf{x}}](\mathbf{p})$$

for  $\mathbf{p} = \tilde{\mathbf{x}}(\bar{\mathbf{p}}) = \tilde{\mathbf{x}}'(\psi^{-1}(\bar{\mathbf{p}}))$  and thus  $E_{area}[\tilde{\mathbf{x}}'] = E_{area}[\tilde{\mathbf{x}}]$ . The same argument holds for the angle, length and combined potentials which are all expressed in terms of the singular values.

As metric potentials, these are also invariant with respect to rigid transformations  $\phi(\mathbf{p}) = \mathbf{R}'\mathbf{p} + \mathbf{t}'$  of the surface  $S$  itself. This is due to the fact that the metric tensor does not change under such deformations. Denoting the metric tensor of  $\phi \circ \tilde{\mathbf{x}}$  by  $\tilde{\mathbf{g}}'$  we have

$$g'_{\alpha\beta} = \phi_{i,j} \tilde{x}_{j,\alpha} \phi_{i,k} \tilde{x}_{k,\beta} = \mathbf{R}_{ij} \mathbf{R}_{ik} \tilde{x}_{j,\alpha} \tilde{x}_{k,\beta} = \delta_{jk} \tilde{x}_{j,\alpha} \tilde{x}_{k,\beta} = g_{\alpha\beta}$$

The density functions  $e_{\{angle,area,length,combined\}}$  are therefore invariant under such transformations. As the area element  $dS = \sqrt{\det \tilde{\mathbf{g}}} d\tilde{S}$  is also defined in terms of the metric tensor, the potentials  $E_{\{angle,area,length,combined\}}$  are also invariant.

As a consequence of the invariance of the potentials with respect to rigid transformations  $\psi$  of the domain  $\tilde{S}$ , additional constraints have to be imposed in order to have a unique minimum. Similar to the LSCM method discussed in Section 4.2.2, it is sufficient to constrain e.g. the texture coordinates of two vertices. The optimization described in the next chapter, however, does not necessarily need such constraints. In absence of constraints it converges to an arbitrary minimum.

### 5.5.2 Infinity on the Domain Boundary

To prevent flips in the optimization, we follow a similar approach that was previously also taken by Hormann [Hor01] and Sander *et al.* [SSGH01b]. For a mesh  $\mathcal{M} = (V, \mathcal{T})$ , we restrict the optimization to the set

$$\mathcal{U}_{valid} = \{(\bar{\mathbf{x}}^v)_{v \in V} \in \mathbb{R}^{|V| \times 2} \mid \forall T \in \mathcal{T} : T_{\bar{S}} \text{ is counterclockwise oriented and } |T_{\bar{S}}| \neq 0\}$$

of valid texture coordinates, i.e. texture coordinates, that contain no face flips and degenerated triangles. We now require that the potentials tend to infinity as the set of texture coordinated  $(\bar{\mathbf{x}}^v)_{v \in V} \in \mathcal{U}_{valid}$  approaches the boundary  $\delta\mathcal{U}_{valid}$ . Given this property, a gradient descent optimization starting from a configuration in  $\mathcal{U}_{valid}$  will not leave this set and thus again result in a valid parameterization.

To show that all the potentials defined in the last Section fulfill this requirement, we first remark that the deformation measures tend to infinity as the set of texture coordinates tends to a configuration that contains a degenerated triangle  $T_{\bar{S}}$ , i.e. a triangle with  $|T_{\bar{S}}| = 0$ . From Equation 5.5 we get

$$\frac{|T_S|}{|T_{\bar{S}}|} = \sqrt{\det(\tilde{\mathbf{g}})} = \sigma_{max}\sigma_{min}.$$

As the left hand side tends to infinity as  $|T_{\bar{S}}|$  vanishes, so must at least one of the singular values. From the definitions of  $e_{area}$  (Eq. 5.7) and  $e_{angle}$  (Eq. 5.8) it then follows that both density functions tend to infinity on the triangle  $T_S$  and so must the corresponding potentials.

Denoting the set of all texture coordinate sets that contain a degenerated triangle by  $\mathcal{U}_{degenerated}$ , it was shown in [Deg03] that

$$\delta\mathcal{U}_{valid} \subset \mathcal{U}_{degenerated}$$

and thus the potential tends to infinity as  $(\bar{\mathbf{x}}^v)_{v \in V} \in \mathcal{U}_{valid}$  approaches the boundary  $\delta\mathcal{U}_{valid}$  as required.

### 5.5.3 Smoothness and Derivatives

The existence of partial derivatives of the potentials with respect to texture coordinates is crucial for an efficient minimization. To show that our potentials are smooth functions of the texture coordinates, we consider a piecewise affine continuous map

$$\bar{\mathbf{y}} = \Psi_v \bar{\mathbf{y}}^v$$

defined via coordinates  $\bar{\mathbf{y}}^v \in \mathbb{R}^2$  at each vertex  $v \in V$ . Because of  $\tilde{\mathbf{x}} = \mathbf{x} \circ \bar{\mathbf{x}}^{-1}$  (see Figure 3.3) a change of the parameterization  $\bar{\mathbf{x}}$  gives rise to change of  $\tilde{\mathbf{x}}$  and



vice versa. We can therefore understand the potential  $E_{combined}$  as functions of  $\bar{\mathbf{x}}$  and write in a slight abuse on notation  $E_{combined} = E_{combined}[\bar{\mathbf{x}}]$ . We will now show, that the variation of the potential

$$\partial_{\bar{\mathbf{x}}} E_{combined}[\bar{\mathbf{x}}](\bar{\mathbf{y}}) = \left. \frac{\partial}{\partial \epsilon} E_{combined}[\mathbf{x} \circ (\bar{\mathbf{x}} + \epsilon \bar{\mathbf{y}})^{-1}] \right|_{\epsilon=0}$$

in direction of  $\bar{\mathbf{y}}$  exists and give explicit expression for the derivatives. The gradient of the potential with respect to coordinate  $j$  of vertex  $w$  is then obtained by choosing

$$\bar{\mathbf{y}}_i^v = \delta_{vw} \delta_{ij}$$

By Equations 5.6 and 5.9 the density functions  $e_{angle}$  and  $e_{area}$  are expressed as smooth function in terms of  $\text{tr } \tilde{\mathbf{g}}$  and  $\det \tilde{\mathbf{g}}$ . As products of powers of those, the combined and length density are also smooth in these terms. Now we have

$$\tilde{g}_{\alpha\beta} = \tilde{x}_{i,\alpha} \tilde{x}_{i,\beta} = x_{i,\gamma} \bar{x}_{\gamma,\alpha}^{-1} x_{i,\delta} \bar{x}_{\delta,\beta}^{-1} = g_{\gamma\delta} \bar{x}_{\gamma,\alpha}^{-1} \bar{x}_{\delta,\beta}^{-1}$$

and thus

$$\text{tr } \tilde{\mathbf{g}} = g_{\gamma\delta} \bar{x}_{\gamma,\alpha}^{-1} \bar{x}_{\delta,\alpha}^{-1} = g_{\gamma\delta} \bar{g}^{\gamma\delta} = \text{tr}(\mathbf{g} \bar{\mathbf{g}}^{-1}) \quad (5.22)$$

$$\det \tilde{\mathbf{g}} = \det \nabla \bar{\mathbf{x}}^{-1} \det \mathbf{g} \det \nabla \bar{\mathbf{x}}^{-1} = \det \mathbf{g} \det \bar{\mathbf{g}}^{-1} \quad (5.23)$$

The variation of the metric tensor  $\bar{\mathbf{g}}$  with respect to  $\bar{\mathbf{x}}$  is found as

$$\partial_{\bar{\mathbf{x}}} \bar{g}_{\alpha\beta}[\bar{\mathbf{x}}](\bar{\mathbf{y}}) = \left. \frac{\partial}{\partial \epsilon} ((\bar{\mathbf{x}} + \epsilon \bar{\mathbf{y}})_{i,\alpha} (\bar{\mathbf{x}} + \epsilon \bar{\mathbf{y}})_{i,\beta}) \right|_{\epsilon=0} = \bar{y}_{i,\alpha} \bar{x}_{i,\beta} + \bar{x}_{i,\alpha} \bar{y}_{i,\beta}.$$

From Equations 5.22-5.23 we then find the variations of  $\text{tr } \tilde{\mathbf{g}}$  and  $\det \tilde{\mathbf{g}}$  as

$$\partial_{\bar{\mathbf{x}}} \text{tr } \tilde{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}) = -\text{tr}(\mathbf{g} \bar{\mathbf{g}}^{-1} (\partial_{\bar{\mathbf{x}}} \bar{\mathbf{g}}) \bar{\mathbf{g}}^{-1}) \quad (5.24)$$

$$\partial_{\bar{\mathbf{x}}} \det \tilde{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}) = -\det \mathbf{g} \det \bar{\mathbf{g}}^{-1} \text{tr}((\partial_{\bar{\mathbf{x}}} \bar{\mathbf{g}}) \bar{\mathbf{g}}^{-1}) \quad (5.25)$$

where we have used the following identities for derivatives with respect to a matrix  $\mathbf{A} \in \mathbb{R}^2$  in direction of a matrix  $\mathbf{C} \in \mathbb{R}^2$ :

$$\begin{aligned} \partial_{\mathbf{A}} \text{tr } \mathbf{A}(\mathbf{C}) &= \text{tr } \mathbf{C} \\ \partial_{\mathbf{A}} \det \mathbf{A}(\mathbf{C}) &= \text{tr}(\mathbf{A}^{-t} \mathbf{C}) \det(\mathbf{A}) \\ \partial_{\mathbf{A}} \mathbf{A}^{-1}(\mathbf{C}) &= -\mathbf{A}^{-1} \mathbf{C} \mathbf{A}^{-1} \end{aligned}$$

The variation of the density functions are now found as

$$\begin{aligned}\partial_{\bar{x}}e_{area} &= \frac{1}{2} \left( 1 - \frac{1}{\det \tilde{\mathbf{g}}} \right) \frac{\partial_{\bar{x}} \det \tilde{\mathbf{g}}}{\sqrt{\det \tilde{\mathbf{g}}}} \\ \partial_{\bar{x}}e_{angle} &= \frac{1}{2\sqrt{\det \tilde{\mathbf{g}}}} \left( 2\partial_{\bar{x}} \operatorname{tr} \tilde{\mathbf{g}} - \frac{\operatorname{tr} \tilde{\mathbf{g}}}{\det \tilde{\mathbf{g}}} \partial_{\bar{x}} \det \tilde{\mathbf{g}} \right) \\ \partial_{\bar{x}}e_{length} &= \left( 1 + \frac{1}{\det \tilde{\mathbf{g}}} \right) \partial_{\bar{x}} \operatorname{tr} \tilde{\mathbf{g}} - \frac{\operatorname{tr} \tilde{\mathbf{g}}}{\det \tilde{\mathbf{g}}^2} \partial_{\bar{x}} \det \tilde{\mathbf{g}} \\ \partial_{\bar{x}}e_{combined} &= e_{area}^{\frac{-1}{\alpha+1}} e_{angle}^{\frac{1}{\alpha+1}} \partial_{\bar{x}}e_{area} + e_{area}^{\frac{\alpha}{\alpha+1}} e_{angle}^{\frac{-\alpha}{\alpha+1}} \partial_{\bar{x}}e_{angle}\end{aligned}$$

From these the variations of the potential are computed by integration over the surface  $S$  as described in Section 2.2.2.

---

## COMPUTING PARAMETERIZATIONS

---

In this chapter we will describe the implementation of a simple optimization algorithm for the proposed potentials. It features an easy implementation, ensures the absence of flips in the final parameterization and is yet efficient. Besides the potentials proposed in the last chapter, it can also be easily adapted to optimize most of the potentials reviewed in Chapter 5. We give a short description of the optimization algorithm in Section 6.1.1

Using a framework based on this minimization algorithm, we compute parameterizations for a set of surfaces as they typically appear in computer graphics applications. Section 6.2 presents a qualitative comparison of results obtained by optimizing different potentials. Moreover, we perform a quantitative evaluation that gives an assessment on geometric shape deformation and run times. This chapter concludes with a discussion of the experimental evaluation and relates our approach to more recent work in surface parameterization.

### 6.1 Optimization

All potential measuring metric deformation reviewed in Section 4.2.2 integrate potential density functions  $e$  that can be expressed in terms of the metric  $\tilde{\mathbf{g}}$  that is constant within each triangle. They thus can be discretized as described in Section 2.2.2. There we also gave an explicit expression for the metric tensor  $\mathbf{g}$  on a triangle. An analogous expression for  $\tilde{\mathbf{g}}$  can be obtained by substituting the coordinates  $\bar{\mathbf{x}}^v$  for  $\mathbf{u}^v$ . The gradient of the discrete potential with respect to texture coordinates  $\bar{\mathbf{x}}^v$  has been derived in Section 5.5.3 and can be discretized in the same way.

Two exceptions are the  $E_{stretch-inf}$  potential (Equation 4.5) and the uniform sampling potential  $E_{uniform}$  (Equation 4.6) that evaluate to the maximal value of the density instead of the surface integral. Although these can be optimized using a similar algorithm we do not discuss this case and refer to [Deg03] for details.

### 6.1.1 Simple Relaxation Optimization

Possible the most simple optimization algorithm for potentials of the above form is called *vertex relaxation*. It was already used in [SSGH01a] and [Hor01] to minimize the the  $E_{stretch-inf}$  and the MIPS potential. Vertex relaxation starts from a valid (i.e. one without flips) set of texture coordinates as for example computed by the Floater method [Flo01]. It iteratively picks a single vertex  $v \in V$  and optimizes the potential with respect to its texture coordinate  $\bar{x}^v$  only. As its texture coordinate only influences the parameterization  $\tilde{x}$  on adjacent triangles, it is sufficient to compute the densities  $e_T$  only for triangles  $T$  with  $v \in T$ . It is thus also possible to optimize vertices  $v, w \in V$  in parallel as long as they are not incident by an edge in  $\mathcal{M}$ .

The convergence rate of vertex relaxation is poor compared to a simultaneous optimization of all vertices and it is more prone to get stuck in local minima. On the other hand its simplicity allows for an easy implementation. But more important, the vertex relaxation gives more control over vertex flips. As shown in Section 5.5.2 an optimization of the combined potential  $E_{combined}$  and its special cases does not converge to a flipped or degenerated configuration of texture coordinates. However, we found in our experiments, that numerically a triangle  $T_{\bar{S}}$  can degenerate in cases when the corresponding surface triangle  $T_S$  is very narrow or also nearly degenerated. While such numerical problems can easily be intercepted in vertex relaxation, it is very difficult to avoid them in a simultaneous optimization of all vertices.

When detecting a degenerated triangle or flip, our implementation simply resets this vertex to a flipless position and proceeds with the optimization of other vertices until eventually the vertex becomes again selected for relaxation. Our current implementation uses a simple non-linear conjugate gradient optimization but restricts line searches to the kernel of the polygon spanned by edges between neighboring vertices. In this way line searches are initialized with reasonable bracketing intervals and become more robust with respect to flips. The property proven in Section 5.5.2 ensures that these bracketing intervals are valid and thus the restriction does not interfere with the optimization. As every relaxation reduces the overall potential and as the potential is positive (and thus bounded from below) the iterative relaxation terminates eventually.

The formulation of the potentials proposed in the last chapter and in particular its variation given in Section 5.5.3 relies on the existence of an auxiliary parameterization  $\mathbf{x}$ . On the other hand, vertex relaxation also needs a proper initialization. In the implementation of the relaxation step we set  $\mathbf{x}$  to the externally provided initialization and initialize  $\bar{x}$  to identity. The optimization will then optimize the parameterization  $\tilde{x} = \mathbf{x} \circ \bar{x}^{-1}$  by varying the texture coordinates  $(\bar{x}^v)_{v \in V}$ .

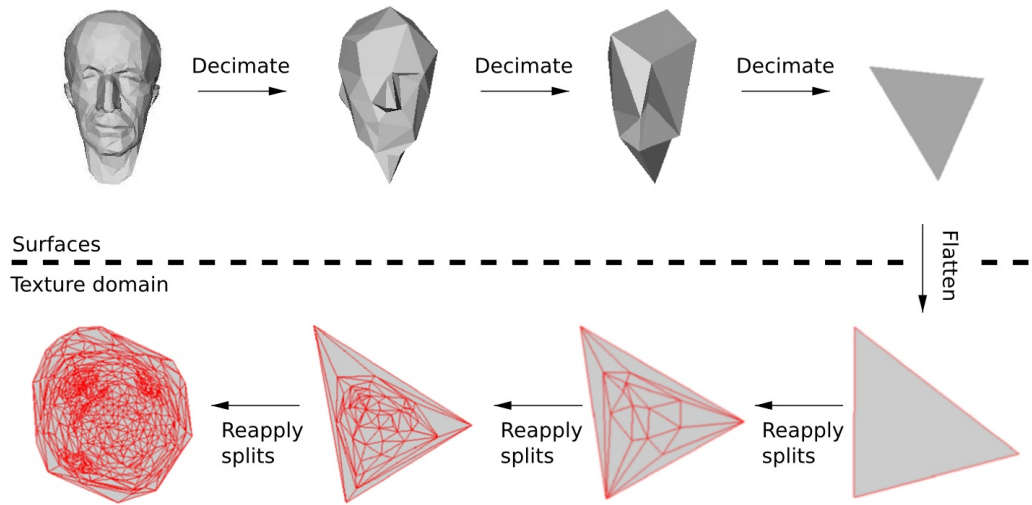


Figure 6.1: An overview over the hierarchical optimization method

### 6.1.2 Hierarchical Optimization

While vertex relaxation is simple, it has a poor convergence rate. Moreover, it must be initialized with a flipless set of texture coordinates. It is therefore usually combined with a hierarchical optimization, that is similar to multigrid methods on linear systems. Figure 6.1 gives an overview over the hierarchical optimization.

The hierarchical optimization proceeds in two phases: in the first phase a series of levels of detail (LOD) is generated from the input mesh. To this extent, a progressive mesh sequence [Hop96] is computed using half edge collapses. After each half edge collapse the vertices in the one ring of the remaining vertex are locked. Subsequent collapses are prohibited whenever the edge is incident with a locked vertex. If no more half edge collapses are possible, the splits are stored in a set, all vertices are unlocked and the decimation proceeds. In this manner, sets of independent splits are generated. This split independence will facilitate the placement of new vertices in the second phase. Since we assume a surface patch of disk-like topology, the last level of detail consists of a single triangle only. For more details on the LOD generation see e.g. [HGC99].

The second phase starts from the lowest level of detail — the so called base mesh — which is in our case a single triangle. The texture coordinates  $\bar{x}^v$  of its vertices are initialized to a congruent triangle in the plane centered in the origin.

In the following, the set of splits corresponding to the next level of detail is applied to the mesh and texture coordinates for new vertices are initialized to save positions, i.e. positions that do not cause flips. To this extent, the texture coordi-

nate  $\bar{x}^v$  of a newly inserted vertex  $v \in V$  can be positioned anywhere within the kernel of the polygon spanned by edges between neighboring vertices. It can be shown, that the kernel is non empty if the texture mesh was flipless before applying the split (see [Deg03]) and our implementation simply chooses its centroid. The independent set property ensures that no two newly inserted vertices are incident in the mesh. The shape deformation caused by the simple initial placement of these vertices can therefore not accumulate over several splits in the same set.

Once all splits in the set are applied, a valid initial parameterization has been established and the vertex relaxation described in the last section can be applied. Once it has converged, the hierarchical optimization proceeds with the next level of detail. As the surfaces of two subsequent levels of detail are similar, the solution of the coarser level provides a good initialization for the finer level and the relaxation converges after a few iteration. In our experiments we have found that the hierarchical method is usually much faster than pure vertex relaxation starting from a parameterization computed by non-metric methods such as the Floater method.

Since collapses may also appear at the boundary, the energy must be able to position boundary vertices in texture space during the optimization. Therefore, energies that require fixed boundary vertices like the sampling based potential  $E_{stretch-inf}$  cannot directly be used with the above hierarchical optimization method.

## 6.2 Experimental Evaluation

Using the optimization described in the last section we computed parameterizations for several surfaces as they typically appear in computer graphics application. The surfaces were digitized by laser scanning devices and postprocessed to remove noise and to reconstruct a manifold surface. In contrast to modeled surfaces that sometimes come with (user designed) texture coordinates, surfaces obtained by range scans come without any parameterization.

To pronounce differences between the parameterization potentials, the objects were chosen to be non-developable, having large areas of non zero Gaussian curvature. Moreover, they show a high surface area to boundary length ratio. Such surfaces are very challenging for potentials that aim at minimal metric deformation as in the planar domain  $\bar{S}$  a certain boundary length supports only a limited area. For these surfaces it is thus not possible to preserve both length of the boundary and surface area at the same time. Figure 6.2 shows parameterizations found as the minimum of the combined potential for  $\alpha = 1$  which coincides with

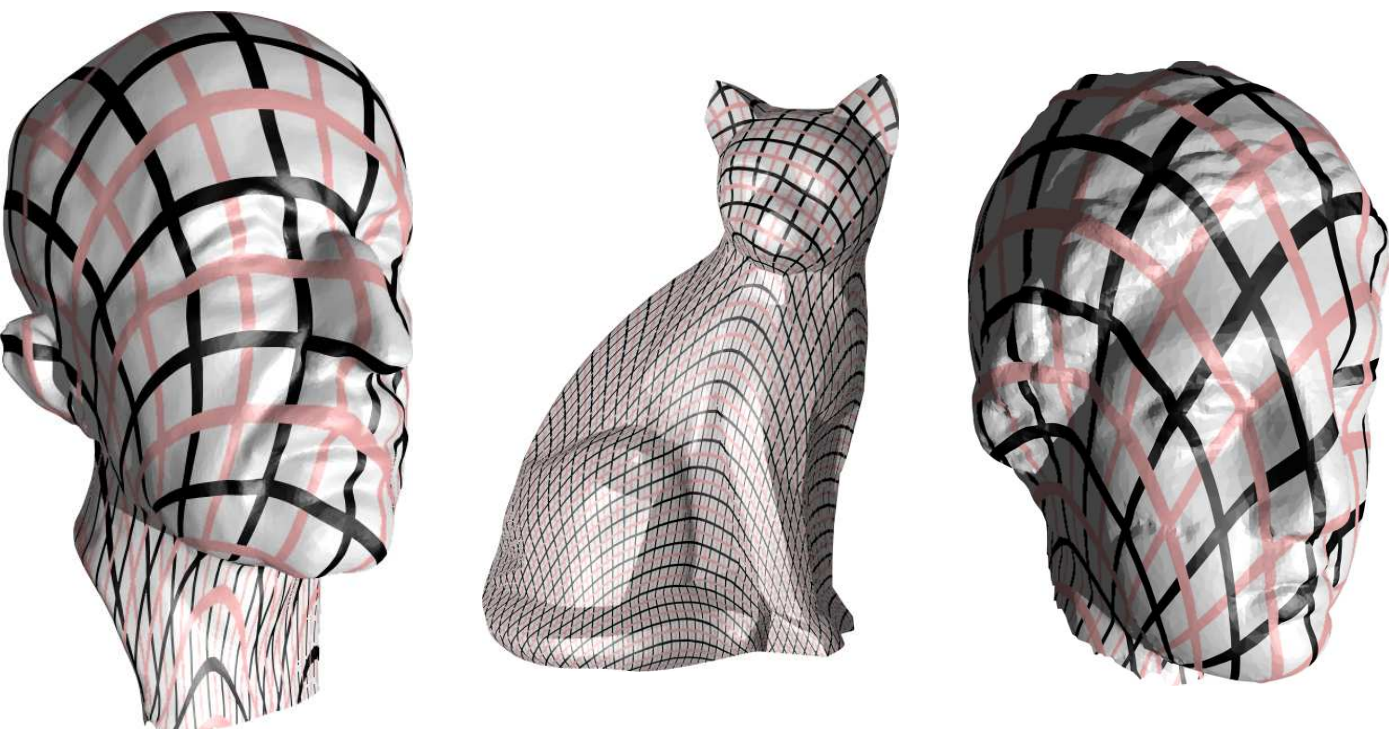


Figure 6.2: Parameterizations obtained by minimizing  $E_{length}$ . To visualize the parameterization a regular texture was mapped to the surface.



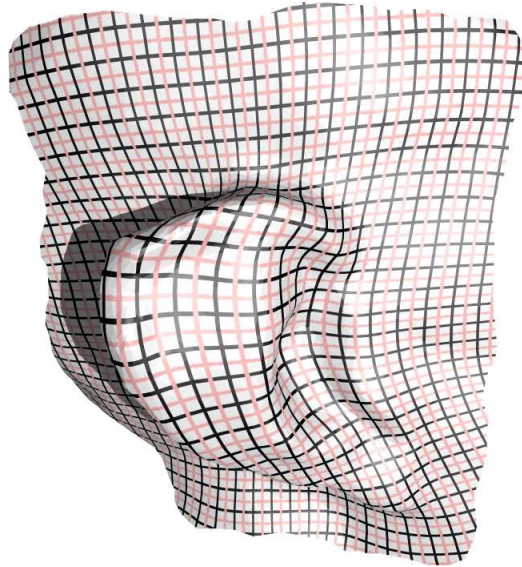


Figure 6.3: Parameterization for a chart cut from Max Planck model. The ratio of surface area to boundary length is far lower than for the whole model. This reduces shape deformation. The parameterization is also computed as minimum of  $E_{length}$ . Shape deformation is hardly noticeable.

$E_{length}$ . To visualize shape deformation, we texture mapped a regular texture consisting of straight lines that cross at angles of  $\pi/2$ . As the shown surfaces are non developable the optimization has to make a trade off between the preservation of intersection angles and area deformation.

In practice, surfaces as shown in this figure are not parameterized as whole but cut into charts to decrease the surface area to boundary length ratio and thus to reduce the unavoidable shape deformation. While, this preprocessing was skipped to reveal the type of shape deformation in Figure 6.2, the parameterization certainly improves a lot by using a charting or seaming algorithm, as demonstrated in Figure 6.3.

Figure 6.4 compares several potentials on the cat model. Both the LSCM energy and MIPS energy measure angle deformation only. As the combined potential coincides with the MIPS energy for  $\alpha = 0$ , the result is similar to that of the LSCM energy in that the angles at cross sections are nicely preserved. While the variation in the conformal factor seems to be a bit higher for the MIPS energy, both maps show significant area deformation. In contrast, the parameterization for  $\alpha = 1$  trades some of the angle preservation to reduce the area deformation. The results for the  $E_{stretch-2}$  does also preserve shapes on large parts of the surface



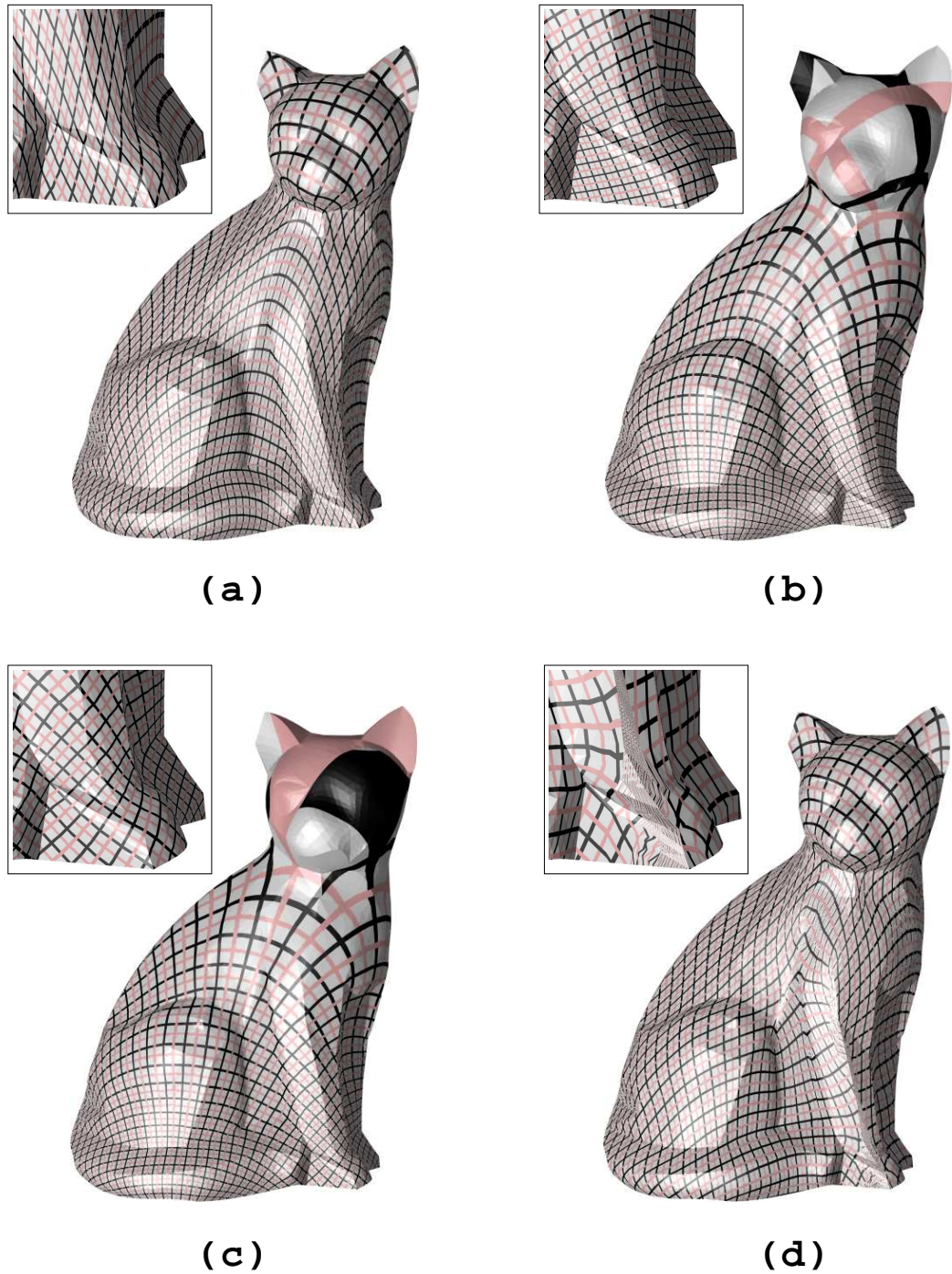


Figure 6.4: Parameterizations of the cat model using different potentials: (a) the combined potential with  $\alpha = 1$  ( $E_{length}$ ), (b) the LSCM energy, (c) the combined potential with  $\alpha = 0$  ( $E_{angle}/MIPS$ ) and (d) the  $E_{stretch-2}$  energy.

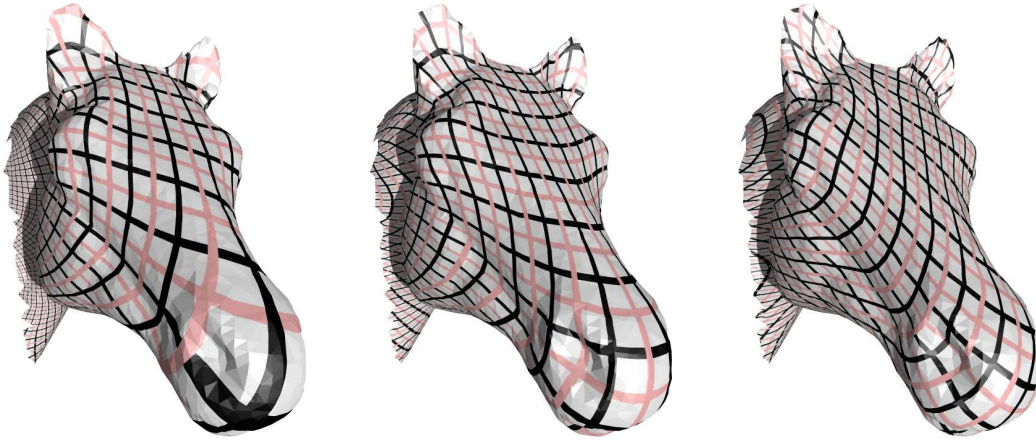


Figure 6.5: Influence of the parameter  $\alpha$ : The horse head model parameterized using different angle/area preservation tradeoffs. The texture mapped model is shown for (from left to right)  $\alpha = 0.3$ ,  $\alpha = 1.0$  and  $\alpha = 3.0$ .

but exhibits regions of high anisotropy (see gap in the close up of the paw).

The influence of the  $\alpha$  parameter of the combined potential is shown in Figure 6.5. As expected, the intersection angles are well preserved for low values of  $\alpha$  while the parameterization exhibits a great amount of area deformation. As the importance of the area term is increased, the parameterizations show lower area deformations. Since the surface is not developable the angle deformation increases at the same time. For very high values of  $\alpha$ , the angles are greatly distorted but at the same time the images of the squares on the surface nearly have the same area.

This observation can also be verified in the angle and area distortion histograms shown in Figure 6.6. These histograms were created from the parameterizations by measuring the per triangle angle and area deformation. The angle deformation was computed as the angle between the directional derivatives and the area deformation as the 2D area/3D area ratio. For low values of  $\alpha$  the angle deformations are concentrated in a narrow peak around the optimal value of 90 degree. At the same time the area deformation histogram show a very high variance. For high  $\alpha$  values this relation is reversed.

In Figure 6.7, the run times for the hierarchical optimization of the combined energy (with  $\alpha = 1$ ) are listed for different models. The direct comparison with the quadratic LSCM potential stresses the advantages of a quadratic energy functional. Although our optimization algorithm is rather simple and recently, im-

## 6.2. EXPERIMENTAL EVALUATION

---

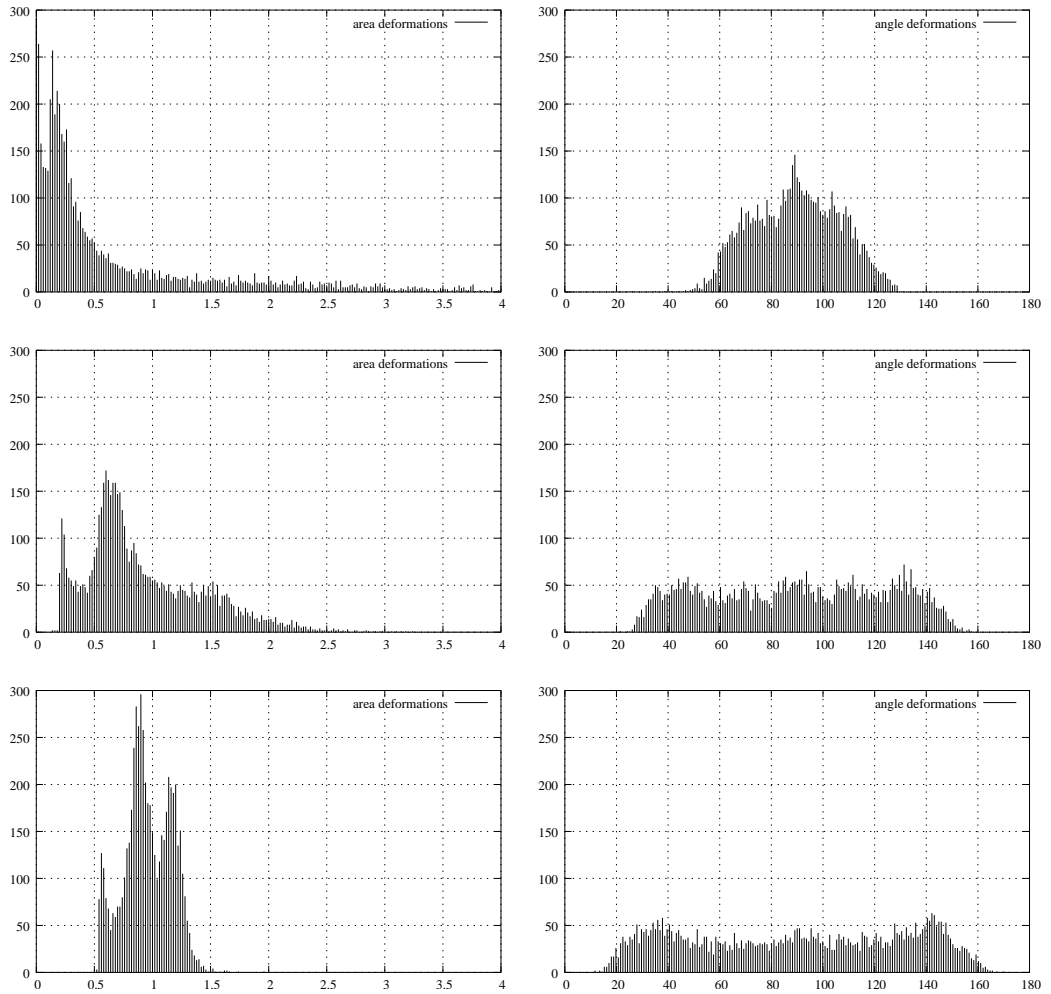


Figure 6.6: Per triangle distortions in area and angle for the horse head data set for (from top to bottom)  $\alpha = 0.3$ ,  $\alpha = 1.0$  and  $\alpha = 3.0$ , as shown in figure 6.5.

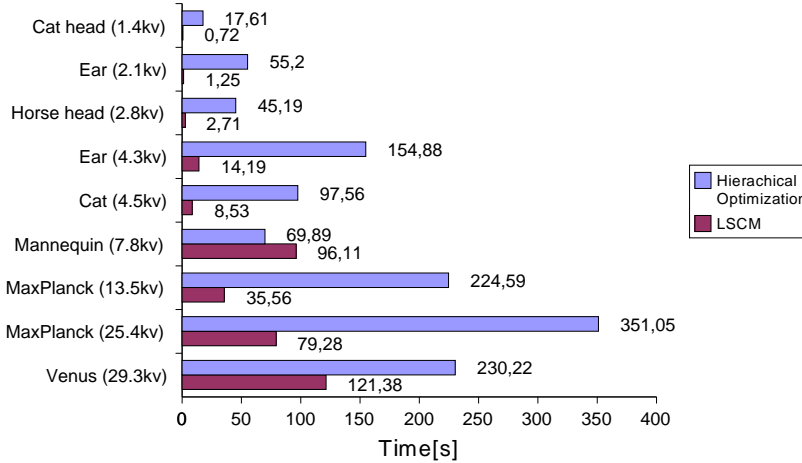


Figure 6.7: Run times for the hierarchical optimization. For comparison, also the run times of the linear LSCM method are shown.

proved optimization methods for the combined potential have been proposed (we will comment on these in Section 6.3), it still seems that a significantly higher optimization time is the price to pay for parameterizations with balanced angle / area distortion. We also point out, that the run time does not increase monotonously with the number of vertices. Other properties like for example the curvature, the size of the boundary loops or the quality of the triangulation also influence the convergence of the optimization.

### 6.3 Discussion

In the last chapters, potentials for surface parameterization over a planar domain were analyzed with a focus on the preservation of the surface metric. While advanced methods exist for conformal parameterization, only few potentials capture both angle and area distortion. Moreover, these potentials suffer from certain drawbacks.

Therefore, a novel potential method was derived that captures both angle and area deformations. The importance weighting parameter  $\alpha$  allows the user to specify the relative importance of angle and area preservation. For  $\alpha = 1$  the potential takes a particular simple form and can be interpreted as a measure of length deformation. The combined potential tends to infinity as triangles degenerate and this property can be used to ensure flipless parameterizations. Moreover, it does not need special boundary conditions and thus enables a natural optimization of the

boundary of the texture domain  $\bar{S}$ .

A simple method to minimize this deformation measure was proposed, that ensures the bijectivity of the resulting mapping. We evaluated the resulting parameterization method on a number of surfaces as commonly found in computer graphics applications. The resulting parameterizations show the desired compromise between angle and area deformation on surfaces that are far from developable. On nearly developable surfaces as they result after cutting or seaming the shape deformation is hardly noticeable.

From the theoretical point of view, a still open question is the uniqueness (up to rigid transformation) of the minimum. While our experiments suggest that the minimum is unique as long as triangles are constrained to consistent orientation, a proof is still missing. An even more fundamental question is whether a minimizing parameterization exists at all. Clarenz et al. [CLR04a] derived a class of parameterization potentials from a set of fundamental properties. Potentials of this class are very similar to the combined potential proposed here. They can also be expressed in terms of the angle and area potential but combine those potentials linearly in contrast to the multiplication of powers proposed here. In addition, Clarenz et al. add a length potential that is similar to the sampling energy  $E_{stretch-2}$ . Similar to our combined energy, the influence of angle, area and length deformation can be specified using linear importance weights. For potentials of this class, Clarenz et al. were able to prove the existence of a minimizing parameterization not only on triangulated surfaces but also in the smooth case.

While the optimization algorithm proposed here is easy to implement and guarantees the absence of flips, it is rather slow compared to linear methods. Recent work therefore also targets at more efficient optimization algorithm. A simultaneous optimization of the texture coordinates of all vertices is likely faster than the vertex relaxation currently used in our implementation. This approach was also taken by Clarenz et al. [CLR04a] who use a Newton method with adaptive line search. However, the authors give no timings for comparison. The iterative method by Dong and Garland [DG07] alternates between two steps. In the first step optimal positions for each vertex within its one ring are computed similar to the relaxation optimization. In contrast to relaxation, vertex positions are not immediately updated. To update positions, the optimal position of each vertex is expressed in terms of the position of neighboring vertices by mean value coordinates [Flo03]. This gives rise to a linear system which is solved to update all positions simultaneously. The reported run times are orders of magnitude faster than our hierarchical optimization but their method needs fixed boundary values. As similar idea was also pursued by Liu et al. [LZX<sup>+</sup>08]. Instead of locally optimal texture coordinates for each vertex, they compute a locally optimal shape for each triangle. These shapes are then integrated in a second phase using a fast linear system. Similar to the approach of Dong and Garland, both steps are repeated

until the method converges to a fix point. Their optimization is very efficient, but cannot guarantee the absence of flips. While the global/local approaches are promising, the question, how the proposed potential (or similar metric potentials) can be optimized efficiently with natural boundary conditions and guaranteed bijectivity is therefore still open.



---

### MATERIAL SPECIFIC TEXTURE MAPS AND PATTERNS FOR FABRIC COVERED SURFACES

---

A large range of industrial products are either made from or covered with an elastic fabric that is initially supplied in a two dimensional form such as steel, woven textiles or rubber. Important examples include cloth and garments, sails, seats and other upholstery, ship hulls and shoes. The production of such objects usually starts from planar pieces ( the *pattern* ) which are stitched together or bend to follow a 3D surface. If the surface itself is not developable, which is in general not the case, the fabric undergoes a deformation in the production process. This in turn causes distortions in the material distribution of the surface that are in particular visible, if the fabric shows some kind of pattern, as e.g. weaving patterns or imprints. An example of such an object is shown in Figure 7.1.

With modern acquisition technology at hand, it is easy to obtain the geometry of the covered or strained surface. However, the underlying fabric pattern and the material deformation cannot be acquired in that way. While this information is not needed to reproduce the exact appearance of the acquired object, changes of material, texture or patterns, as required to evaluate different options e.g. in the design of cloth, automotive seats or shoes are not possible.

Certainly the parameterization methods described in the last chapters can be used to generate a texture map of the acquired geometry. The resulting deformation will, however, vary depending on the chosen potential and will most likely not correspond to the actual material deformation as those potentials (with the notable exception of the elastic membrane potential) do not reflect physical material properties and, second, are of isotropic nature, while most existing materials are anisotropic.

In computer graphics much effort has been put in measurement and reproduction of realistic material reflectance properties over the last years [LGM07]. In contrast, texture maps with realistic material deformation have not been considered so far. In this chapter we compute texture maps that show realistic defor-

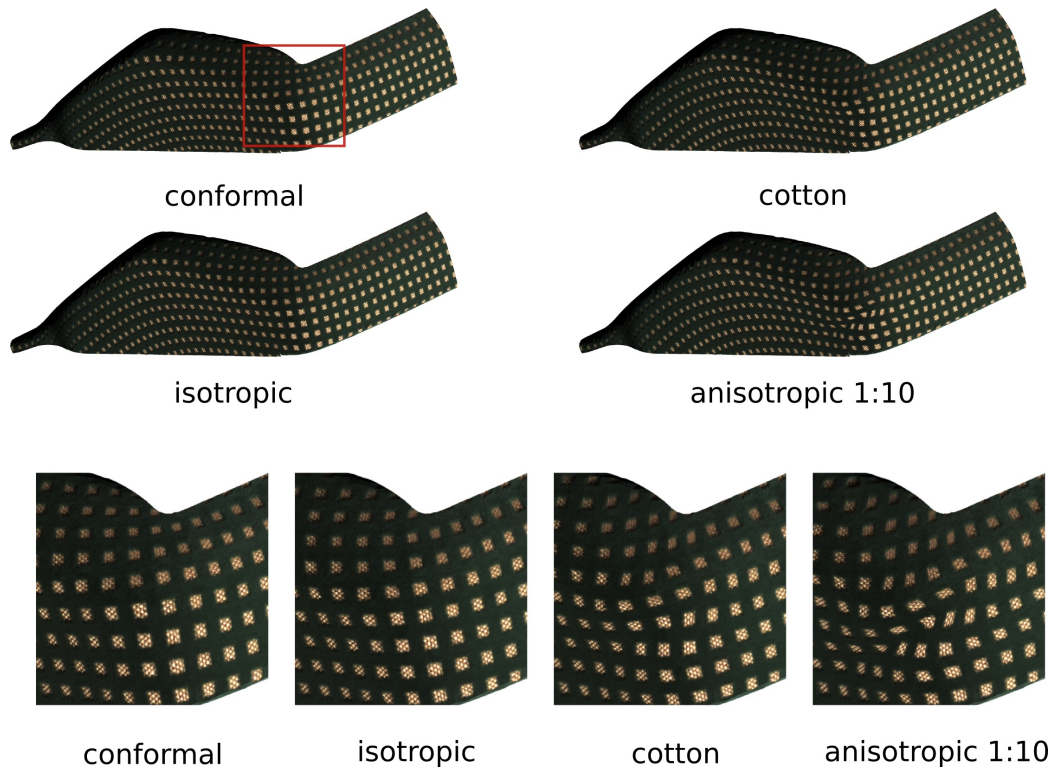


Figure 7.1: Visible fabric deformation on a part of an automotive seat. On top different texture maps are shown while texture and geometry does not change. Texture maps are created using the conformal potential and the membrane potential for various materials. In the bottom closeups are shown. “Isotropic” refers to an isotropic material and “anisotropic 1:10” to a material with a ratio of directional anisotropy of 1 : 10. “Cotton” refers to measured material parameters with anisotropy ratio of 1 : 2.2.



mations coherent with a given material. In contrast to parameterization methods discussed in the last chapters, that aim at mostly isometric mappings, the intention of this chapter is to produce texture maps that show shape distortion reflecting the physical material strain in the production process, including effects due to anisotropic materials. To this extent, we will revisit the physical elastic membrane potential for parameterization. We see the applications for such *material specific texture maps* primarily in connection with high quality material rendering methods.

To compute material specific texture maps, we will assume that the original pattern is known. This is usually the case once the object in question has reached production stage as the pattern is essential for production. However, while production of objects as characterized above starts from planar patterns, their design often proceeds in the opposite direction. For example in the design of automotive seating first a three dimensional shape is sketched. Appropriate flat patterns are then generated in a second step which is also referred to as *flattening*. In practice, even today, flattening is usually done in manual trial and error process. As manual flattening proceeds in several iterations, even a rough approximation can significantly reduce the number of iterations and thus the required effort. The approach suggested in this chapter is based on surface parameterization with natural boundary conditions. As the outline of the preimage  $\bar{S} = \tilde{\mathbf{x}}^{-1}(S)$  evolves freely from the optimization we expect it to at least approximate the actual, unknown patterns.

If both pattern and geometry of underlying shapes are known, the final surface and material deformation can be inferred by physical cloth or draping simulation [NMK<sup>+</sup>06]. The work presented here is complementary to such simulations: It assumes the shape of the covered or strained surface as fixed and solves for patterns or material deformations for a given material. We thus consider only intrinsic deformations and neglect extrinsic shape deformation due to pattern or material changes. While the extrinsic shape can in general vary strongly, our assumption of a static shape is nevertheless justified in many applications where external forces enforce a certain, desired shape as e.g. tight fitting cloth, upholstered furniture, automotive seats or shoes.

## 7.1 Material Specific Texture Maps

As shown in Figure 7.2 we consider in this section an initial planar pattern boundary  $\bar{S}$  and a target shape  $S$  as given and solve for deformation  $\tilde{\mathbf{x}} : \bar{S} \rightarrow S$ . As the map  $\tilde{\mathbf{x}}$  should approximate the actual material deformation of the fabric, we will also need some information about the elastic properties of the material which will be detailed in a moment. While the shape  $S$  is potentially influenced by the

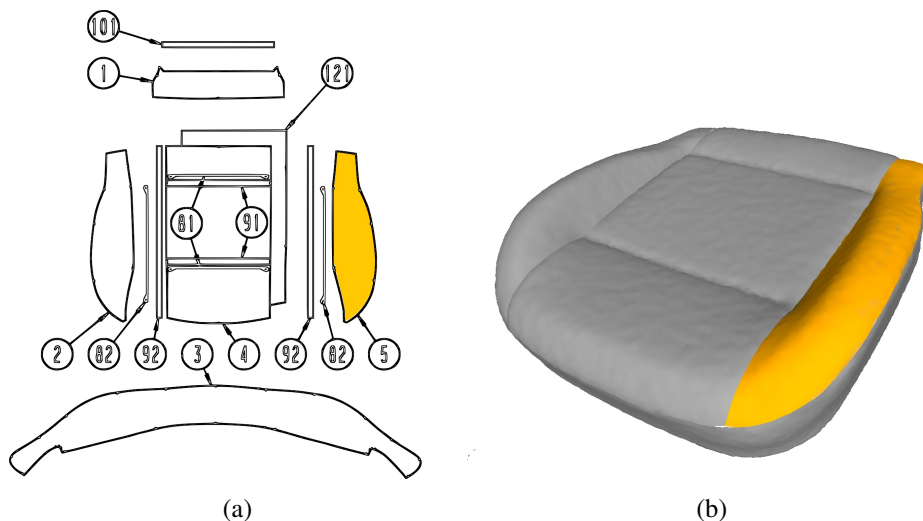


Figure 7.2: This data is assumed as given in this section: (a) A planar pattern. (Small annotations give additional sewing instructions). (b) the acquired surface of the final upholstered seat. One connected component of the pattern is highlighted.

choice of the covering fabric as well as the elastic properties of the cushioning, we neglect these effects and assume the shape of the target shape  $S$  as fixed. Albeit it seems strong, this assumption is justified in many applications. For example, in the production of automotive seating shape changes due to varying cover materials are usually neglected.

To find the resulting deformation  $\tilde{\mathbf{x}}$  we follow the plaster cast metaphor introduced by Litke et al. [LDRS05] for surface matching that is illustrated in Figure 7.3 but adapt it to respect additional sewing information: First, we imagine the boundary of the planar pattern sewn to the boundary of  $S$ . This sewing is guided by additional sewing instruction annotated in the pattern as shown in Figure 7.2a that give rise to a mapping

$$\mathbf{b} : \delta\bar{S} \rightarrow \delta S$$

between the boundaries  $\delta\bar{S}$  and  $\delta S$  of the pattern and the surface. Sewing boundaries together might already result in a significant membrane strain but the stretched pattern still need not follow the given shape. It is therefore in a second step, pressed between two plaster casts taken from the surface  $S$ . Neglecting friction, the fabric will now float between the two casts according to material restoring forces to relax but at the same time it cannot leave the given shape. The final deformation is then obtained as equilibrium configuration, i.e. as a minimum of the elastic potential.

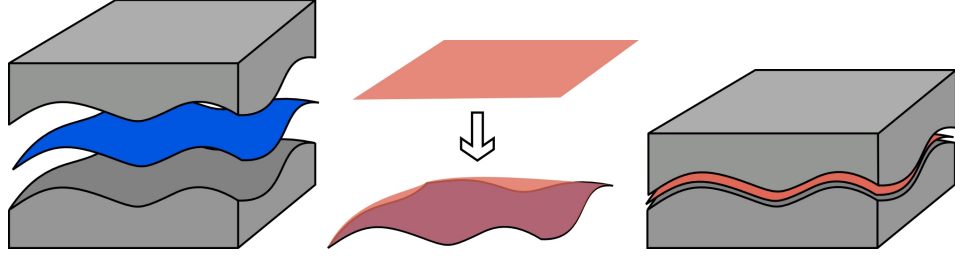


Figure 7.3: We use the plaster cast metaphor from [LDRS05] to find material deformation: (a) A plaster cast is extracted from the original surface  $S$  shown in blue. (b) The boundary of the pattern shown in red is sewn to the boundary of  $S$ . (c) To enforce adherence to the given shape  $S$ , the casts are pressed from both sides to the deformed pattern.

In addition, to the constant shape assumption we consider fabrics with negligible bending resistance like textiles or sufficiently thin shells. Consequently, the bending term of the elastic potential in Equation 3.13 can be omitted. The deformation  $\tilde{\mathbf{x}}$  is thus found as minimizer of the elastic membrane potential  $e_{membrane}$  alone. Using the general deformation setup from Figure 3.2 as in the previous chapters relaxing the fabric between the two casts can be mathematically modeled by considering variations of the membrane potential with respect to the parameterization  $\mathbf{x}$ . To respect sewing instructions, these variations must be restricted so that

$$\tilde{\mathbf{x}}|_{\bar{S}} = \mathbf{b} . \quad (7.1)$$

In addition, it must be ensured that the given shape  $S$  is not altered, and thus variations have to be restricted further to work only tangential. However, this latter restriction poses technical problems for triangulated surfaces. We therefore take a different approach here: In the smooth case, tangential variations of the parameterization  $\mathbf{x}$  can be mapped back to variations of the parameterization  $\bar{\mathbf{x}}$ . As  $\bar{\mathbf{x}}$  maps into a planar space the restrictions to tangential variations can be omitted and no other constraint other than Equation 7.1 has to be imposed. Computing material specific texture maps thus amounts to minimizing the elastic membrane potential  $E_{membrane}(\tilde{\mathbf{x}}) = E_{membrane}(\mathbf{x} \circ \bar{\mathbf{x}}^{-1})$  with respect to the parameterization  $\bar{\mathbf{x}}$  subject to condition 7.1.

For an isotropic material the membrane potential's density was given in Equation 3.13. For a general anisotropic elastic material the density takes a similar form:

$$e_{membrane} = \frac{h}{2} C^{\alpha\beta\gamma\delta} (g_{\alpha\beta} - \bar{g}_{\alpha\beta})(g_{\gamma\delta} - \bar{g}_{\gamma\delta}) \quad (7.2)$$

where  $h$  is the constant shell thickness and  $C^{\alpha\beta\gamma\delta}$  is a material specific elasticity tensor (for the choice of  $C^{\alpha\beta\gamma\delta} = \frac{E}{(1-\nu^2)} H^{\alpha\beta\gamma\delta}$  we obtain the density for an

isotropic material as given in Equation 3.13). In the above equation, the tensor  $C^{\alpha\beta\gamma\delta}$  specifies material properties in coordinates  $\theta^\alpha$  of  $\omega$  and thus changes under variations of the parameterization  $\bar{x}$ . In our setting it is more convenient to consider the same tensor in coordinates  $\bar{x}^\alpha$  of  $\bar{S}$  which is constant with respect to variations of  $\bar{x}$ . Now,  $C^{\alpha\beta\gamma\delta}$  is a so called *contravariant tensor*, which means that it transforms from coordinates  $\theta^\alpha$  to coordinates  $\bar{x}^\alpha$  according to

$$c^{\alpha\beta\gamma\delta} = C^{\alpha'\beta'\gamma'\delta'} \frac{\partial \bar{x}^\alpha}{\partial \theta^{\alpha'}} \frac{\partial \bar{x}^\beta}{\partial \theta^{\beta'}} \frac{\partial \bar{x}^\gamma}{\partial \theta^{\gamma'}} \frac{\partial \bar{x}^\delta}{\partial \theta^{\delta'}} \quad (7.3)$$

where  $c^{\alpha\beta\gamma\delta}$  denotes the same tensor in coordinates  $\bar{x}^\alpha$ . Substituting this in Equation 7.2 and using the following simple identities

$$\frac{\partial \theta^{\alpha'}}{\partial \bar{x}^\alpha} \frac{\partial \theta^{\beta'}}{\partial \bar{x}^\beta} g_{\alpha'\beta'} = \frac{\partial \theta^{\alpha'}}{\partial \bar{x}^\alpha} \frac{\partial \theta^{\beta'}}{\partial \bar{x}^\beta} \frac{\partial x^\gamma}{\partial \theta^{\alpha'}} \frac{\partial x^\gamma}{\partial \theta^{\beta'}} = \frac{\partial x^\gamma}{\partial \bar{x}^\alpha} \frac{\partial x^\gamma}{\partial \bar{x}^\beta} = \tilde{g}_{\alpha\beta} \quad (7.4)$$

$$\frac{\partial \theta^{\alpha'}}{\partial \bar{x}^\alpha} \frac{\partial \theta^{\beta'}}{\partial \bar{x}^\beta} \bar{g}_{\alpha'\beta'} = \frac{\partial \theta^{\alpha'}}{\partial \bar{x}^\alpha} \frac{\partial \theta^{\beta'}}{\partial \bar{x}^\beta} \frac{\partial \bar{x}^\gamma}{\partial \theta^{\alpha'}} \frac{\partial \bar{x}^\gamma}{\partial \theta^{\beta'}} = \frac{\partial \bar{x}^\gamma}{\partial \bar{x}^\alpha} \frac{\partial \bar{x}^\gamma}{\partial \bar{x}^\beta} = \delta_{\alpha\beta} \quad (7.5)$$

we get

$$\begin{aligned} e_{membrane} &= \frac{h}{2} c^{\alpha\beta\gamma\delta} \frac{\partial \theta^{\alpha'}}{\partial \bar{x}^\alpha} \frac{\partial \theta^{\beta'}}{\partial \bar{x}^\beta} (g_{\alpha'\beta'} - \bar{g}_{\alpha'\beta'}) \frac{\partial \theta^{\gamma'}}{\partial \bar{x}^\gamma} \frac{\partial \theta^{\delta'}}{\partial \bar{x}^\delta} (g_{\gamma'\delta'} - \bar{g}_{\gamma'\delta'}) \\ &= \frac{h}{2} c^{\alpha\beta\gamma\delta} (\tilde{g}_{\alpha\beta} - \delta_{\alpha\beta})(\tilde{g}_{\gamma\delta} - \delta_{\gamma\delta}) \end{aligned}$$

The elastic membrane potential for an anisotropic material is then obtained by integration over the undeformed configuration surface  $\bar{S}$  as

$$E_{membrane} = \int_{\bar{S}} e_{membrane} d\bar{S} = \int_{\omega} e_{membrane} (\det \bar{g})^{\frac{1}{2}} d\omega$$

Computing material dependent texture maps for an isotropic material is thus nearly equivalent to planar parameterization using the elastic energy as discussed in Section 4.2.2. The only difference is the constraint 7.1 that fixes the boundaries to the outlines of pattern patches. In context of parameterization, we observed gap artifacts that were due to the missing convexity of the membrane potential. The same problem can be observed for other materials and fixed boundaries and thus also applies to material dependent texture maps.

The above given physical metaphor allows for a different perspective on these problems: The linear elastic model as introduced in Section 3.1 that is the basis for Equation 7.2 assumes a linear stress/strain relation. In case of an isotropic material this relation was given in Equation 3.12 and it is easily generalized for anisotropic materials as

$$n^{\alpha\beta} = h C^{\alpha\beta\gamma\delta} (g_{\gamma\delta} - \bar{g}_{\gamma\delta}) \quad (7.6)$$

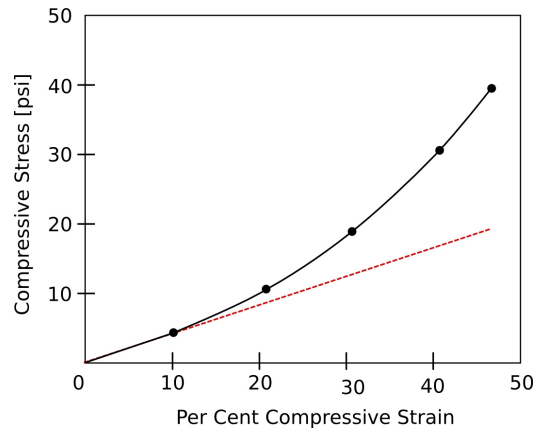


Figure 7.4: The stress strain curve for felt under compressive strain. While for small strain, the relation is nearly linear, it quickly becomes non-linear for larger compression.

where  $n^{\alpha\beta}$  is the stress tensor in coordinates  $\theta^\alpha$ . While this is a reasonable approximation for many fabrics under small strain, it no longer holds for larger deformations. The stress strain relation of real materials becomes non symmetric for compression and stretch. In particular for compression, many fabrics show a non-linear stress strain relation under strong compression as shown in Figure 7.4.

Figure 7.5 again shows a closeup on the parameterization of the cat head model as it was shown in Figure 4.3. Looking at the gap artifacts, we notice that the highlighted cross section in  $\bar{S}$  is mapped to a small line segment on the surface  $S$ . It therefore undergoes a large compressive strain when it is mapped by  $\tilde{x}$ . Clearly, the linear stress/strain assumption no longer holds in this case.

In context of parameterizations surfaces are often far from developable and large strain cannot be avoided as in the case of the cat head model. Surfaces considered in this chapter are obtained by putting a fabric into shape in a physical production process. To ease production, care is usually taken in the design of patterns to keep strain small as large strain hinders upholstery. (In particular, for automotive seating and upholstery very low compressive strain can be safely assumed: This is because compressive strain in the fabric leads to folds and wrinkles which are highly undesired.) For simplicity we therefore stick to the linear model in this chapter. However, it certainly depends on the application, whether the assumption of a linear stress strain relation is justified. We will come back to this issue in Section 7.3.

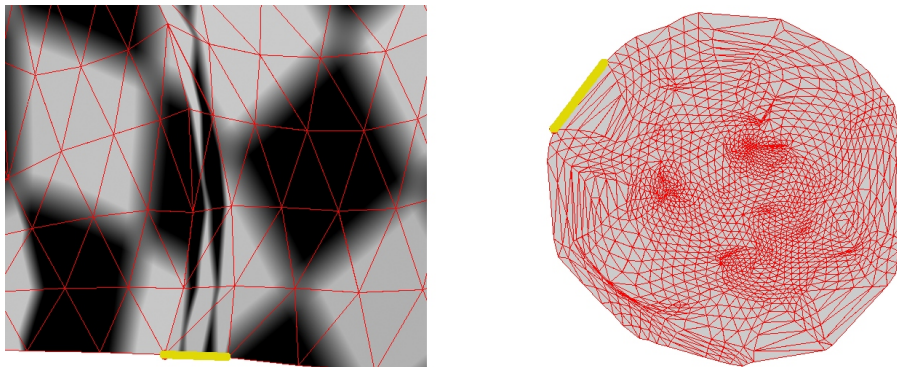


Figure 7.5: The parameterization of the cat head model (see 4.3) using the elastic membrane potential. The large highlighted cross section in the gap in the texture domain  $\bar{S}$  (right) is mapped onto a small line segment on the surface  $S$  (right). This corresponds to a large compressive strain in this area.

### 7.1.1 Discretization and Numerical Optimization

Material specific texture maps are computed by optimizing the elastic potential subject to the constraint 7.1. For discretization we again assume piecewise affine maps  $\mathbf{x}$  and  $\bar{\mathbf{x}}$ . First order derivatives and all derived properties like metric tensors are then constant within each triangle and discretization proceeds as described in Section 2.2.2.

In the discrete case, the constraint 7.1 prescribed values for boundary vertices. Consequently, we simply exclude these vertices from the optimization and keep their values fixed. While the simple relaxation algorithm outlined in Section 6.1.1 can be applied with almost no changes (relaxation is only performed on interior vertices), adapting the more efficient hierarchical optimization in Section 6.1.2 to respect fix boundary conditions poses problems. The initial placement of split vertices ensuring an admissible parameterization might conflict with the prescribed boundary values so that absence of flips can no longer be guaranteed.<sup>1</sup> Although admissibility in general cannot be guaranteed, we encountered no such problems in our experiments if actual pattern curves were used to fix boundaries.

Having stripped the guaranteed fliplessness requirement, we can also employ standard Newton trust region optimization instead of vertex relaxation to further speed up the optimization. Applying the discretization scheme detailed in Section 2.2.2 for an implementation only first and second order variations of the elastic potential are missing, that we will state in the following. As for the variations of the combined potential in Section 5.5.3, we consider a small change  $\bar{\mathbf{x}} + \epsilon \bar{\mathbf{y}}$  of  $\bar{\mathbf{x}}$

<sup>1</sup>In fact e.g. for a self intersecting boundary no admissible parameterization is possible. But flips can also result for intersection free boundary shapes.

in the direction of map  $\bar{\mathbf{y}}$  and write

$$\partial_{\bar{\mathbf{x}}} F[\bar{\mathbf{x}}](\bar{\mathbf{y}}) = \left. \frac{\partial}{\partial \epsilon} F[\mathbf{x} \circ (\bar{\mathbf{x}} + \epsilon \bar{\mathbf{y}})^{-1}] \right|_{\epsilon=0}$$

to denote the variation of a functional  $F$  with respect to  $\bar{\mathbf{x}}$ . (By considering only maps  $\bar{\mathbf{y}}$  that vanish on the boundary  $\delta\omega$  we can easily account for the boundary constraint.) The expressions summarized in the following can be obtained applying basic rules of calculus and the equalities given in Section 5.5.3.

$$\begin{aligned} \partial_{\bar{\mathbf{x}}} E_{\text{membrane}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}) &= \int_{\omega} (\partial_{\bar{\mathbf{x}}} e_{\text{membrane}}[\bar{\mathbf{x}}](\bar{\mathbf{y}})) (\det \bar{\mathbf{g}})^{\frac{1}{2}} + e_{\text{membrane}} \frac{\partial_{\bar{\mathbf{x}}} \det \bar{\mathbf{g}}}{2(\det \bar{\mathbf{g}})^{\frac{1}{2}}} d\omega \\ \partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} E_{\text{membrane}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}, \bar{\mathbf{z}}) &= \int_{\omega} (\partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} e_{\text{membrane}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}, \bar{\mathbf{z}})) (\det \bar{\mathbf{g}})^{\frac{1}{2}} + \\ &+ \partial_{\bar{\mathbf{x}}} e_{\text{membrane}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}) \frac{\partial_{\bar{\mathbf{x}}} \det \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{z}})}{2(\det \bar{\mathbf{g}})^{\frac{1}{2}}} + \\ &+ \partial_{\bar{\mathbf{x}}} e_{\text{membrane}}[\bar{\mathbf{x}}](\bar{\mathbf{z}}) \frac{\partial_{\bar{\mathbf{x}}} \det \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}})}{2(\det \bar{\mathbf{g}})^{\frac{1}{2}}} + \\ &+ e_{\text{membrane}} \left( \frac{\partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} \det \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}, \bar{\mathbf{z}})}{2(\det \bar{\mathbf{g}})^{\frac{1}{2}}} - \right. \\ &\quad \left. - \frac{\partial_{\bar{\mathbf{x}}} \det \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}) \partial_{\bar{\mathbf{x}}} \det \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{z}})}{4(\det \bar{\mathbf{g}})^{\frac{3}{2}}} \right) d\omega \\ \partial_{\bar{\mathbf{x}}} e_{\text{membrane}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}) &= hc^{\alpha\beta\gamma\delta} (\tilde{g}_{\alpha\beta} - \delta_{\alpha\beta}) (\partial_{\bar{\mathbf{x}}} \tilde{g}[\bar{\mathbf{x}}](\bar{\mathbf{y}}))_{\gamma\delta} \\ \partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} e_{\text{membrane}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}, \bar{\mathbf{z}}) &= hc^{\alpha\beta\gamma\delta} ((\partial_{\bar{\mathbf{x}}} \tilde{g}[\bar{\mathbf{x}}](\bar{\mathbf{y}}))_{\alpha\beta} (\partial_{\bar{\mathbf{x}}} \tilde{g}[\bar{\mathbf{x}}](\bar{\mathbf{z}}))_{\gamma\delta} + \\ &\quad + (\tilde{g}_{\gamma\delta} - \delta_{\gamma\delta}) (\partial_{\bar{\mathbf{x}}} \tilde{g}[\bar{\mathbf{x}}](\bar{\mathbf{y}}, \bar{\mathbf{z}}))_{\alpha\beta}) \\ \partial_{\bar{\mathbf{x}}} \det \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}) &= \text{tr} (\bar{\mathbf{g}}^{-1} \partial_{\bar{\mathbf{x}}} \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}})) \det \bar{\mathbf{g}} \\ \partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} \det \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}, \bar{\mathbf{z}}) &= (\text{tr} (\bar{\mathbf{g}}^{-1} \partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} \bar{\mathbf{g}}}) - \text{tr} (\partial_{\bar{\mathbf{x}}} \bar{\mathbf{g}}(\bar{\mathbf{y}}) \bar{\mathbf{g}}^{-1} \partial_{\bar{\mathbf{x}}} \bar{\mathbf{g}}(\bar{\mathbf{z}}) \bar{\mathbf{g}}^{-1})) + \\ &\quad + \text{tr} (\bar{\mathbf{g}}^{-1} \partial_{\bar{\mathbf{x}}} \bar{\mathbf{g}}(\bar{\mathbf{y}})) \text{tr} (\bar{\mathbf{g}}^{-1} \partial_{\bar{\mathbf{x}}} \bar{\mathbf{g}}(\bar{\mathbf{z}})) \det \bar{\mathbf{g}} \\ (\partial_{\bar{\mathbf{x}}} \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}))_{\alpha\beta} &= \bar{y}_{i,\alpha} \bar{x}_{i,\beta} + \bar{x}_{i,\alpha} \bar{y}_{i,\beta} \\ (\partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} \bar{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}, \bar{\mathbf{z}}))_{\alpha\beta} &= \bar{y}_{i,\alpha} \bar{z}_{i,\beta} + \bar{z}_{i,\alpha} \bar{y}_{i,\beta} \\ (\partial_{\bar{\mathbf{x}}} \tilde{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}))_{\alpha\beta} &= 2g_{\gamma\delta} \bar{x}_{\gamma,\alpha}^{-1} \partial_{\bar{\mathbf{x}}} \bar{x}_{\delta,\beta}^{-1} \\ (\partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} \tilde{\mathbf{g}}[\bar{\mathbf{x}}](\bar{\mathbf{y}}, \bar{\mathbf{z}}))_{\alpha\beta} &= 2g_{\gamma\delta} ((\partial_{\bar{\mathbf{x}}} \bar{x}^{-1}(\bar{\mathbf{z}}))_{\gamma,\alpha} (\partial_{\bar{\mathbf{x}}} \bar{x}^{-1}(\bar{\mathbf{y}}))_{\delta,\beta} + \bar{x}_{\gamma,\alpha}^{-1} \partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} \bar{x}_{\delta,\beta}^{-1}) \\ \partial_{\bar{\mathbf{x}}} (\bar{x}^{-1}[\bar{\mathbf{x}}](\bar{\mathbf{y}}))_{\alpha,\beta} &= -\bar{x}_{\alpha,\gamma}^{-1} \bar{y}_{\gamma,\delta} \bar{x}_{\delta,\beta}^{-1} \\ \partial_{\bar{\mathbf{x}\bar{\mathbf{x}}} (\bar{x}^{-1}[\bar{\mathbf{x}}](\bar{\mathbf{y}}, \bar{\mathbf{z}}))_{\alpha,\beta} &= \bar{x}_{\alpha,\mu}^{-1} \bar{z}_{\mu,\nu} \bar{x}_{\nu,\gamma}^{-1} \bar{y}_{\gamma,\delta} \bar{x}_{\delta,\beta}^{-1} + \bar{x}_{\alpha,\gamma}^{-1} \bar{y}_{\gamma,\delta} \bar{x}_{\delta,\mu}^{-1} \bar{z}_{\mu,\nu} \bar{x}_{\nu,\beta}^{-1} \end{aligned}$$



## 7.2 Application: Pattern Generation for Upholstery

As sketched in the introduction, the production of fabric covered surfaces starts from a planar pattern, that is cut, sewn and bent into its final shape. The design of fabric covered surfaces usually proceeds in the opposite direction: For example the shape of automotive seats is typically prototyped using a clay model or a digital surface model. For upholstering, first the position and routes of seams have to be specified on the surface. As placement of seams is subject to aesthetic considerations it is to large parts left to the designer although an upholsterer has to check the desired seam placement for technical feasibility. The second step then consists in finding a planar pattern and associated sewing instructions. This happens in a manual trial and error process in which the pattern is iteratively updated, sewn and upholstered to check for a tight fit. The ultimate goal can vary with applications but often the tightest possible fit and absence of wrinkles are desirable.

To reduce the number of iterations in pattern design, automatic methods have already been proposed. However, upholstering is an inherently complex task influenced by many different factors as material properties of upholstery frame, surface fabric but also friction, temperature and humidity. Computing a perfectly fitting pattern thus requires knowledge of many physical parameters whose measurement is not practical or too costly. Automatic approaches as the one proposed here therefore target only at a reduction of the necessary manual iterations.

By optimizing the elastic potential as described in Section 7.1 we cannot only compute material specific texture maps, but also find an initial guess for pattern design. To this extent we cut the surface  $S$  along seam paths given by the designer resulting in a set of  $k$  connected surface components  $(S_m)_{m=1\dots k}$ . For many applications a pattern is desired that can be mapped to the given surface with low stress (or equivalently with little physical work). Neglecting friction and bending forces, such patterns can again be found by optimizing the elastic membrane potential. We can thus run the same optimization as in the previous section for each component but without the boundary constraint in Equation 7.1 so that the pattern shapes  $\delta\bar{S}_m$  evolve freely. After convergence we simply extract these boundary curves to find outlines of pattern pieces. The basic assumption of this approach, that the sought after pattern minimizes the elastic potential is in general desirable even though in certain applications other properties as e.g. a prescribed stress pattern are required.

### 7.2.1 Relations to Other Approaches To Pattern Generation

Due to its importance in industrial design, flattening has been studied in computer aided design for nearly thirty years. Except for early approaches, all recent method use strain energies to quantize the unavoidable shape deformation. We concentrate



this discussion on energy based approaches and refer to [AS04] and [YZS07] for earlier approaches.

Interestingly, the edge length potential as proposed by Maillot et al. [MYV93] for parameterization ( see Equation 4.4) has become a popular strain measure for flattening in computer aided design. An optimization of this potential is common to several methods that differ only in the way the initial mapping is found. Azariadis et al. [ANA02] lay out developable strips of triangles in the plane and subsequently close gaps between these strips in the optimizations. McCartney et al. [MHS99] and Wang et al. [WSY02] progressively lay out triangles in a breadth first traversal. Wang et al. [WTY05] trace geodesics on the surface to form a uniform quadrilateral mesh. In contrast to our approach, none of these methods can guarantee a valid initial layout without flips. Zhong and Xu [ZX06] circumvent this problem. They use the physical model of [BW98] and apply bending forces to unfold winged edges before colliding them with a plane. As discussed in Section 4.2.2, the edge length potential is not physically plausible. Moreover, it measures strain only in the direction of edges and thus provides only a rough approximation of the actual deformation behavior, in particular for anisotropic materials.

Only few alternatives for the edge length potential have been proposed in context of pattern generation. The Wirewarp method [Wan08] generates patterns by laying out feature curves on the surface directly. In contrast to the edge length potential, their energy also accounts for differences in geodesic curvature of feature curves on the surface and in the planar domain. McCartney et al. [MHC05] propose a strain energy for woven orthotropic materials. For flattening, triangles are progressively laid out in the planar domain starting from a single seed triangle and new vertices are positioned by locally minimizing the strain energy of adjacent triangles. In contrast to these energies, our strain potential is physically accurate (at least within the material’s elastic range) and is directly derived from elasticity theory. It allows for general anisotropic materials specified by physical parameters that can be determined directly from measurements.

### 7.2.2 Seams

So far the pattern generation method considers each of the connected components  $(S_m)_{m=1\dots k}$  separately and individual patterns  $(\bar{S}_m)_{m=1\dots k}$  for these parts do not influence each other. For assembly, these patches are sewn together along shared boundary segments. As with the current method strain on individual patches can vary, the boundary segments of two patches connected by a seam can also vary significantly in length. This is, however, undesired as it complicates sewing and can lead to wrinkling. We therefore like to impose an additional constraint on the optimization, that enforces equal strain along the seam direction for each pair of adjacent patches. We refer to this constraint as *seam strain constraint*.

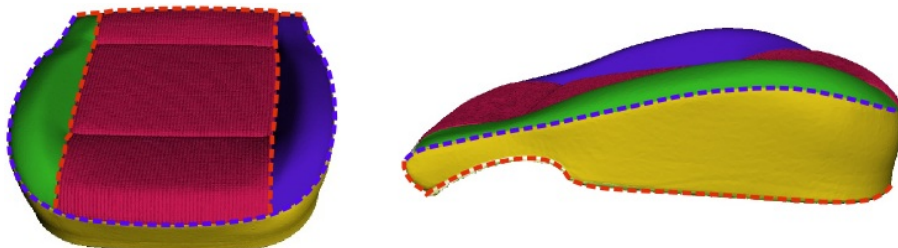


Figure 7.6: Fixed and floating seams on an seat model: Fixed seams (shown in red) are seams between two pattern patches, that are additionally sewn to the upholstery frame. In contrast, a floating seam (in blue) is a seam between patches that is not fixed on the frame and can thus float on the surface. To ensure that floating seams stay in place, we require that membrane forces at each side of the seam cancel exactly.

Besides equal strain along seamed boundaries, it is in some cases also necessary to constrain stress forces at seams. In the absence of such constraints, the stress across a seam between two patches  $S_m$  and  $S_n$  does in general not vanish. During upholstery, the seam will in the absence of other forces thus float on the surface. However, there are ample reasons for the upholsterer to keep them at a well defined position. E.g. for automotive seating the position of seams contributes to the visual and tactile appearance of the seat and is thus specified by the seat designer. To ensure that the position of seams is maintained after upholstery, we introduce a *seam stress constraint*, that requires vanishing stress across seams.

In contrast to the seam strain constraint that is applied to all seams, the seam stress constraint is only applied to a certain type of seam. In general, we distinguish two types of seams which are illustrated in Figure 7.6: *Fixed seams* connect not only two patches of fabric, but are also fixed to the upholstery frame. As the fixation already enforces the designer intended position during upholstery stress constraints are not necessary for this type of seam. In contrast, *floating seams* are not fixed on the frame and can thus float over the surface during upholstery. To maintain their position, we add stress constraints.

To formalize both types of seam constraints we need to extend our notation slightly to represent seams as illustrated in Figure 7.7. We consider two patches  $S_m, S_n$  of the surface  $S$  that share a common boundary represented by a regular curve  $\mathbf{d} : I \subset \mathbb{R} \rightarrow S$ . For simplicity we assume that  $\mathbf{d}$  is parameterized by arc length. Moreover, we continue to assume that  $S$  is parameterized by a single, auxiliary parameterization  $\mathbf{x}$ . Again, this assumption is only made to simplify the discussion and not actually necessary for the optimization algorithm. The

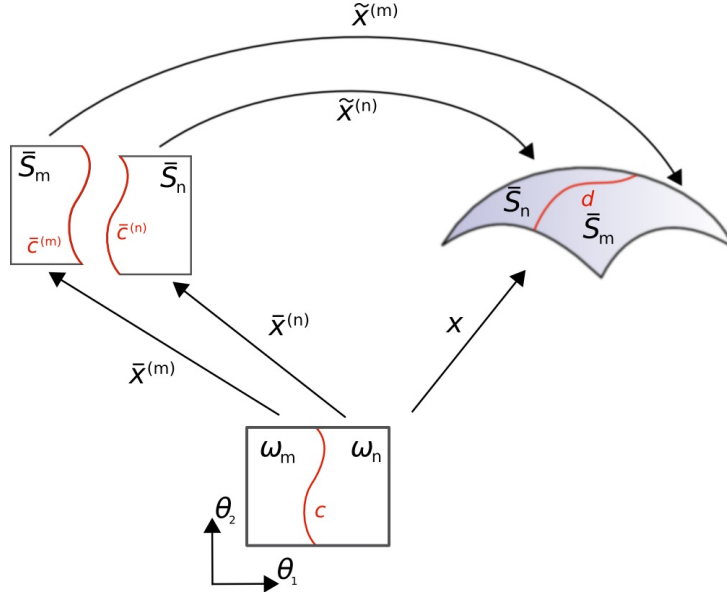


Figure 7.7: Notation for pattern inference in the presence of seams: A seam between two patches  $S_m$  and  $S_n$  is represented by a parametric curve  $d$  on the surface whose preimage is denoted by  $c$ . The two patches are mapped individually by  $\tilde{x}^{(m)}$  and  $\tilde{x}^{(n)}$  to allow for a discontinuity at the seam  $d$ . Consequently, we have for each patch a parameterization ( $\bar{x}^{(m)}$  and  $\bar{x}^{(n)}$ ).

preimage of the seam curve  $d$  under  $x$  is denoted by  $c$ .

To allow discontinuous texture coordinates  $\tilde{x}^{-1}$  at the seam, we replace the parameterization  $\tilde{x}$  and accordingly  $x$  by two continuous maps  $\tilde{x}^{(m)}$ ,  $\tilde{x}^{(n)}$  and  $\bar{x}^{(m)}$ ,  $\bar{x}^{(n)}$  respectively (see Figure 7.7). These give rise to two curves  $\tilde{c}^{(m)} = \tilde{x}^{(m)} \circ c$  and  $\tilde{c}^{(n)} = \tilde{x}^{(n)} \circ c$  that represent those parts of the patch boundaries  $\delta\bar{S}_m, \delta\bar{S}_n$  that are to be stitched together to form the seam.

### Seam Strain Constraint

Using the notation of Figure 7.7, the condition that strain on both sides of the seam matches can be expressed as

$$\left\| \frac{\partial}{\partial t} \tilde{c}^{(m)} \right\|^2 = \left\| \frac{\partial}{\partial t} \tilde{c}^{(n)} \right\|^2 \quad (7.7)$$

i.e. the length of derivatives of the two preimages of  $d$  under  $\tilde{x}$  must match for all  $t \in I$ . For later use, we also define a *seam strain constraint potential* as squared

residual of condition 7.7

$$\begin{aligned} E_{strainconstr.} &= \int_I \left( \left\| \frac{\partial}{\partial t} \bar{\mathbf{c}}^{(m)} \right\|^2 - \left\| \frac{\partial}{\partial t} \bar{\mathbf{c}}^{(n)} \right\|^2 \right) dt \\ &= \int_I \dot{c}^\alpha (\bar{g}_{\alpha\beta}^{(m)} - \bar{g}_{\alpha\beta}^{(n)}) \dot{c}^\beta dt \end{aligned}$$

where  $\dot{c}^\alpha$  denotes the components of the derivative  $\frac{\partial}{\partial t} \mathbf{c}$  in coordinates  $\theta^\alpha$  and  $\bar{\mathbf{g}}^{(m)}$  is the metric tensor of  $\bar{\mathbf{x}}$ .

### Seam Stress Constraint

For the boundary stress constrained we first neglect seams for a moment and consider the general, continuous deformation setup described in Chapter 3. We recall Equation 3.9 that gives the stress force  $\mathbf{t}$  acting on an imaginary surface element with unit normal  $\nu$  inside an elastic solid as

$$t^i = \tau^{ij} \nu_j .$$

For the shell case, we substitute the general stress tensor  $\tau^{ij}$  by the membrane stress tensor  $n^{\alpha\beta}$  given in 7.6 which yields

$$\begin{aligned} t^\alpha &= n^{\alpha\beta} \nu_\beta \\ &= h C^{\alpha\beta\gamma\delta} (g_{\gamma\delta} - \bar{g}_{\gamma\delta}) \end{aligned}$$

and by Equations 7.3-7.5

$$t^\alpha = h c^{\alpha'\beta'\gamma'\delta'} (\tilde{g}_{\gamma'\delta'} - \delta_{\gamma'\delta'}) \frac{\partial \theta^\alpha}{\partial \bar{x}^{\alpha'}} \frac{\partial \theta^\beta}{\partial \bar{x}^{\beta'}} \nu_\beta$$

In the above equation, the stress vector  $\mathbf{t}$  is given by *contravariant* coordinates  $t^\alpha$ , that refer to the coordinate frame  $\mathbf{a}_i$ , i.e.  $\mathbf{t} = t^\alpha \mathbf{a}_\alpha$ . In contrast, the coordinates  $\nu_\beta$  of  $\nu$  are *covariant*, i.e. understood with respect to the system  $\mathbf{a}^i$ . It can be easily verified, that co- and contravariant coordinates are related by the metric tensor, more precisely

$$\nu_\alpha = \bar{g}_{\alpha\beta} \nu^\beta$$

and using this relation the above expression for the stress force can be rewritten as

$$t^\alpha = h c^{\alpha'\beta'\gamma'\delta'} (\tilde{g}_{\gamma'\delta'} - \delta_{\gamma'\delta'}) \frac{\partial \theta^\alpha}{\partial \bar{x}^{\alpha'}} \frac{\partial \theta^\beta}{\partial \bar{x}^{\beta'}} \bar{g}_{\kappa\beta} \nu^\kappa \quad (7.8)$$

$$= h c^{\alpha'\beta'\gamma'\delta'} (\tilde{g}_{\gamma'\delta'} - \delta_{\gamma'\delta'}) \frac{\partial \theta^\alpha}{\partial \bar{x}^{\alpha'}} \frac{\partial \bar{x}^{\beta'}}{\partial \theta^\kappa} \nu^\kappa \quad (7.9)$$

In the presence of multiple patches  $S_m$  and seams, we obtain a similar expression for the stress forces in each patch:

$$(t^{(m)})^\alpha = hc^{\alpha'\beta'\gamma'\delta'} (\bar{g}_{\gamma'\delta'}^{(m)} - \delta_{\gamma'\delta'}) \frac{\partial\theta^\alpha}{\partial(\bar{x}^{(m)})^{\alpha'}} \frac{\partial(\bar{x}^{(m)})^{\beta'}}{\partial\theta^\kappa} \nu^\kappa \quad (7.10)$$

To find stress forces at the seam, we choose  $\nu = \nu(t)$  as a normal vector field to the curve  $\mathbf{c}$ . More precisely, as we are interested in the stress in the final deformed configuration  $S$ , the vector field is chosen to be normal to the curve in the metric  $\mathbf{g}$ . I.e. in each point  $\mathbf{c}(t)$ , the coordinates  $\nu^\alpha(t)$  are chosen such that

$$\nu^\alpha g_{\alpha\beta} \dot{\mathbf{c}}^\beta = 0 \quad \text{and} \quad \nu^\alpha g_{\alpha\beta} \nu^\beta = 1$$

(mapping the thus obtained vector field using  $\nabla_{\mathbf{x}}$  in fact yields a normal vector field to the curve  $\mathbf{d}$  on  $S$ ). Using the above relation, we then find a field of stress forces  $\mathbf{t}(t)$  along the seam curve  $\mathbf{c}$ . At each point on the curve, the force vector  $\mathbf{t}^{(m)}$  can be split into a force acting tangential to the seam and an orthogonal force acting across the seam, i.e. along the normal direction  $\nu$ . For the position of the seam on the surface only the latter component is important — the so called *normal stress* — whose magnitude is the dot product  $\mathbf{t}^{(m)} \cdot \nu$ . Using the metric  $\bar{\mathbf{g}}^{(m)}$  and Equation 7.10 we get

$$\begin{aligned} \mathbf{t}^{(m)} \cdot \nu &= (t^{(m)})^\alpha \bar{g}_{\alpha\lambda}^{(m)} \nu^\lambda \\ &= hc^{\alpha\beta\gamma\delta} (\bar{g}_{\gamma\delta}^{(m)} - \delta_{\gamma\delta}) \frac{\partial(\bar{x}^{(m)})^\alpha}{\partial\theta^\lambda} \frac{\partial(\bar{x}^{(m)})^\beta}{\partial\theta^\kappa} \nu^\kappa \nu^\lambda \end{aligned}$$

Requiring, that a seam between patches  $S_m$  and  $S_n$  stays in place after upholstering is equivalent to demand canceling normal stress in each point along the seam, i.e.

$$\mathbf{t}^{(m)} \cdot \nu = -\mathbf{t}^{(n)} \cdot \nu \quad (7.11)$$

so that after sewing, the sum of both stresses vanishes. Similar to the strain boundary constraint, we also define a *stress constraint potential* as the squared residual

$$E_{stressconstr.} = \int_I (\mathbf{t}^{(m)} \cdot \nu + \mathbf{t}^{(n)} \cdot \nu)^2 dt$$

### Optimizing Patterns

As argued in the introduction, in the absence of seam constraints patterns can be computed simply by a minimization of the elastic potential. In this case, for example the hierarchical optimization method of Section 6.1.2 can be used to infer pattern shapes.

However, in the presence of seams the optimization problem becomes more complicated. Here also pattern shapes are sought that can be mapped with minimal elastic potential. But in addition, the seam constraints must be satisfied in the equilibrium configuration, after upholstering when the fabric has “settled”. For a given patterns  $\delta\bar{S}_m$ , the equilibrium is characterized by a vanishing potential gradient in the patch interior. More precisely,

$$\partial_{\bar{\mathbf{x}}^{(m)}} E_{membrane}[\bar{\mathbf{x}}^{(m)}](\bar{\mathbf{y}}^{(m)}) = 0 \quad (7.12)$$

for all patches  $\bar{S}_m$  and variations  $\bar{\mathbf{y}}^{(m)}$  that vanish on the boundary, i.e.

$$\bar{\mathbf{y}}^{(m)}|_{\delta\omega_m} = 0$$

The pattern inference problem with seams can thus be put as the following constrained optimization: Find rest state configurations  $(\bar{\mathbf{x}}^{(m)})_{m=1\dots k}$  that minimize  $E_{membrane}$  subject to the equilibrium condition 7.12 as well as the seam conditions 7.7 and 7.11. Please note, that the additional equilibrium condition is in fact necessary to ensure, that strain and stress values in the formulation of the seam constraints actually correspond to the physical equilibrium configuration that is assumed after sewing and upholstering the pattern. If it is omitted, for any vertex  $v$  we have by the Lagrange multiplier rule

$$\frac{\partial E_{membrane}}{\partial \bar{\mathbf{x}}^v} = -\lambda_v \frac{\partial E_{constr}}{\partial \bar{\mathbf{x}}^v} \quad (7.13)$$

where  $\lambda_v$  denotes the Lagrange multiplier and  $E_{constr}$  some function that vanishes iff the seam constraints are met. In particular, the gradient does in general not vanish for inner vertices. In this case, the optimizing configuration will thus not reflect the physical equilibrium.

Unfortunately, the above constrained optimization problem is rather complex and even the search for an admissible initialization is computationally involved. We thus suggest here a simple approximation scheme that is based on the optimization algorithm developed for surface parameterization. To this extent, we first replace the seam constraints in the above problem statement by so called soft constraints, which amounts to minimizing

$$E := E_{membrane} + \alpha E_{strainconstr.} + \beta E_{stressconstr.}$$

subject to the equilibrium condition 7.12 only. As the seam constraint potentials are defined as the sums of squared residuals, the optimization will strive to satisfy both constraints of both types in least squares sense. The parameters  $\alpha$  and  $\beta$  must be chosen relatively large to penalize deviation from the seam constrain. Using soft constrains an admissible initialization must only satisfy the equilibrium constraint and can thus be found using the pattern inference that ignores seams.

Second, we approximate the constraint optimization by alternating an unconstrained optimization of patch boundaries  $\delta\bar{S}_m$  and a separate optimization of inner points  $\bar{S}_m \setminus \delta\bar{S}_m$ . The complete optimization algorithm is sketched in Algorithm 1. While the first step optimizes patch boundaries to conform with the constraints and to reduce the elastic potential, the second step only reestablishes the equilibrium configuration in each iteration. The suggested simple approximation

---

**Algorithm 1** The pattern inference algorithm for patterns with seams.

---

Initialization:

find  $\bar{\mathbf{x}}^{(m)}$  as minimum of  $E_{membrane}[\bar{\mathbf{x}}^{(m)}]$  for all patches  $m = 1 \dots k$

**repeat**

  step 1 (optimize boundary): optimize  $E$  with respect to patch boundaries  $\bar{\mathbf{x}}^{(m)}|_{\delta\omega_m}$  for all patches.

  step 2 (reestablish equilibrium): optimize  $E_{membrane}$  with respect to inner points  $\bar{\mathbf{x}}^{(m)}|_{\omega_m \setminus \delta\omega_m}$  for all patches.

**until** convergence of  $\bar{\mathbf{x}}^{(m)}$

---

is easy to implement and in our experiments we always observed convergence to an admissible configuration satisfying both seam and equilibrium constraint even though we do not give a theoretical guarantee here.

## 7.3 Results

To analyze the visual impact of material parameters on the texture map, we first considered an artificial example surface shown in Figure 7.8. A rectangular texture image with several concentric arcs was mapped on the surface using various parameter sets: Two parameter sets were obtained from measurements of woven cotton fabrics and were extracted from [WAY03]. Isotropic refers to a perfectly isotropic fabric. We also added three artificial materials with different levels of anisotropy in Young’s modulus for comparison. Although, for knitted textiles, levels of anisotropy of 1 : 5 and higher have been reported, a relative difference of 1 : 100 or higher is up to our knowledge not realistic for materials commonly used on fabric covered surfaces. These materials were added to visualize the effect of anisotropy.

As shown in the figure, differences become noticeable even for the two cotton fabrics whose elastic behavior is relatively similar. It is most of all the level of anisotropy that influences the shape of the concentric arcs. While the relative difference in Young’s modulus is about 1 : 1.2 for cotton 1, the second cotton material is slightly more anisotropic with a relative difference of 1 : 2.2. As



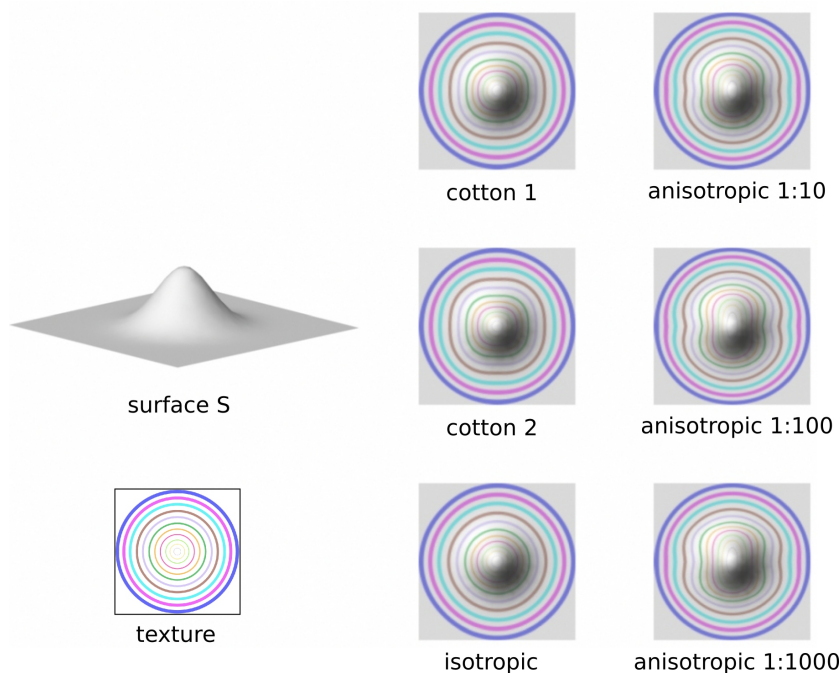


Figure 7.8: Material specific texture maps for an artificial surface. Cotton 1 and Cotton 2 refer to measured woven fabrics. Anisotropic refers to an artificial anisotropic fabric with different levels of anisotropy in the Young’s modulus.

a result arcs appear more roundish for cotton 1. For the isotropic material the circles are perfectly preserved. The observed effects become more pronounced with increasing anisotropy. In contrast, differences in the texture maps that are due to changes of the Poisson’s ratio were hardly noticeable.

Figure 7.9 shows texture maps computed for a human torso generated by a statistical model based on a database of body scans [HSS<sup>+</sup>09]. The shown surface covers only front facing part of the torso. We first computed an optimal pattern for the database average shown in Figure 7.9a using our pattern inference method (with isotropic material settings). For this, the boundary of the model was used as a single seam. Due to the nature of our pattern inference the resulting strain on the mean model is very low. As shown in the figure, differences in the texture map caused by material changes are hardly visible. However, when using the same pattern on other models (Figure 7.9b-d) differences are clearly noticeable in regions of high strain like hip, chest and belly.

In case of the larger masculine model (Fig. 7.9d), there are visible artifacts around the shoulders. In fact, the compressive strain on this model is very high.



For high strains as in this case, our linear stress/strain approximation is no longer valid as discussed in Section 7.1 and the optimization results in gap artifacts. The limitation of our model to moderate strain is also demonstrated in Figure 7.10. Here a non-developable surface with high surface area to boundary length ratio was chosen to provoke gap artifacts. As shown in the figure, gaps appear more often and more pronounced in the anisotropic case. In the isotropic case, the fattened surface  $\bar{S}$  resembles the result obtained by minimizing the length potential  $E_{length}$  discussed in Section 5.3.

To examine the practicability of the suggested pattern generation we put it to work on an automotive seat model. For comparison a car seat was upholstered and digitized using a range scanner. The resulting surface (shown in Figures 7.6 and 7.2) was cut along seams and we first applied our method to infer patterns for each connected components separately ignoring seams. Figure 7.11 shows the resulting patterns for different materials and a comparison with the actual pattern (see Figure 7.2) used in the upholstery. Please note, that in contrast to our results, the reference pattern includes an extra seam margin at the boundary (By the time of writing the exact size of the margin for the reference data set was, unfortunately, unknown. The dotted line shown in Figure 7.11 is based on a hypothetical value and was added for illustration.). Except for an isotropic material, the orientation of the planar pattern  $\bar{S}$  is not arbitrary. As we impose no orientation constraint, it is rather rotated by the optimization so that strain is mostly concentrated along the axis of smallest stiffness. For comparison, each pattern piece was manually aligned with the reference by rotation and translation.

The comparison shows a high compliance for the patch in the middle (shown in red). Only the pattern for the anisotropic material differs which is slightly compressed along the diagonal (that coincides with the direction of minimal stiffness). The remaining pieces involve higher strain and patterns differ considerably. Although, unfortunately, the material parameters of the original fabric were not available for our comparison, but at least the results for the isotropic and cotton materials are very similar (In fact, the difference between cotton 1 and cotton 2 is not noticeable and only the cotton 1 is shown). Again, differences become more pronounced with increasing anisotropy. However, in case of the green patch, all patterns generated with the elastic potential clearly deviate from the reference pattern.

The effect of seam constraints is demonstrated in Figure 7.12. At the time of writing a complete specification of seam types was unfortunately not available. The seam types assumed for the presented results are indicated in Figure 7.6. For this result we run 3000 iterations of the approximation algorithm which effectively enforces both types of seam conditions as visible in the figure. The effect of seam

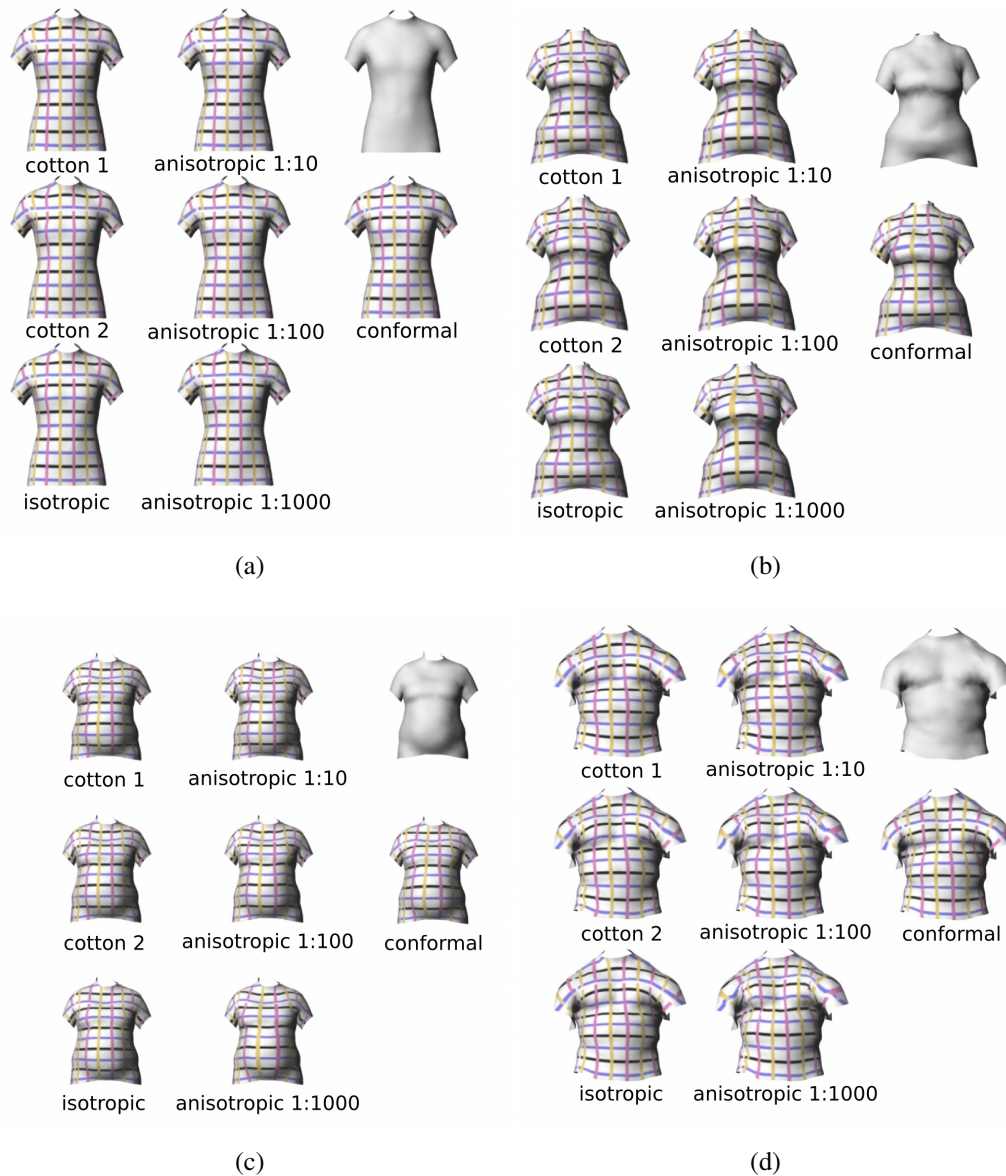


Figure 7.9: Material specific texture maps on a human torso: (a) Results for the mean torso for different materials. A pattern for an optimal fitting shirt was computed for this model. Consequently strain is very low on this model and differences due to material changes are hardly visible. (b) Using the same pattern on a feminine torso results in more strain and differences become noticeable (e.g. for the cotton materials on the hip). The same is true for the pregnant torso model shown in (c) where effects are apparent at the belly. (d) A larger masculine body model. Strain on this model is very extreme and artifacts due to the linear stress/strain relation become visible at the shoulders.

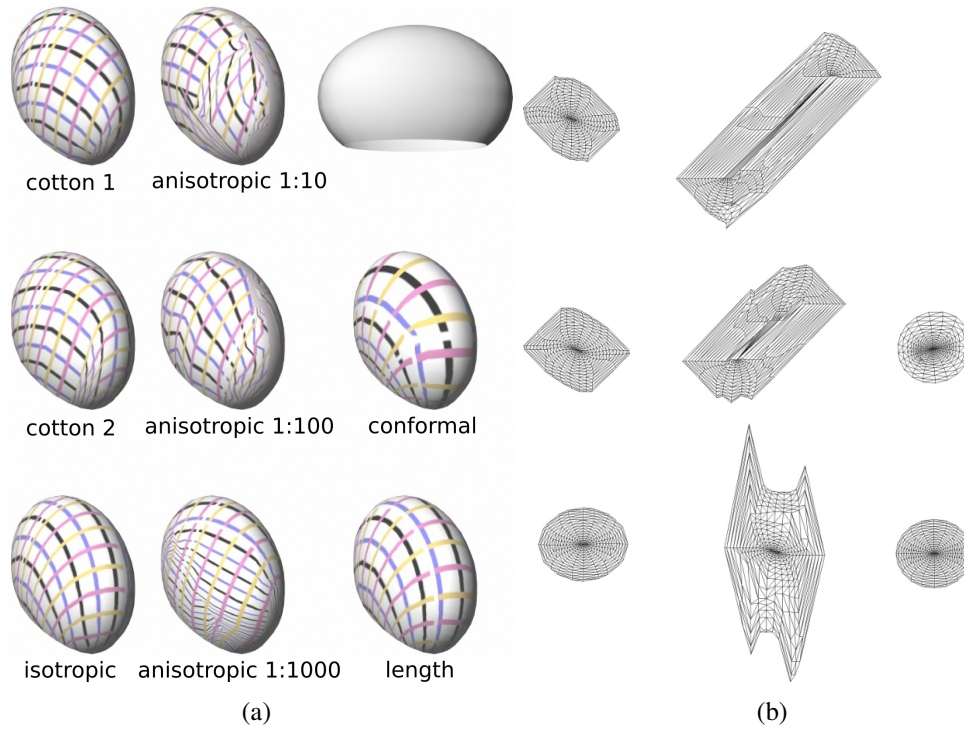


Figure 7.10: Limitations of the linear elastic model: Flattening this artificial surface results in very high strain. In this situation the linear stress/strain assumption no longer holds. As a result our method produces gap artifacts that become more pronounced with increasing material anisotropy. In (a) the texture mapped surfaces  $S$  are shown while (b) shows the corresponding flattened surfaces  $\bar{S}$ . Minimizers of the conformal potential and  $E_{length}$  are shown for comparison.

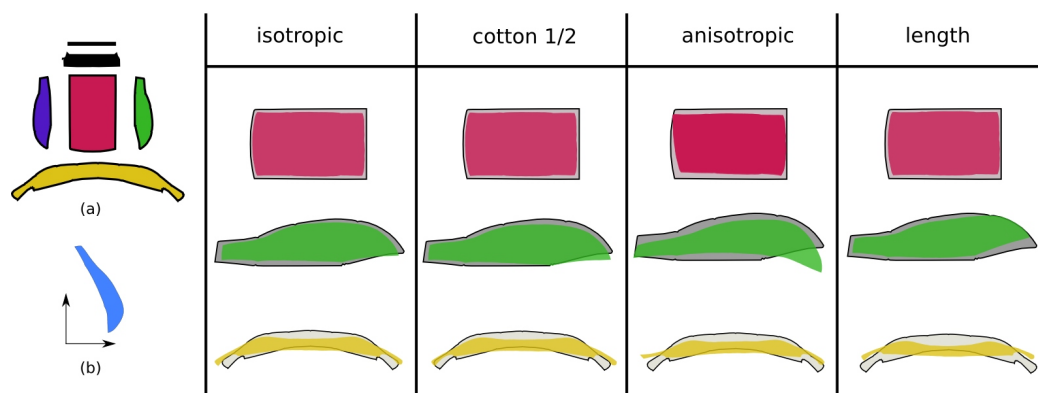


Figure 7.11: Comparison of generated patterns for the seat model with the actual reference pattern (shown in (a) and in gray in the table). Patch colors correspond to those in Figure 7.6. The table shows results obtained by optimizing the respective potentials without boundary constraint. Cotton 1 and cotton 2 result in nearly identical patterns. Anisotropic refers to the artificial anisotropic material with relative Young’s modulus difference of 1 : 10. For comparison, the result using  $E_{length}$  is also shown. (b) Optimization results in a specific orientation of the patch in warp and weft directions. In the table, all patches were aligned manually by a rigid transformation.

constraints on the pattern shape is rather subtle and local as shown in the Figure 7.12c. However, the compliance with the reference pattern slightly improves.

## 7.4 Discussion

Both the results on the artificial surface and on the torso model demonstrate, that material parameters can have significant influence on the visual appearance. This influence increases with material anisotropy and larger strain. While material specific effects might not be noticeable for some textures, they become apparent for textures dominated by strong regular patterns. In the latter case, the impact of the texture map on the appearance is drastic as shown in Figure 7.1. We therefore advocate the use of material specific texture maps for such materials in particular in applications where visual realism is crucial. We also like to point out, that material specific texture maps complement generalized texture functions as e.g. *bidirectional texture functions* which have recently become commonplace in computer graphics. While bidirectional texture functions capture photometric material properties, our material specific texture maps account for elastic properties.

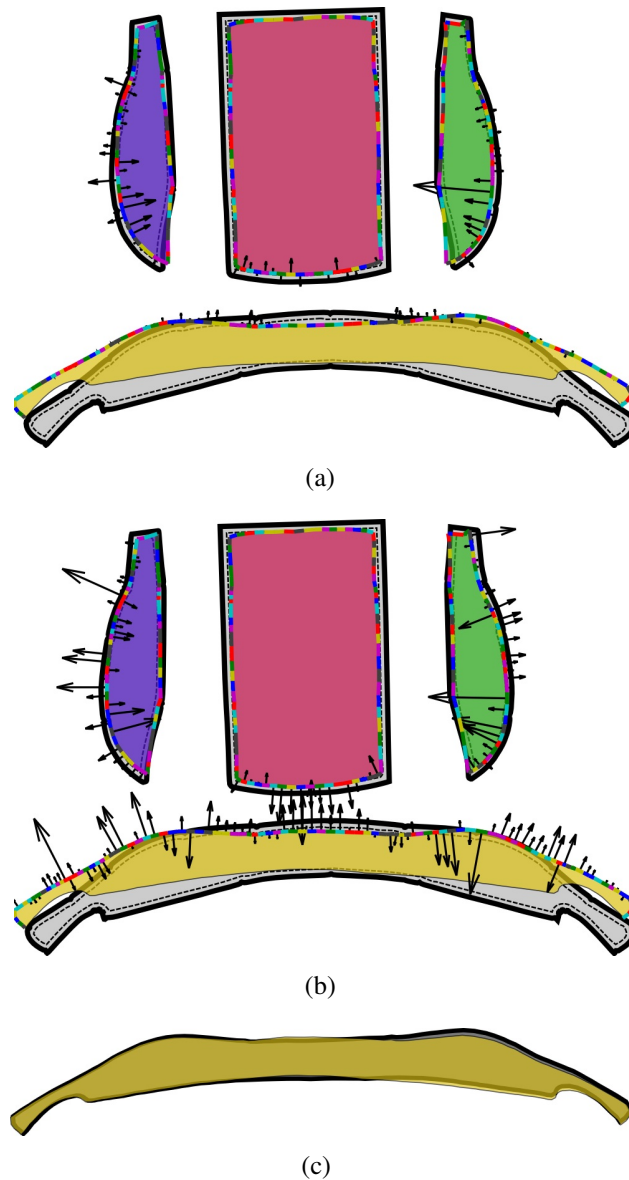


Figure 7.12: Effect of seam constrains on the pattern. (a) pattern inferred for an isotropic material without seam constrains. The dotted line indicates an additional (hypothetical) seam margin of the reference pattern. (b) pattern inferred with seam constrains. The original reference pattern is underlying in gray. Corresponding seam segments are colored with identical colors to visualize strain along the seam and normal stresses are shown on seam segments assumed as floating. From these visualizations it becomes clear that our pattern inference with seams succeeded in matching boundary strain and normal stresses. Changes in the pattern shape are visualized in (c) where the result without seams is underlying in gray.

Ideally photometric and elastic material parameters should be chosen consistently to achieve high degrees of realism.

Albeit our results are promising, we see several possible improvements. Even on the torso model strain becomes large enough to violate the linear stress/strain assumption. A natural extension is therefore an elastic potential based on a non-linear model (like the neo-hookean or Mooney-Rivlin model [Cia88]) that would render the proposed method more accurate and robust in the presence of high strain. Moreover, a comparison of our texture maps with strain measurements is pending.

Concerning the proposed pattern generations, preparations for an evaluation in automotive seating design are currently underway. Even if the preliminary comparison given in the last section reveals considerable deviations from the reference pattern, our results might serve as a good starting point for the iterative manual design and save some of the necessary iterations. Apart from that, we conjecture that pattern compliance improves with a more accurate non-linear stress/strain model. We also like to explore the use of additional potentials for pattern inference. While it is reasonable to require small strain on patterns, there are ample reasons to consider further optimizations targets. E.g. for upholstery the absence of wrinkles is often crucial. As wrinkles appear under the influence of compressive strain but not with tensile strain, it is worth considering an asymmetric potential that penalized compressive strain more rigorously than tensile strain.



---

## COMPUTING PARAMETERIZATIONS FOR INCONSISTENT MESHES AND POINT CLOUDS

---

Parameterizations obtained by minimizing the potentials discussed in Chapter 5 are of high quality and in particular show low shape deformation as required in most computer graphics applications. However, as explained in Section 4.1.1 a surface triangulation must have certain consistency properties to support parameterization over a planar domain. In particular, it must be 2-manifold, consistently oriented and topologically equivalent to a disk.

Moreover, small gaps in the surface (see Figure 8.2a ) can pose a problem. Often such gaps are artifacts of triangulation algorithms and do not correspond to actual geometric holes in the surface but exist only in the triangulation. While a parameterization using the discussed algorithms is technically possible, the resulting map is not continuous across the gap. For texture mapping this results in



Figure 8.1: A CAD model with non-manifold edges and gaps (marked yellow). Our method can create texture maps for such inconsistent triangulations that are smooth at gaps and boundaries.

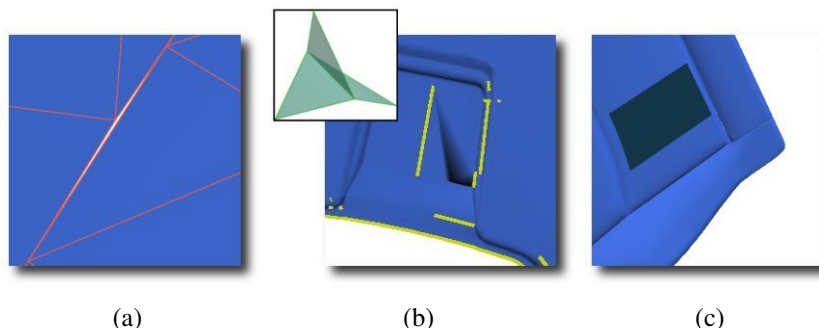


Figure 8.2: Three types of triangulation artifacts that hinder parameterization: (a) small gaps in the triangulation can cause visible discontinuities in the texture. (b) non-manifold edges with more than two adjacent faces. (c) inconsistent face orientation.

a visible discontinuity in the texture which might be undesirable. Therefore, surface triangulation should ideally be “watertight” meaning that border edges in the triangulation exist only at geometric surface boundaries.

In day-to-day practice models of different quality and origin have to be processed that often fail to meet these triangulation consistency requirements. For example computer aided design models converted from NURBS representations for downstream applications like simulation or visualization frequently consist of unconnected patches and are thus not watertight. But also non-manifold edges and inconsistent face orientations (see Figure 8.2) are frequently encountered triangulation artifacts.

To deal with inconsistent surface representations surface modelers have to resort to semi-interactive texture mapping tools based on classical two-part mapping as introduced by Bier and Sloan [BS86]. In two part mapping the surface is embedded into a box, sphere or cylinder and texture coordinates are assigned by certain simple projections. Though no constraints are posed on the representation a tedious choice and placement of proxy objects is required to produce good results.

In this chapter we propose a texture mapping method for inconsistent meshes inspired by classical two-part mapping that proceeds in three steps (see Figure 8.3): First, a proxy surface with the above mentioned consistency properties is created automatically as an iso-surface of the distance function. Second, a parameterization is generated for this proxy using state-of-the-art charting tools and the parameterization method proposed in the last chapter. Finally, a novel projection method is applied to map the texture signal of the proxy to the original geometry.



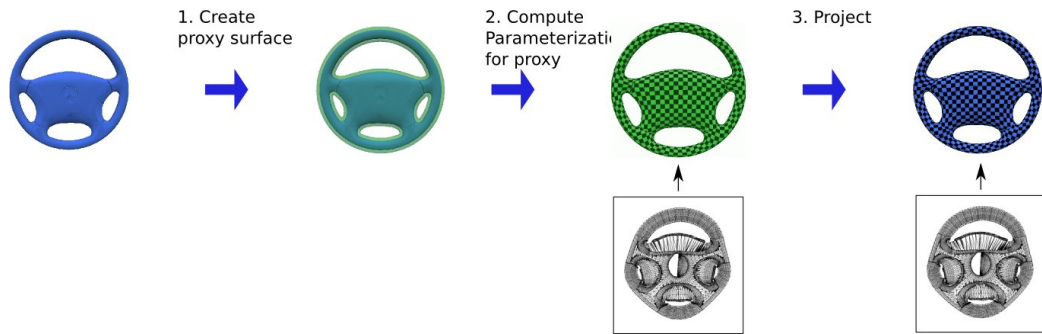


Figure 8.3: Our method proceeds in three steps: 1. Create a proxy surface for a given mesh by offsetting. 2. Parameterize the proxy surface. 3. Project original surface onto proxy. A texture map is then obtained by concatenating the proxy parameterization and the projection.

To circumvent manifoldness and orientation problems, our projection actually maps onto a double cover of the original surface, i.e. it creates texture coordinates for both sides of each triangle. While parameterizations of the double cover have a slightly more limited applicability than parameterizations of the surface itself, they are well suited for texture mapping which is one of the most important applications of parameterizations. An example of a texture map created with our approach is shown in Figure 8.1.

Our method requires no user interaction for neither proxy placement nor mapping and handles triangulation degenerations of all three types. Moreover, it leaves the original geometry untouched and thus all geometric features are perfectly preserved. Our approach can also be applied to point sets which enables the application of standard texture atlas generation tools to point sampled geometry.

While we use the combined parameterization potential discussed in chapter 5, our approach is in principle not restricted to a particular parameterization potential. In the remainder of this chapter we summarize the most important aspects of our method and refer to [DK07] for more details.

## 8.1 Related Work

For inconsistent meshes repairing methods have been designed that either fix or completely substitute the flawed triangulation. Surface completion methods go even further by closing larger holes or missing parts of a surface. Both kinds of approaches are clearly related to our concern and we will briefly comment on them here.

**Mesh repairing** For triangle meshes, an alternative to our approach is to apply mesh repairing methods to the input mesh to fix the aforementioned inconsistencies. These methods can be roughly divided into two categories: Local approaches like [BK05, BNK02, Lie03, GTLH98] try to fix inconsistencies by local remeshing operations and preserve as much of the original triangulation as possible. The triangle quality is, however, not significantly improved. Subsequent parameterization can thus be slow or might encounter numerical difficulties. In contrast volumetric approaches as [HK06, BPK05, SOS04, NT03, Ju04, GPRJ00] convert the mesh into a volumetric representation and extract a new surface from that. However, for models showing several types of inconsistency the reconstruction often fails to reproduce sharp features and boundaries or heavily oversamples the surface. Although it also extracts a proxy surface from a volumetric representation, our approach preserves the original geometry and thus circumvents reconstruction problems. While most volumetric approaches try to build a signed implicit representation, there are some methods like e.g. [HK06], that use unsigned functions. The latter is advantageous if surface orientation is inconsistent or unreliable. Similarly, we construct our proxy as a double cover of the original surface to avoid orientation problems.

**Surface completion** Our method builds on a novel projection method that is used to map a texture signal from a proxy surface onto an inconsistent mesh. Conceptually similar, many surface completion methods [ASK<sup>+</sup>05, KS05, ACP03, SP04, KHYS02, HSC02] establish a map between an incomplete mesh and a template surface to fill holes or missing parts in a surface. Except for [HSC02], all these methods need a set of correspondences between feature points on both surfaces. In particular, if the inconsistent mesh consists of many disconnected components (as it can be often observed in meshes triangulated from NURBS) the number of required correspondences and thus the amount of necessary user interaction is very high. In [HSC02], Hilton et al. fit a template mesh to a point set and use normal-volume mapping to define a map between the two surfaces. In Section 8.4 we will see that projecting along normals does not lead to a well-defined mapping for our choice of a proxy surface.

**Point sets** Only few methods exist that create texture maps for point sets. Floater and Reimers proposed in [FR01] a meshless parameterization which assumes that points are sampled from single surface patch. Their method has been generalized to spherical objects [ZG04] and to genus one surfaces [TGG05]. Furthermore, methods from reverse engineering for point set parameterization exist [WK95, Aza04, PT01] which also assume that points are sampled from a B-spline patch. Zwicker et al. [ZPKG02] developed a parameterization for point sets that can extrapolate texture boundaries. It requires manual charts layout and specification of at least two constraints per chart.

**Other related approaches** To texture implicit surfaces, Zonenschein et al. pro-

posed in [ZGVdF97] to trace particles through the implicit function’s gradient field. While we use a similar idea to establish a map between surface and proxy, we define a smooth guidance field that is independent of the proxy’s implicit representation and respects the geometry of both proxy and original surface.

In [JSW05] Ju et al. generalize mean value coordinates [Flo03] to 3D tetrahedral meshes. They use these coordinates to interpolate 2D texture coordinates of a solid surrounding tetrahedral meshes to a 2D surface. Though the surrounding solid mesh is somewhat similar to our proxy, it has to be designed manually and projection properties heavily depend on its layout. To avoid interpolation artifacts at texture boundaries, Tarini et al. [THCM04] wrap the surface into a poly cube and define a mapping from the poly-cube onto the surface. The poly-cube is designed manually and further user interaction is required to specify 3D texture coordinates within it. Nevertheless, the poly-cube is conceptually similar to our proxy.

## 8.2 General Setup

The notation for this chapter is summarized in Figure 8.4. We denote the original surface by  $\hat{S}$  and assume it is given by a triangulation  $\hat{\mathcal{M}} = (\hat{V}, \hat{\mathcal{T}})$  which is not necessarily manifold, oriented or connected. Alternatively, it can be given as a discrete set of points with associated normals. In the first case we associate a vertex position  $\mathbf{p}^{\hat{v}} \in \mathbb{R}^3$  with each vertex. We also use  $(\mathbf{p}^{\hat{v}})_{\hat{v} \in \hat{V}}$  to denote the point set in the latter case and associated normals are denoted by  $(\mathbf{n}^{\hat{v}})_{\hat{v} \in \hat{V}}$ . The proxy surface will be denoted by  $S$  and is given by an oriented and manifold triangulation  $\mathcal{M} = (V, \mathcal{T})$  with vertex positions  $(\mathbf{x}^v)_{v \in V}$ . As defined in Section 4.1.1, we will denote its parameterization over the planar domain  $\tilde{S} \subset \mathbb{R}^2$  by  $\tilde{\mathbf{x}}$ .

For texture mapping we are interested in a texture map  $\hat{\mathbf{x}}^{-1} : \hat{S} \rightarrow \tilde{S}$  for  $\hat{S}$ . After defining an appropriate projection  $\pi : \hat{S} \rightarrow S$  it can be computed as  $\hat{\mathbf{x}}^{-1} = \tilde{\mathbf{x}}^{-1} \circ \pi$ , where  $\tilde{\mathbf{x}}$  is computed by applying the parameterization algorithm described in Chapter 6 to a suitable consistent proxy surface  $S$ . The projection  $\pi$  computed in this chapter is not necessarily bijective and thus the texture map  $\hat{\mathbf{x}}^{-1}$  cannot be inverted to give a valid parameterization  $\hat{\mathbf{x}}$ . However, we focus in this chapter primarily on texture mapping applications which rely only on  $\hat{\mathbf{x}}^{-1}$ . Our method produces texture maps of high visual quality even if it does not guarantee bijectivity.

To deal with non-manifold edges and inconsistent face orientations we consider the surface  $\hat{S}$  as double sided. For each triangle  $T \in \mathcal{T}$  we add an oppositely oriented triangle to the triangulation (For point sets, we add for each point a point with the opposite normal). We then map each triangle separately to the texture

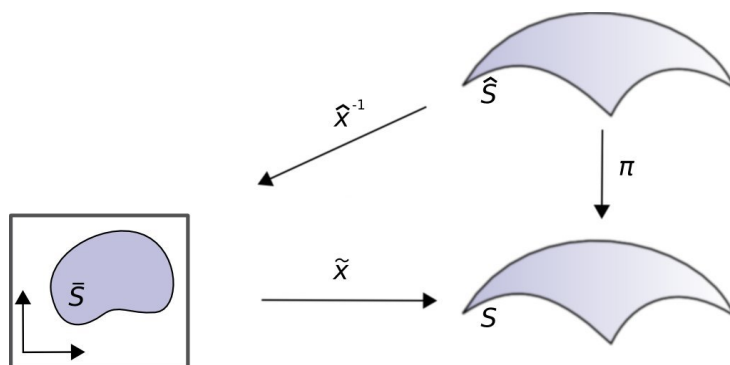


Figure 8.4: Notation for this chapter: For the inconsistent triangle mesh  $\hat{S}$  a consistent proxy surface  $S$  is computed. The latter is parameterized by  $\tilde{x}$  over  $\bar{S} \subset \mathbb{R}^2$ . Using an appropriate projection  $\pi$  a texture map  $\hat{x}^{-1}$  for  $\hat{S}$  can be computed as  $\hat{x}^{-1} = \tilde{x}^{-1} \circ \pi$ .

domain (by assigning texture coordinates to pairs of vertices and faces instead to vertices directly. Such coordinates are called *per face texture coordinates*). During rendering, the back facing triangles will be culled automatically. Our projection ensures that the correct texture map for each orientation of a triangle is chosen and that the texture map is smooth across gaps and edges.

### 8.3 Generating the Proxy Surface

The shape deformation induced by the projection  $\pi$  depends on the distance and similarity between surface and proxy geometry. Ideal results are obtained if both surfaces are identical. On the other hand, the proxy surface should have all properties that enable and facilitate parameterization, i.e. manifoldness, watertightness, consistent orientation, and nicely shaped triangles.

We define our proxy surface as an iso-surface of the unsigned distance function. This has several advantages: It is easy and fast to compute via standard contouring methods and is naturally manifold and oriented. Furthermore, it can be extracted at arbitrary small distances of the original surface. Its extraction does not require a signed distance field, whose computation is often problematic especially in the presence of inconsistent orientations, holes and non-manifold edges. Moreover, iso-surfaces of the unsigned distance function wrap the geometry from both sides and thus actually provides an approximation for a double cover of the surface  $\hat{S}$ . The creation of the proxy surface is shown in Figure 8.5.

For a user-specified distance  $d$  we extract the iso-surface using a marching cubes implementation that enforces the correct manifold topology. The resulting

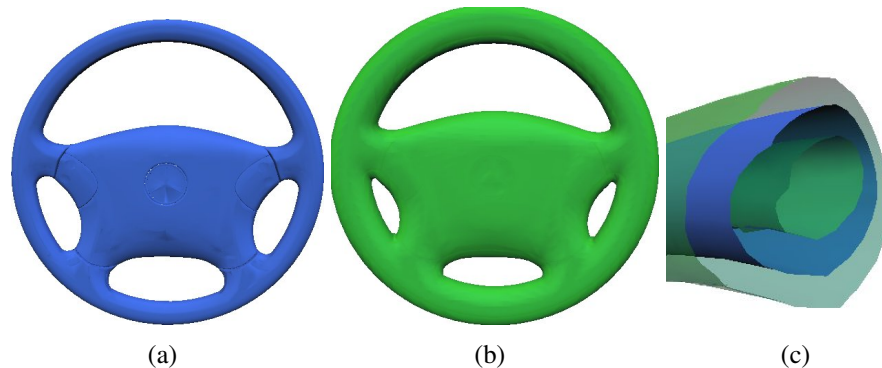


Figure 8.5: Construction of the proxy surface: (a) original surface  $\hat{S}$ . (b) proxy surface  $S$  computed as iso-surface of the (unsigned) distance function. (c) a cut through both surfaces. The proxy  $S$  wraps the original surface  $\hat{S}$  from both sides.

triangulated surface is then cut into chart of disk topology using the method in [LPRM02b]. To speed up parameterization and to enhance robustness we decimate and smooth the resulting triangulation prior to computing the parameterization  $\tilde{x}$  for each chart.

## 8.4 Mapping between Proxy and Surface

In context of two-part mapping, three simple choices for the projection  $\pi$  have been proposed: a) projection towards the proxy object center, b) along the normals of the proxy and c) along the normals of the original surface. Unfortunately, all three choices have certain drawbacks:

- Projection towards the center is only suitable for very simple objects. A straight forward generalization is to project each point on the surface onto its closest point on the proxy. (This yields the same mapping for sphere proxies). In the general case, however, there are ambiguities which cannot easily be resolved (see Fig.8.6a).
- The same is true for projection along the normals of the proxy surface (see Figure 8.6c).
- Projecting along normals of the original surface can result in an undesired mapping (see Figure 8.6b).

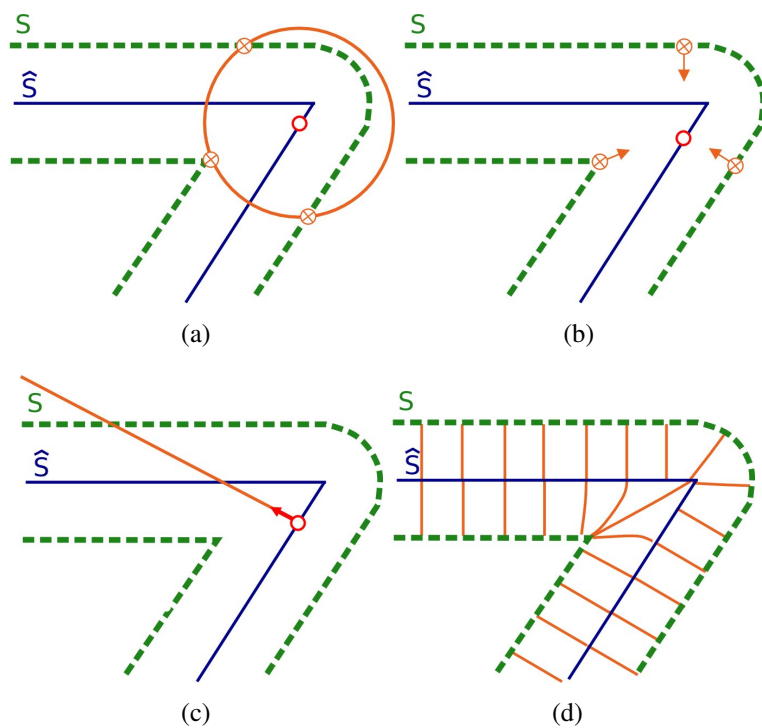


Figure 8.6: Choices for the projection operator  $\pi$ : (a) mapping onto closest points. (b) Projection along the normals of  $S$  and (c) along normals of  $\hat{S}$ . (d) mapping along electric field lines

Because of these problems, we propose a slightly more elaborate projection operator: We assume for a moment that for each point  $\hat{\mathbf{p}} \in \hat{S}$  the a surface normal is given. To define the projection of  $\hat{\mathbf{p}}$ , we emit a particle in the direction of the normal and trace its way through a guidance field until it hits the proxy  $S$  in a point  $\mathbf{p} \in S$ . We then set  $\pi(\hat{\mathbf{p}}) = \mathbf{p}$ . A simple choice for the guidance field is the gradient field of the distance function. However, the distance function is not smooth and its gradient field is discontinuous which complicates stream line tracing. Furthermore, stream lines of the distance function's gradient field can meet in the space between original and proxy surface and the map  $\pi$  degenerates at these points.

To define a smooth field between source and proxy surface we therefore use the following approach which is physically motivated: We apply electric charges of the same magnitude with opposite signs to  $\hat{S}$  and  $S$ . If  $\hat{S}$  has a positive charge, we trace for each  $\hat{\mathbf{p}} \in \hat{S}$  the path of a positively charged zero-mass particle through the electric field until it hits the negatively charged proxy surface. Paths of particles then correspond to electric field lines. In contrast to the distance function the electric field is smooth everywhere between  $\hat{S}$  and  $S$ . The projection defined in this way leads to a correct mapping even in the special case shown in Figure 8.6.

Since the electric field is only defined in the space between  $\hat{S}$  and  $S$  and not on the surfaces itself, we cannot emit particles in a point  $\hat{\mathbf{p}} \in \hat{S}$  itself but have to offset it in the direction of its normal ( we choose  $\hat{\mathbf{p}} + \epsilon * \hat{\mathbf{n}}$  as a start point, where the offset  $\epsilon$  can be chosen arbitrary small). Choosing starting points in this way introduces a dependency on the surface normals of  $S$  and can lead to high shape deformation in the projection  $\pi$  if the provided normals are incorrect or noisy. For our method we suggest a simple normal reestimation procedure for inconsistent meshes that is detailed in [DK07].

### 8.4.1 Electric Field of Points and Triangles

In order to trace the field lines that define  $\pi$  we must be able to evaluate the electric field at any point in the space between  $\hat{S}$  and  $S$ . The electric field at a point  $\mathbf{p} \in \mathbb{R}^3$  is given as

$$\mathbf{E}(\mathbf{p}) = \int \rho(\mathbf{p}') \frac{\mathbf{p} - \mathbf{p}'}{\|\mathbf{p} - \mathbf{p}'\|^3} d\mathbf{p}' \quad (8.1)$$

where  $\rho(\mathbf{p}')$  denotes the density of the charge in the point  $\mathbf{p}'$ . If charge density functions  $\rho^{\hat{T}}$  and  $\rho^T$  are given for triangles  $\hat{T} \in \hat{\mathcal{T}}$  and  $T \in \mathcal{T}$  we can write

$$\mathbf{E} = \sum_{\hat{T} \in \hat{\mathcal{T}}} \mathbf{E}^{\hat{T}} + \sum_{T \in \mathcal{T}} \mathbf{E}^T$$

where  $\mathbf{E}^{\hat{T}}$  and  $\mathbf{E}^T$  are the fields resulting from a single triangle  $\hat{T}$  or  $T$ . In case of a point set, we will define charge density function  $\rho^{\hat{v}}$  for each point  $\hat{\mathbf{p}}^{\hat{v}}$  and Equation 8.1 becomes:

$$\mathbf{E} = \sum_{\hat{v} \in \hat{V}} \mathbf{E}^{\hat{v}} + \sum_{T \in \mathcal{T}} \mathbf{E}^T$$

In case of a point sampled surface  $\hat{S}$  we put a point charge in each point  $\hat{\mathbf{p}}^{\hat{v}}$  and thus choose  $\rho^{\hat{v}}$  as

$$\rho^{\hat{v}}(\mathbf{p}) = \delta(\|\mathbf{p} - \hat{\mathbf{p}}^{\hat{v}}\|)$$

where  $\delta$  denotes the Dirac delta distribution. This simple choice gives good results if the sample density is roughly uniform on the surface  $S$ . This is a common assumption for point sampled geometry in computer graphics and holds e.g. for surfaces acquired by laser range scanning.<sup>1</sup> Substituting this simple density in Equation 8.1 yields the radial field

$$\mathbf{E}^{\hat{v}}(\mathbf{p}) = (\mathbf{p} - \hat{\mathbf{p}}^{\hat{v}}) / \|\mathbf{p} - \hat{\mathbf{p}}^{\hat{v}}\|^3.$$

Unlike in the case of point sets, the vertices of  $\hat{\mathcal{M}}$  sample the surface  $\hat{S}$  only sparsely and irregularly. Placing point charges in the vertices does not lead to a good approximation of a field induced by a uniform charge density on  $\hat{S}$ . For meshes, we thus refrain from an approximation and set

$$\rho^{\hat{T}}(\mathbf{p}) = -\delta_{\hat{T}_{\hat{S}}}(\mathbf{p}) \quad (8.2)$$

where  $\delta_{\hat{T}_{\hat{S}}}$  is a generalized delta distribution satisfying

$$\int_{\mathbb{R}^3} \delta_{\hat{T}_{\hat{S}}}(\mathbf{p}) f(\mathbf{p}) dx = \int_{\hat{T}_{\hat{S}}} f(\mathbf{p}) d\hat{S} \quad (8.3)$$

for real valued integrable functions  $f$ . A closed form of the resulting field  $\mathbf{E}^{\hat{T}}$  is given in the next section.

### Closed Form for the Potential of a Triangle

While the evaluation of the electric field of a triangle is computationally more expensive than a simple radial field, we were able to derive an analytic expression. We briefly sketch the derivation in the following: We consider a general triangle

---

<sup>1</sup>Alternatively, it is possible to scale the charge density by some local density estimate or even to distribute the charge over a small disc centered in  $\hat{\mathbf{p}}^{\hat{v}}$ . However, the additional computational effort is usually not justified.



$\Delta = (\mathbf{p}^1 \mathbf{p}^2 \mathbf{p}^3)$  spanned by points  $\mathbf{p}^i \in \mathbf{R}^3$ . Instead of its electric field we will derive a closed form expression for its electric potential:

$$P_{\Delta}(\mathbf{p}) = \int \delta_{\Delta}(\mathbf{p}') \|\mathbf{p} - \mathbf{p}'\|^{-1} d\mathbf{p}'$$

The electric field can then be obtained as its gradient field  $\mathbf{E}_{\Delta} = -\partial P_{\Delta} / \partial \mathbf{p}$ . Using Equation 8.3 we can write this as a surface integral

$$P_{\Delta}(\mathbf{p}) = \int_{\Delta} \|\mathbf{p} - \mathbf{p}'\|^{-1} d\Delta$$

As the area element  $dS$  is invariant with respect to a rotation  $\mathbf{R}$  and translation  $\mathbf{t}$  of  $S$ , we have

$$P_{\Delta}(\mathbf{p}) = P_{\mathbf{R}\Delta + \mathbf{t}}(\mathbf{R}\mathbf{p} + \mathbf{t})$$

By choosing  $\mathbf{R}$  and  $\mathbf{t}$  appropriately,  $\Delta$  can be rotated into the plane  $\theta^3 = 0$ . It is therefore sufficient to consider the only special case  $p_1 = p_2 = 0$  and  $p_3^i = 0$  for  $i = 1 \dots 3$ . The general case is then obtained by choosing  $\mathbf{R}$  and  $\mathbf{t}$  appropriately. For the special case, the potential evaluates to

$$P_{\Delta}(\mathbf{p}) = \int_{\Delta} \|(\theta^1, \theta^2, 0)^t - (0, 0, p_3)^t\|^{-1} d\theta^1 d\theta^2$$

To apply Gauss's theorem, we define a planar vector field  $\mathbf{f}$  as

$$f_{\alpha} := \frac{\sqrt{(\theta^1)^2 + (\theta^2)^2 + (p_3)^2}}{(\theta^1)^2 + (\theta^2)^2}.$$

whose divergence equals the integrand and write

$$P_{\Delta}(\mathbf{p}) = \int_{\Delta} \operatorname{div} \mathbf{f} d\theta^1 d\theta^2 = \sum_{i=0}^3 \int_{\mathbf{p}^i \mathbf{p}^{i+1}} \mathbf{f} \cdot (\mathbf{n}^i)^t ds \quad (8.4)$$

where  $\mathbf{n}^i$  denotes the outward normal on the edge  $\mathbf{p}^i \mathbf{p}^{i+1}$ . Finally, the line integral in Eq. 8.4 is found (using computer algebra tools) as

$$\int_{\mathbf{p}^i \mathbf{p}^{i+1}} \mathbf{f} \cdot (\mathbf{n}_i)^t ds = \left[ p_3 \tan^{-1} \left( \frac{p_3 t}{u_i \sqrt{(p_3)^2 + u_i^2 + t^2}} \right) + u_i \log \left( t + \sqrt{(p_3)^2 + u_i^2 + t^2} \right) \right]_{t=v_i}^{t=v_{i+1}}$$

with

$$u_i = \mathbf{n}^i \cdot (\mathbf{p} - \mathbf{p}^i)^t \quad v_i = \frac{\mathbf{p}^{i+1} - \mathbf{p}^i}{\|\mathbf{p}^{i+1} - \mathbf{p}^i\|} \cdot (\mathbf{p}^i - \mathbf{p})^t.$$

A small complication arises if  $\mathbf{0} \in \Delta$  as the field  $\mathbf{f}$  has a singularity in  $\mathbf{0}$ . In this case, we have to exclude the singularity from the integration in order to apply Gauss's theorem for the second equality in Equation 8.4. To this extend we remove a small ball  $B_\epsilon(\mathbf{0})$  of radius  $\epsilon$  around the origin from the integration domain  $\Delta$ . However, it can be easily verified that

$$\lim_{\epsilon \rightarrow 0} \int_{\partial B_\epsilon} \mathbf{f} \cdot \mathbf{n}_{\partial B_\epsilon}^t ds = 2\pi \sqrt{(p_3)^2}$$

If  $\mathbf{0} \in \Delta$  Equation 8.4 therefore becomes

$$P_\Delta(\mathbf{p}) = \sum_{i=0}^3 \int_{\mathbf{p}^i \mathbf{p}^{i+1}} \mathbf{f} \cdot (\mathbf{n}^i)^t ds - 2\pi \sqrt{(p_3)^2}$$

## 8.4.2 Projection for Point Sets

To compute the projection  $\pi$  in case the surface  $\hat{S}$  is given as a point set is now straightforward. With an evaluation scheme of the electric field at hand, we can apply numerical integration to trace a field line for each point  $\hat{\mathbf{p}}^{\hat{v}}$  and find its intersection with the proxy  $S$ .

Evaluating the electric field is, however, very costly since we must sum the contributions of all primitives which becomes infeasible even on small sized models. Fortunately, while the influence of an electric charge is not compactly supported, it drops rapidly with the square of the distance and it is sufficient to consider only primitives in a neighborhood of the actual tracing position. Moreover, for densely sampled point sets field lines of neighboring points often happen to run nearly parallel to each other to the proxy surface. This fact can be exploited to further reduce the cost by selectively omitting the field line tracing. For details on the accelerated field line tracing we refer to [DK07].

Guided by the electric field particles nearly always find their way to the proxy geometry. As long as the distance of the iso surface is chosen properly, exceptions only occur on hardly visible parts of the surface  $\hat{S}$ . Such a case is shown in Figure 8.7. We will comment on the choice of the iso-surface distance in Section 8.5.

## 8.4.3 Projection for Triangulations

To compute the projection  $\pi$  for a inconsistent triangulation field lines can then be traced to find texture coordinates for every vertex. For efficient rendering, texture

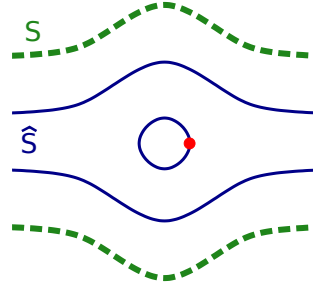


Figure 8.7: A problematic case where field lines do not find their way to the proxy. Here an internal connected component of  $\hat{S}$  is cut off from the proxy  $S$ . If the distance of the iso surface is properly chosen, such problems occur only in hardly visible regions of  $\hat{S}$ .

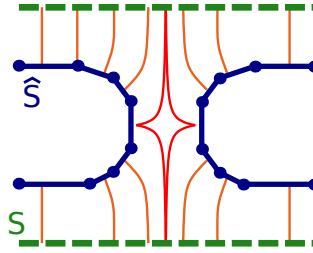


Figure 8.8: A discontinuity in the map  $\pi$  due to a tunnel in the surface  $\hat{S}$ . As the tunnel is smaller than the isosurface distance, the  $S$  does not cover  $\hat{S}$  from both sides around the tunnel.

maps on triangle meshes are usually required to be affine on each triangle. Thus, it has to be ensured that the final texture map  $\hat{\mathbf{x}}^{-1}$  is adequately approximated by an affine interpolation of the vertex texture coordinates across triangles of  $\hat{M}$ . This is certainly only possible, if the final texture map  $\pi$  is nearly affine on the triangles of  $S$ . While both the parameterization  $\tilde{\mathbf{x}}$  and the projection  $\pi$  are smooth on large parts of the surface, problems might occur on large triangles of  $\hat{S}$  and at exceptional points as chart boundaries of  $\tilde{\mathbf{x}}$  or holes or tunnels in  $\hat{S}$  smaller than the isosurface distance (see Figure 8.8). In both cases the resulting combined map  $\hat{\mathbf{x}}^{-1}$  is discontinuous. Since a piecewise affine map on the triangulation of  $\hat{S}$  can only be discontinuous at edges, such discontinuities must be detected and triangles must be split to ensure that  $\hat{\mathbf{x}}^{-1}$  is continuous inside each triangle. We found that the following heuristic works well in practice: After tracing field lines at vertices, we compute for each edge in  $\hat{S}$  the ratio

$$\hat{d}_{vw} := \|\hat{\mathbf{x}}^{-1}(\hat{\mathbf{p}}^{\hat{v}}) - \hat{\mathbf{x}}^{-1}(\hat{\mathbf{p}}^{\hat{w}})\| / \|\hat{\mathbf{p}}^{\hat{v}} - \hat{\mathbf{p}}^{\hat{w}}\|$$

which is the directional stretch of the linear interpolation on the edge  $(\hat{v}, \hat{w})$ . We compare this value with the average directional derivative of the inverse parameterization  $\tilde{\mathbf{x}}^{-1}$  in the points  $\pi(\hat{\mathbf{p}}^{\hat{v}})$  and  $\pi(\hat{\mathbf{p}}^{\hat{w}})$  in the direction of the edge denoted by  $d_{vw}$ . Now, if  $\hat{\mathbf{x}}^{-1}$  is continuous on the edge the derivative of the projection  $\pi$  is around one and  $\hat{d}_{vw}$  should roughly equal  $d_{vw}$ .

In contrast, if  $\hat{\mathbf{x}}^{-1}$  is not continuous along the edge  $(v, w)$  there are two cases: In the first case  $\hat{\mathbf{p}}^{\hat{v}}$  and  $\hat{\mathbf{p}}^{\hat{w}}$  map to different sides of a texture seam and the distance  $\|\hat{\mathbf{x}}^{-1}(\hat{\mathbf{p}}^{\hat{v}}) - \hat{\mathbf{x}}^{-1}(\hat{\mathbf{p}}^{\hat{w}})\|$  is large. In the second case,  $\pi$  maps the two points to different “sides” on the proxy (see Figure 8.8). In this case the geodesic distance of their images  $\pi(\hat{\mathbf{p}}^{\hat{v}})$  and  $\pi(\hat{\mathbf{p}}^{\hat{w}})$  on  $S$  is very large and  $\|\hat{\mathbf{x}}^{-1}(\hat{\mathbf{p}}^{\hat{v}}) - \hat{\mathbf{x}}^{-1}(\hat{\mathbf{p}}^{\hat{w}})\|$  is large as well (assuming an roughly isometric  $\tilde{\mathbf{x}}$ ). In both cases we have  $\hat{d}_{vw} \gg d_{vw}$ . Therefore, we consider  $\hat{\mathbf{x}}^{-1}$  as continuous on an edge if  $|\hat{d}_{vw} - d_{vw}| < \epsilon$ . As the increase of  $\hat{d}_{vw}$  is rather drastic for a discontinuous edge the choice of the threshold  $\epsilon$  is uncritical.

If a discontinuity is detected the edge and adjacent triangles are subdivided and the test is repeated for the resulting fragments. For details on the subdivision procedure we again refer to [DK07]. We note, however, that the algorithm only uses planar subdivision of faces, i.e. the geometry and features of  $\hat{S}$  are perfectly preserved. As already mentioned, it happens that a field line does not reach the proxy surface so that  $\pi$  is undefined for some vertices. These cases are also handled by the algorithm which assigns invalid texture coordinates after a number of subdivisions.

## 8.5 Evaluation and Results

The shape deformation of the parameterization depends heavily on a suitable choice of the proxy distance. Clearly, smaller distances result in a better approximation of  $\hat{S}$  and less shape deformation. On the other hand, smaller distances also increase the complexity of the triangulation of  $S$  (number of faces and vertices) and consequently increases processing time (most notably the time for optimizing  $\tilde{\mathbf{x}}$ ). We propose to initially choose a large distance and then to gradually decrease it if necessary. Usually a quick visual inspection of the unparameterized proxy  $S$  is sufficient to decide if a proxy covers all important features. By interactively visualizing iso-surfaces we were able to determine a suitable value in a few minutes for all tested models. We applied our method to both point sets and inconsistent triangle meshes to demonstrate its practicability and the quality of resulting texture maps. The results obtained for point sets are shown in Figure 8.10. By far the largest part of the preprocessing time is spent on the parameterization of the proxy

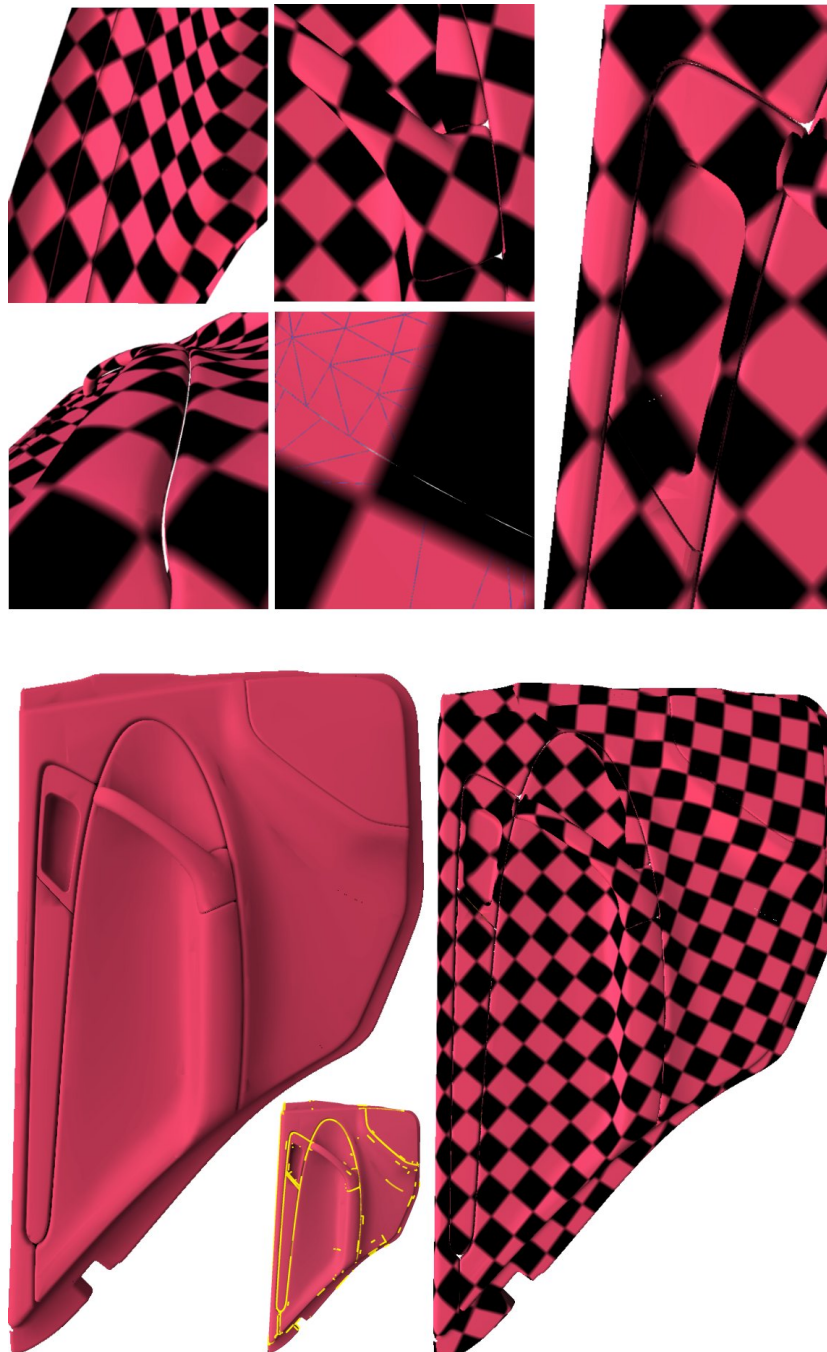


Figure 8.9: Texture maps for the car door model. Middel left: boundary edges are marked yellow (underlying model courtesy of Daimler Chrysler AG)

model	# vert.	# vert. proxy	Timings(seconds)	
			preprocessing (% param.)	projection
Point sets				
face (not shown)	41k	135k	143(69%)	44
ball joint	137k	281k	663(85%)	122
rocker arm	186k	84k	494(90%)	348
Meshes				
handle (not shown)	0.9k	1.6k	75(90%)	7
car door	8k	243k	2208(96%)	365

Table 8.1: Processing times for the models shown obtained on a dual Xeon 2.4 Ghz CPU.

surface as shown in Table 8.1 so that the projection takes less than 20 percent of the processing time on average. For consistently oriented point or triangle sets it is possible to select and remove interior components of the proxy manually which roughly halves the processing time. However, for the timings listed in table 8.1 the automatically generated proxy was not modified.

We also applied our method to real world data sets extracted from an industrial CAD model of a car. The seat model and the car door model shown in figure 8.1 and Figure 8.9 were triangulated from their original NURBS representation. During the triangulation individual NURBS patches were not or only partially stitched which leads to a high number of gaps in the surface (these gaps are marked yellow in the figures). Additionally, they contain a high amount of narrow or degenerated triangles as well as non-manifold edges. As demonstrated in the figures, the resulting texture maps are of high visual quality and the texture signal is smooth even across gaps and narrow triangles. Texture seams are handled by the subdivision algorithm and the corresponding discontinuity is very well represented in the adaptively subdivided meshes (see top of Figure 8.11).

Depending on the type of the model a specific texture seam layout might be more or less important. E.g. for the seat model a designer usually wants to enforce a particular seam layout which automatic charting methods cannot produce. The user has certainly the option to alter the seam layout if it is unsatisfactory. In case of the seat texture seams were manually specified on the proxy for the four connected components and each component was processed individually. On the other hand, the complex car door model was processed as a whole without any manual modification. We think that the ability to process surfaces with inconsistencies automatically is an important advantage of our method in day to day modeling



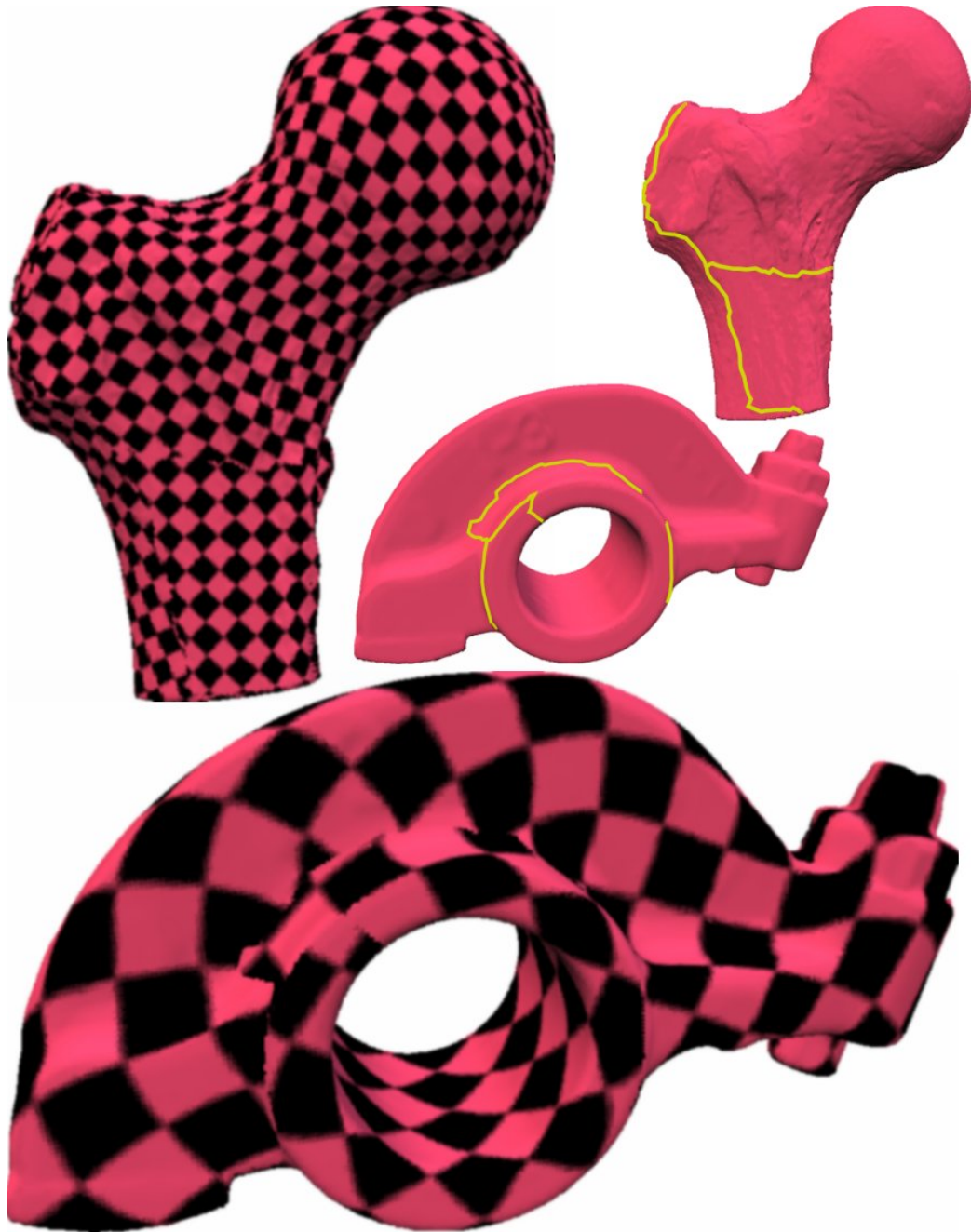


Figure 8.10: Texture maps for point sets (texture seams are marked yellow)

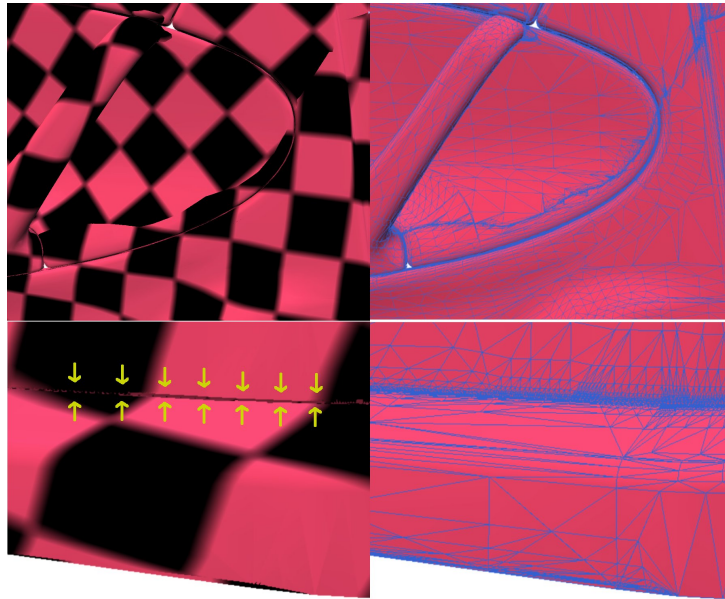


Figure 8.11: From left to right: projected texture near seam, triangles near seam, texture and wire frame near discontinuity

practice.

### 8.5.1 Conclusion

In this chapter an approach was proposed that allows the application of parameterization and charting tools developed for consistent triangle meshes to surface triangulations showing several types of inconsistencies as gaps, degenerated triangles or non-manifold edges. In contrast to mesh repairing methods, the original geometry is not altered and thus features are perfectly preserved. The same approach can also be applied to point sets. Although a suitable proxy distance error must be manually specified, no further manual intervention is necessary. Compared to that, a manual repair or texture atlas creation takes considerably more time for industrial CAD models. We therefore believe, that our method is of high practical relevance.

It would be nice to eliminate necessary user interaction completely by replacing the manual proxy distance choice with an automatic method that takes geometric properties like local feature size and visibility into account. Such a method could adaptively choose smaller distances near features while keeping the overall complexity of the proxy low.



## **Part III**

# **Interactive Deformation Potentials**



## CHAPTER 9

---

### INTRODUCTION

---

In Part II we have seen how optimal surface parameterizations can be computed by minimizing a deformation potential. The potential was designed to measure membrane strain of a parameterization that maps from flat parametric domain  $\bar{S}$  onto a surface  $S$ . In this chapter, we consider the more general case of maps  $\tilde{x}$  between two curved surfaces  $\bar{S}$  and  $S$  in three space (see Figure 3.2). Analogous to the flat case, we search for measures that quantify the deformation imposed by a given map  $\tilde{x}$ .

In this more general setting it is, of course, possible to apply the potentials derived in Part I to the deformation  $\tilde{x}$  (more specifically to its metric tensor). As outlined there, these potentials bear some similarity with the elastic membrane potential. While these quantify membrane strain i.e. tangential deformation, changes in curvature between the two surfaces are not fully captured: for example, a plane can be mapped onto a cylinder by an isometry, i.e. with  $\tilde{g} = \mathbf{id}$ . A deformation potential solely expressed in terms of the metric tensor is thus invariant with respect to the above described isometric deformation as it induces no membrane strain, even though it clearly alters shape and curvature of the surface. If we are interested in the extent to which shape is deformed, a bending potential is therefore necessary to consider changes in curvatures. Mathematically, such potentials are functions of the second fundamental form.

While deformation potentials on surfaces are relevant to many computer graphics applications we focus our attention on interactive shape editing applications which have gained much attention in recent years [BS08, SB09]. In these applications, the user can interactively edit large high resolution meshes as obtained from scanning real-world objects. To this extent, individual parts of the surface can be grabbed and moved while remaining unconstrained surface parts deform interactively according to a deformation model (see Figure 1.2).

Contrary to other editing methods, surface deformation based approaches are highly intuitive giving the user the impression of pushing or pulling a thin object made of an elastic material as e.g. rubber. Therefore, they allow even inexperi-

enced users to apply complex edits to large scanned models which is one of the reasons for their success.

In order to handle such large meshes interactively, in these approaches rather simple deformation potentials are used that support very fast minimization. Discretizing the elastic shell potential described in Section 3.2.1, in contrast, leads to a complex non-linear optimization problem, severely limiting mesh resolutions. But also the discretization as such is a problem on unstructured triangle meshes. The elastic bending potential involves second order derivatives which are not directly available on linear finite elements. Numerical problems can also result from often poor triangle quality of meshes as they are typically obtained from range scanning. But apart from these technical problems a physical deformation potential also restricts shape modeling to physically possible operations. Simple desirable operations as scaling surface parts are beyond these limits and thus not realizable. For interactive editing, strict *physical accuracy* is therefore abandoned in favor of higher frame rates and greater modeling freedom even though some methods [BPGK06a] aim at an deformation behaviour that is still at least *physically plausible*.

Summing up, for shape editing deformation potentials must meet two important requirements: First, they must support an efficient optimization to enable interactive frame rates on large models and, second, they must be well-defined for triangle meshes as the predominant representation for scanned objects.

In this part of the thesis we concentrate on deformation potentials that satisfy these properties. We will give an account on the discrete deformation potentials on meshes in this subclass and will then propose a novel deformation potential suitable for mesh editing applications which is physically plausible and can imitate the behaviour of different materials. The proposed potential is non-linear, but we demonstrate how it can efficiently be optimized on graphics hardware by a simple tailored optimization algorithm. Finally, we describe a further application of interactive deformation potentials in the visualization of upholstery in industrial design. In this context we introduce force field constraints that can be effectively used to simulate cushioning and fit a deformable mesh to point samples.

---

## PREVIOUS WORK ON DEFORMABLE MODELS FOR SHAPE EDITING

---

### 10.1 General Approaches to Shape Editing

Shape editing has been an extremely active field ever since the early beginnings of computer graphics and accordingly a variety of approaches exist reaching from classical splines to multiresolution techniques and space deformations. We will give only a brief overview over the principal approaches here and refer for a more general review of shape editing to [BPR<sup>+</sup>06] and [SB09].

**Tensor product splines** are nowadays the prevailing representation for surfaces in computer aided design. In many applications surfaces are created from scratch in this representation. Spline basis functions have a number of desirable properties, e.g. local support and positive partition of unity that give linear coefficients — the control points — an intuitive interpretation and thus simplify subsequent editing. Conversion of large unstructured point clouds or triangle meshes as acquired by scanning devices or photometric stereo into a spline based representation is, however, difficult and still an active topic of research. Large or complex surfaces usually require tensor product spline patches with several hundreds of degrees of freedom. Except for changes to small details, editing of such surfaces thus involves modifications of many control points and is therefore often tedious and troublesome.

To overcome restrictions of tensor product spline surfaces, a second class of approaches [BKS03, PKKG03, ZRKS05] is based on what is called **transformation propagation**. The user initializes editing by selecting a subset of the surface as region-of-interest  $R$ . Within this region, she then selects two further subset: a fixed subset  $F$  and a handle subset  $H$ . The points in the handle subset  $H$  are then transformed interactively by some user specified transformation (usually translation, rotation and scaling). The deformable model interactively computes positions for the remaining surface points in the region-of-interest. This general

editing metaphor introduced in [BKS03] and [BK04] is also used in shape editing based on deformation potentials that will be discussed below. In transformation propagation approaches, the handle transformation is propagated through the regions of interest until it reaches the fixed subset  $F$ . Each point within  $R$  is transformed with an interpolation of the handle transformation and the original fixed transformation at  $F$  according to some function of its distance to these regions. As demonstrated in [SB09] transformation propagation does not necessarily lead to intuitive deformations.

**Multiresolution deformation** techniques decompose the original surface into a low-pass filtered coarse approximation and high-frequency details. The actual parameters of the filter can be configured by the user to select the level-of-detail of her interest. Modifications are then applied to the coarse version of the surface e.g. by transformation propagation or an arbitrary other editing technique. Finally, the stored high-frequency details are added on top of the edited version of the coarse approximation. Approaches differ first of all in the way, high frequency detail is represented and fused with the edited coarse mesh.

In all surface-based deformation techniques the quality of deformations is inherently linked to mesh quality. Problems in the triangulation like cracks or degenerate triangles inevitable lead to deformation artifacts. Furthermore, the speed of such methods decreases with the mesh resolution so that very high mesh resolutions result in non-interactive frame rates. **Space deformation** methods avoid these problems by deforming the space surrounding the object. The embedded surface is deformed by applying this space deformation to every point. Space deformations can for example be defined by tensor product splines, radial basis functions or specially designed cages around the surface in question. The complexity of the deformation then only depends on the complexity of the control structure, e.g. the number of control points or cage cells. As the actual surface mesh is not involved in the computations, space deformation approaches handle meshing problems gracefully and can equally be applied to point sets. On the downside, special care must be taken to ensure sufficient resolution and correct topology of control structures.

Finally, another class of approaches minimizes surface-based **deformation potentials** for shape editing. For user interaction the same general editing metaphor is used as with transformation propagation based editing. In contrast to transformation propagation, the unconstrained surface within the region-of-interest is updated by minimizing a potential functional on the surface. Possible choices of the potential functional include linear approximations of the elastic energy discussed in Section 3.2.1. Apart from these, mesh editing methods based on differential representations as they have become very popular in the last years also naturally lead to deformation potentials and can therefore be subsumed into this class of approaches. Deformation potentials will be discussed in more detail in the following

section.

## 10.2 Shape Editing with Deformation Potentials

Deformation potentials have first been introduced to computer graphics by Terzopoulos et al. [TPBF87] to compute animations of elastic surfaces and bodies. Even though their method does not aim at interactivity, they use a simplified elastic potential to keep computations feasible with the following density function

$$e = k_s^{\alpha\beta} \alpha_{\alpha\beta}^2 + k_b^{\alpha\beta} \beta_{\alpha\beta}^2 .$$

$\alpha$  and  $\beta$  denote differences in the first and second fundamental forms as defined in Equation 3.11 in Section 3.2.1. The elasticity tensor  $H^{ijkl}$  is replaced by two matrix norms weighted by some material constants  $k_s^{ij}$  and  $k_b^{ij}$ . Moreover, in their method, the second term in this elastic potential is linearized to simplify computations further.

For interactive freeform design, Welch and Witkin [WW92] as well as Celniker and Gossard [CG91] simplified this potential even more by approximating the first and second fundamental forms by norms of partial derivatives of the parameterization:

$$e = k_s \sum_{\alpha=1}^2 \|\mathbf{x}_{,\alpha}\|^2 + k_b \sum_{\alpha,\beta=1}^2 \|\mathbf{x}_{,\alpha\beta}\|^2 .$$

This potential was also used in the context of interactive surface modeling by Kobbelt et al. [KCVS98] within a multi resolution framework. However, in the above form, it is not invariant with respect to changes in the parameterization  $\mathbf{x}$  and also captures distortions induced by the parameterization. To avoid this, Botsch and Kobbelt [BK04] considered the parameterization of  $S$  over  $\bar{S}$ , i.e. they substituted  $\tilde{\mathbf{x}}$  for  $\mathbf{x}$  in the above potential. The first term in the potential is then the Dirichlet energy of the surface and thus coincides in case of a conformal parameterization with the surface area. Minimizing this energy for relatively small deformations thus gives surfaces of minimal area. The second term coincides with the so called *thin spline energy* [Duc77] that captures surface bending so that its minimization leads to  $C^1$  continuous surface deformations [BK04]. In interactive editing, optimizing potentials of the above form therefore replace the original surface within the user-specified region-of-interest with a surface of minimal area and bending. However, this means that surface detail is in general lost during editing and must be added back by some other method, e.g. by adding high frequency displacements in a multi resolution framework.

Alternatively, details can be preserved to some extent by applying the above simplified potential to local displacements  $\mathbf{d} = \mathbf{x} - \bar{\mathbf{x}}$  instead of the deformation

$\tilde{\mathbf{x}}$  itself as pointed out by Sorkine and Botsch in [SB09]. In this case, the potential enforces only smoothness of the displacements field  $\mathbf{d}$  instead of the surface itself. It is easy to see, that this potential is not invariant with respect rigid transformations of the surface. More precisely, a rotation  $x_i = R_{ij}\tilde{x}_j$  of the surface results in non-vanishing displacement derivatives and thus in a positive potential energy. This leads to contra-intuitive deformation behaviour especially for rotations.

The reason for this lies in our conception of 3D shape as a property, that does not change with the orientation or position of objects. If a 3D object undergoes a purely rigid transformation, we would still recognize its shape as unchanged. In differential geometry, this understanding of shape has led to a rotation and translation invariant shape description by first and second fundamental forms. For shape editing, deformation potentials are used to quantize the difference in shape between original and deformed object. It is therefore reasonable to require that deformation potentials should also not change under rigid transformations. The elastic potential (and its simplified version as used by Terzopoulos) defined in terms of the fundamental forms clearly show this invariance, whereas the Dirichlet energy of the displacements is not rotation invariant.

### 10.2.1 Differential Representations

Recently, a new class of shape editing approaches has become popular whose common feature is a differential representation of the surface to which some authors also refer as “coordinates”. For editing, the original coordinates  $\bar{\mathbf{x}}$  of the surface are first transformed into this differential representation or coordinates by applying a generalized coordinate transformation:

$$\bar{\mathbf{c}} = T(\bar{\mathbf{x}})$$

The user can then interactively move and constrain positions of selected vertices and the mesh is reconstructed from its differential representation subject to these constraints. Mathematically, this reconstruction is an over-constrained least squares problem and thus corresponds to minimizing a potential energy, whose density is given as the squared residual:

$$e = (\bar{\mathbf{c}} - T(\mathbf{x}))^2 \tag{10.1}$$

For efficiency reasons, a linear coordinate transformation  $T$  in the coordinates  $\mathbf{x}$  is very desirable as this leads to a quadratic potential energy. The minimization of such a potential corresponds to solving a linear system.

Differential coordinates or differential representation for interactive mesh editing have been introduced by the work of Alexa [Ale03], Lipman et al. [LSCO<sup>+</sup>04] and Yu et al. [YZX<sup>+</sup>04]. Alexa and Lipman et al. encode local shape as surface



Laplacians, i.e. they use the coordinate transformation  $T(\mathbf{x}) = \Delta'_{\bar{S}}\mathbf{x}$  where  $\Delta'_{\bar{S}}$  is some discrete approximation of the Laplace-Beltrami operator on the surface  $\bar{S}$ . The resulting coordinates are therefore also called *Laplacian coordinates*. They can be given an interpretation as mean curvature normals.<sup>1</sup> Yu et al. use the gradient of the coordinate function as differential representation, i.e.  $T(\mathbf{x}) = \nabla_S\mathbf{x}$  where  $\nabla_S$  is the surface gradient (in this case, we understand  $\mathbf{x}$  as the coordinate function defined on  $S$  itself). Minimizing the corresponding potential leads to a Poisson equation [SB09] so that Yu et al. call their method *Poisson mesh editing*.

Unfortunately, minimizing the potentials corresponding to Laplacian or gradient-based coordinates directly yields not the desired results, especially in the presence of handle rotations. The common problem is, again, related to rigid transformations: In order to arrive at a potential that is invariant with respect to rigid motions, the coordinate transformation  $T$  must already have this property. This, however, is not possible with linear transformation. In particular, both Laplacian or gradient based coordinates are not invariant with respect to rotations.

To achieve rotation-invariance, both types of coordinates are therefore expressed with respect to local frames defined by the surfaces normals and tangents. The coordinate transformation  $T$  is thus replaced with  $T_{local}(\mathbf{x}) = \bar{R}_{\bar{\mathbf{x}}}^t T(\mathbf{x})$  where  $\bar{R}_{\bar{\mathbf{x}}}$  denotes the orthogonal matrix whose columns are the surface normal and two orthogonal tangents of the undeformed surface  $\bar{S}$  at  $\bar{\mathbf{x}}$ . For reconstruction, the differential coordinates  $\bar{\mathbf{c}}$  are transformed back to world space coordinates, i.e. the potential's density is given by

$$e = (R_{\mathbf{x}}\bar{\mathbf{c}} - T(\mathbf{x}))^2$$

where in this case  $R_{\mathbf{x}}$  is defined by normal and tangents of the deformed surface  $S$ .

### Linear Approaches

Reconstruction using the above rotation invariant potential, unfortunately, leads to a chicken and egg problem as local frames of the deformed surface have to be known to reconstruct it from its differential coordinates. To circumvent this problem, several methods have been proposed:

In [YZX<sup>+</sup>04] Yu et al. estimate local frames  $R_{\mathbf{x}}$  of the deformed surface by interpolating the handle rotations over the region of interest. Zayer et al. [ZRKS05] achieve more intuitive deformations by using harmonic functions for interpolation. Their approach was further generalized by Popa et al. [PJS06] to non-homogeneous materials. All three methods share a common problem called

<sup>1</sup>It is well known in differential geometry, that in the continuous case  $\Delta_S\mathbf{x} = -h\mathbf{n}$  where  $h$  denotes the surface's mean curvature and  $\mathbf{n}$  its normal at point  $\mathbf{x}$  (see [dC92]).

*translation-insensitivity*: As local frames are estimated from handle rotations only, a translation of the handle does not affect the frames. In contrast, handle translations clearly lead to a deformation of the surface, so that frames and surface geometry become inconsistent. This inconsistency results in contra-intuitive deformations as shown in Figure 10.1.

In the original work of Lipman et al. [LSCO<sup>+</sup>04] frames are estimated from an underlying smooth base surface. This works well as long as the surface is relatively smooth but produces undesired deformations if it cannot be described as a height field over its base surface. Similarly, Botsch et al. extracted in [BSPG06] a base surface that is deformed using a variational minimization approach and estimated local frames from it. Their method can also handle cases where the original surface is no longer a height field over the base surface but still yields contra-intuitive results for large deformations.

Another approach is to solve for both local frames  $R_x$  and deformed surface coordinates  $\mathbf{x}$  simultaneously as Sorkine et al. proposed in [SLCO<sup>+</sup>04]. In this case, however, the above given potential is no longer quadratic in the unknowns  $(R_x, \mathbf{x})$  and its minimization becomes a non-linear problem. For an efficient reconstruction attempts have been made to avoid the complexity of non-linear optimization by approximating the problem linearly. Sorkine et al. proposed in [SLCO<sup>+</sup>04] to linearize rotations which causes artifacts for larger rotations as shown in [BS08]. Following the same basic idea Fu et al. [FAT07] also solve for local transformations but without any rigidity constraint. In a second step, they factor out rotations from these transformations to estimate local frames. While their method better copes with rotations their results seems to be not as good as those obtained by transformation interpolation [ZRKS05]. Furthermore, vertices in flattish areas need special treatment which complicates their algorithm.

In [LSLCO05] Lipman et al. proposed another approach to estimate local frames: Borrowing from differential geometry they encode local frame changes by discretizing first and second fundamental forms. This leads to a representation invariant to rigid transformations that they term rotation-invariant coordinates. Reconstruction is done in two stages: First local frames are reconstructed using the discretized forms and then geometry is obtained by solving the Poisson equation. Their method yields good results for large rotations. However, it also shows the problem of translation-insensitivity described above. Moreover, in their linear framework local frames  $R_x$  cannot be constrained to be orthogonal as this implies a non-linear constraint. So frames deform in a non-rigid way or even degenerate which causes further artifacts. Their approach has been recently improved in [LCOGL07] to handle rotations larger than  $2\pi$  and to improve volume preservation.

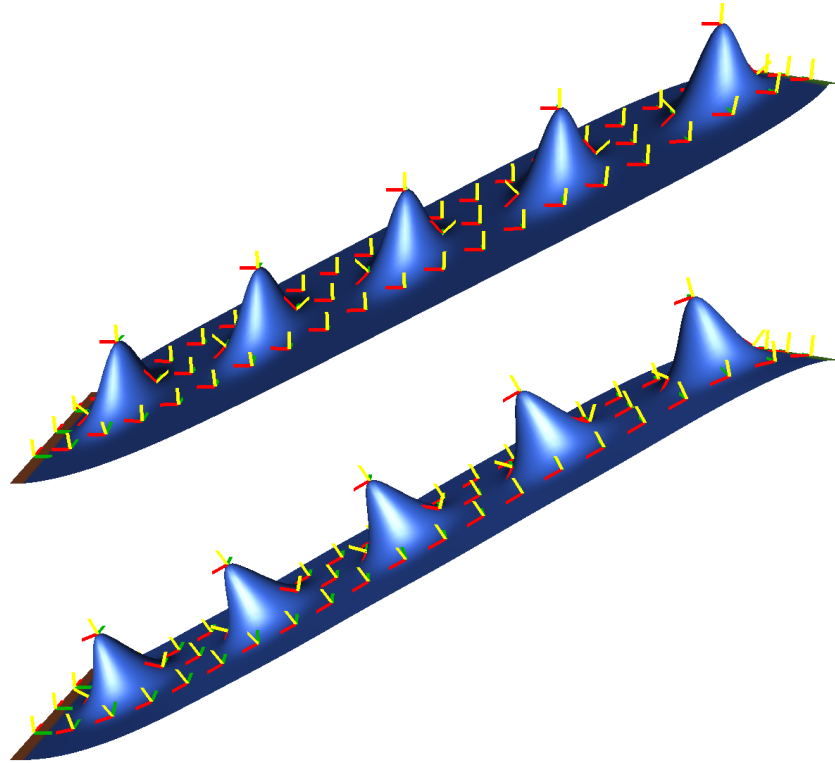


Figure 10.1: A pure translation is applied to the handle. Top: If local frames are not consistent with the actual geometry, features on the surface are reconstructed in a contra-intuitive and undesired way. Bottom: Enforcing consistent frames leads to a correct reconstruction (The shown result was obtained with our method).

### 10.2.2 Non-Linear Approaches

While linear methods are undoubtedly fast, a survey by Botsch and Sorkine [BS08] reveals that all methods result in contra-intuitive deformations in particular under large deformations and pure handle translations. The reasons that were already briefly sketched in the last section can be summarized as follows:

- In the linear framework, local frames cannot be constrained to be orthogonal (this is a non-linear constraint). So they can deform in a non-rigid way or even degenerate which causes contra-intuitive surface deformation.
- Local frames cannot be constrained to be consistent with the geometry, i.e. the frames normal vector can deviate from the actual normal of the deformed surface. Inconsistent frames also result in contra-intuitive deformation (Figure 10.1).

As a consequence, attempts have been made to solve the inherent non-linear reconstruction problem directly by applying non-linear optimizations: Pyramid coordinates [SK04] encode local one ring structures by a set of angles and edge lengths at each vertex. These quantities are invariant under rotations and so is the resulting coordinate transformation. Consequently, these coordinates do not have to be transformed to local coordinate frames and the simpler potential in Equation 10.1 can be minimized for reconstruction. The corresponding minimization problem is, however, non-linear and solved using an iterative optimization. At the price of a rather slow reconstruction the method is robust even for large deformations and translation-sensitive.

The approach of Botsch et al. [BPGK06b] aims at physical plausible deformations albeit it does not derive from the elastic shell model. The idea is to extrude surface triangles to rigid prisms, that are connected by elastic joints. The prisms itself are not allowed to deform. In this way Botsch et al. circumvent problems with degenerate elements as typically encountered in finite element discretizations of thin structures. Although their potential joint-based energy is not of the form in Equation 10.1, its minimization preserves angles between adjacent triangle which corresponds to an implicit preservation of mean curvature as shown in [GHDS03]. In this regard it is related to the approaches here although it does not explicitly extract a differential representation. For mesh editing at interactive frame rates a high dimensional non-linear optimization problem is solved using a sophisticated hierarchical solver. The method yields physically plausible deformations for all kinds of handle transformations. The authors also extended their method to space deformations [BPWG07].

Huang et al. [HSL<sup>+</sup>06] use the fact, that the cotangent discretization of the surface Laplacian [DMSB99] can be characterized as linear combination of adjacent triangle normals to obtain a rotation-invariant non-linear transformation  $T$ . For reconstruction, the resulting non-linear optimization problem is solved by Gauss-Newton iterations on a subspace domain. The deformation of the subspace is extrapolated to the full domain using mean value coordinates. Similar to the method of Botsch et al. [BPGK06b] their optimization procedure is rather complex and involves construction of a suitable subspace domain.

In [ATLF06] Au et al. suggest a coordinate transformation  $T$  that extracts the norm of the surface Laplacian as well as its cotangent weights. They also propose a simple and fast iterative reconstruction. However, their method cannot handle surfaces with free boundaries and does not allow for handle scaling.

---

## A NON-LINEAR POTENTIAL FOR INTERACTIVE EDITING

---

A lesson learned from the vast previous work on interactive deformation potentials is that quadratic potentials, while efficient to minimize, result in deformation artifacts for all but a small subclass of handle movements. In contrast, existing non-linear potentials yield very plausible and intuitive deformations even for large handle transformations and pure translations. However, this comes at the price of comparatively slow reconstruction. Moreover, the specialized optimization algorithms are highly sophisticated and their implementation burdensome.

In this chapter we propose an interactive mesh deformation model that addresses these issues. We propose a novel differential representation that can be regarded as a discretization of metric tensor and second fundamental form. Our coordinate transformation is invariant with respect to rigid transformations and implicitly enforces orthogonal frames. At the same time, the coordinate transformation is carefully chosen to allow an efficient optimization of the resulting potential.

Apt to this representation we formulate a non-linear reconstruction problem that includes a frame consistency constraint and thereby avoids a common shortcoming of linear approaches. We will argue that this formulation enforces conformal deformations that preserve the shape of texture features locally.

We propose a fast iterative optimization tailored to our formulation of the reconstruction problem. The key to this is to avoid iterated matrix factorizations as required by standard non-linear solvers. In addition, it does not require sophisticated hierarchical optimization and naturally exhibits frame-to-frame coherence.

Finally, we describe a simple and efficient implementation of our algorithm, that allows large parts of the computation to run parallel on graphics hardware. This allows editing of large models at interactive frame rates.

## Setup and Notation

In this chapter, we assume the discrete setup described in Section 2.2, i.e. that both undeformed surface  $\bar{S}$  and deformed surface  $S$  are given as discrete triangle meshes with the same connectivity  $\mathcal{M}$ . In particular, the corresponding parameterizations  $\bar{\mathbf{x}}$  and  $\mathbf{x}$  are piecewise affine mappings.

Our differential coordinates are based on quaternions either associated with vertices or directed edges. These quaternions will always be denoted by  $\mathbf{q}_v$  with some vertex index  $u, v, w$  as subscript or by  $\mathbf{q}_v^w$  for a directed edge  $(v, w)$ . If not otherwise stated, a product of the form  $\mathbf{q}_v \mathbf{q}_w$  is understood as a multiplication in the quaternion ring  $\mathbb{H}$ . The conjugated quaternion is denoted by  $\mathbf{q}^*$ . There is a canonical embedding of quaternions into  $\mathbb{R}^4$  and we denote the components of quaternions  $\mathbf{q}_v, \mathbf{q}_v^w$  embedded into this vector space by  $q_{vi}$  and  $q_{vi}^w$  respectively for  $i = 1 \dots 4$ . With these components, the quaternion product  $\mathbf{q}_u = \mathbf{q}_v \mathbf{q}_w$  can be written as:

$$\begin{aligned} q_{u1} &= q_{v1}q_{w1} - q_{v1}q_{w2} - q_{v1}q_{w3} - q_{v1}q_{w4} \\ q_{u2} &= q_{v2}q_{w2} + q_{v2}q_{w1} + q_{v2}q_{w4} - q_{v2}q_{w3} \\ q_{u3} &= q_{v3}q_{w3} - q_{v3}q_{w4} + q_{v3}q_{w1} + q_{v3}q_{w2} \\ q_{u4} &= q_{v4}q_{w4} + q_{v4}q_{w3} - q_{v4}q_{w2} + q_{v4}q_{w1} \end{aligned}$$

To be able to compactly write quaternion products in index notation, we define a symbol  $\rho_{ijk}$  from the signs of the terms in the above equations. Using this operator, the same product can be compactly written using Einstein sum conventions as

$$q_{ui} = \rho_{ijk} q_{vj} q_{wk}$$

In accordance with the popular handle/region-of-interest mesh editing metaphor described earlier, we expect that a subset  $R \subset V$  corresponding to the region-of-influence (ROI) is given. Moreover, a further subset  $H \subset V$  contains vertices for which either a specific rigid transformation has been specified by the user or that have been constrained to stay fixed. The latter can be regarded as a special case of the former so that we can keep both kinds in a single set. For mesh editing, the problem consists then in finding a deformed mesh whose handle regions are transformed as specified while the shape of the unconstrained mesh area is preserved as much as possible.

## 11.1 Quaternion-based Coordinates

### 11.1.1 From Fundamental Forms to Local Frames

The characterization of local shape and its preservation is at the heart of every differential representation. The basic question that has to be answered is the following: What comprises the shape of a surface? While many different definitions of shape have been proposed for triangle meshes over the last years, differential geometry provides us with a simple answer to the question in case of smooth surfaces. In this case, the shape of a surface is locally described by the first and second fundamental form as introduced in Chapter 2. But given these entities over the hole surface, its global shape is also uniquely determined up to rigid transformation. This is asserted by the fundamental theorem on surface theory (see e.g. [Cia05], chapter 2).

Let us consider for a moment the continuous case. We recall that for each point on the surface  $\bar{S}$  a local frame  $(\bar{\mathbf{a}}_1, \bar{\mathbf{a}}_2, \bar{\mathbf{n}})$  is given via  $\bar{\mathbf{a}}_\alpha = \bar{\mathbf{x}}_{,\alpha}$  and the surface normal  $\bar{\mathbf{n}} = \bar{\mathbf{a}}_3 = (\bar{\mathbf{a}}_1 \times \bar{\mathbf{a}}_2) / \|\bar{\mathbf{a}}_1 \times \bar{\mathbf{a}}_2\|$ . In particular, both fundamental forms can be expressed in terms of these frames as

$$\bar{g}_{\alpha\beta} = \bar{\mathbf{a}}_\alpha \cdot \bar{\mathbf{a}}_\beta \quad \bar{h}_{\alpha\beta} = \bar{\mathbf{a}}_{\alpha,\beta} \cdot \bar{\mathbf{n}} \quad (11.1)$$

If, for a rotation  $R$  and an arbitrary  $\mathbf{t} \in \mathbb{R}^3$ , the rigid transformation  $R\mathbf{p} + \mathbf{t}$  is applied to  $\bar{S}$ , local frames transform to  $(R\bar{\mathbf{a}}_1, R\bar{\mathbf{a}}_2, R\bar{\mathbf{n}})$  and we have

$$R\bar{\mathbf{a}}_\alpha \cdot R\bar{\mathbf{a}}_\beta = \bar{\mathbf{a}}_\alpha \cdot \bar{\mathbf{a}}_\beta \quad R\bar{\mathbf{a}}_{\alpha,\beta} \cdot R\bar{\mathbf{n}} = \bar{\mathbf{a}}_{\alpha,\beta} \cdot \bar{\mathbf{n}} \quad (11.2)$$

and thus both forms are invariant with respect to rigid transformations. As already mentioned earlier, this corresponds to our conception of shape as a position and orientation independent property. Because of these properties, first and second fundamental forms are ideal differential coordinates.

To transfer the concept of fundamental forms to the discrete case, we make the following observation, that links local frames and fundamental forms: The first fundamental form  $g_{\alpha\beta}$  encodes the length and angle of the frame's tangential components while the second fundamental form encodes frame differences between neighboring frames. Based on this observation is the differential representation of Lipman et al. [LSLCO05] who present a discretization of the second fundamental form that encodes differences of adjacent local frames. Inspired by their approach we propose here a novel discrete differential mesh representation that avoids the above mentioned typical problems of linear approaches. It implicitly preserves the first fundamental form by enforcing orthogonal local frames. Similar to the second fundamental form this representation encodes differences of neighboring frames in a way that is invariant to rigid transformations.

### 11.1.2 Encoding Local Frames

To derive our mesh-based differential representation, we begin by fixing for all vertices in the region of interest  $v \in R$  an arbitrary orthonormal local frame  $\bar{F}_v = (\bar{\mathbf{a}}_1^v, \bar{\mathbf{a}}_2^v, \bar{\mathbf{n}}^v)$  with right hand orientation where  $\bar{\mathbf{n}}^v$  denotes the normal at vertex  $v$  e.g. obtained by averaging adjacent triangle normals. In the following we understand  $\bar{F}_v$  as a matrix whose columns equal tangent vectors and normal. To find a discrete analog for the second fundamental form, we strive to capture differences of adjacent frames. We achieve this by expressing neighboring frames in coordinates of the frame  $\bar{F}_v$ , i.e. for a pair of vertices  $v, w$  adjacent in  $\mathcal{M}$  we compute the matrix

$$\bar{F}_v^w := \bar{F}_v^{-1} \cdot \bar{F}_w. \quad (11.3)$$

Now, if a rigid transformation  $R\mathbf{x} + \mathbf{t}$  with a rotation matrix  $R$  and  $\mathbf{t} \in \mathbb{R}^3$  is applied to the mesh local frames  $\bar{F}_v$  will rotate to  $R \cdot \bar{F}_v$  and we have

$$(R \cdot \bar{F}_v)^{-1} \cdot R \cdot \bar{F}_w = \bar{F}_v^{-1} \cdot \bar{F}_w = \bar{F}_v^w.$$

Therefore, the definition of  $\bar{F}_v^w$  is invariant with respect to rigid transformations as required.

Since all frames  $\bar{F}_v$  are orthonormal with right handed orientation so are the differences  $\bar{F}_v^w$  and both kind of matrices correspond to rotations. We can therefore rewrite Equation (11.3) using unit quaternions and the quaternion multiplication as

$$\bar{\mathbf{q}}_v^w = \bar{\mathbf{q}}_v^{-1} \cdot \bar{\mathbf{q}}_w \quad (11.4)$$

where  $\bar{\mathbf{q}}_v$ ,  $\bar{\mathbf{q}}_w$  and  $\bar{\mathbf{q}}_v^w$  correspond to the rotations  $\bar{F}_v$ ,  $\bar{F}_w$  and  $\bar{F}_v^w$  respectively.

The chosen representation of frames as unit quaternions implicitly enforces orthonormal frames. This property ensures that frames do not degenerate even if we add handle constraints to incorporate the user specified transformations in the reconstruction. By keeping all frames orthonormal mesh editing is essentially restricted to isometric deformations, i.e. deformations that preserve the metric tensor and thus angle, length and area of the parameterization. Even if in our least squares reconstruction (that we will discuss in a minute) deformations might violate this constrained to some extent, enforcing orthonormal frames is usually too restrictive for mesh editing applications. For example the user might want to scale certain parts of a surface which certainly scales length and area of the parameterization. To enable such deformations it is common practice to allow a local isotropic scaling [SLCO<sup>+</sup>04, ZRKS05].

In our case, this means that we only restrict frames to be orthogonal not orthonormal. In terms of the parameterization, this allows for deformations that are no longer isometric but conformal. If the mesh is textured, local features in



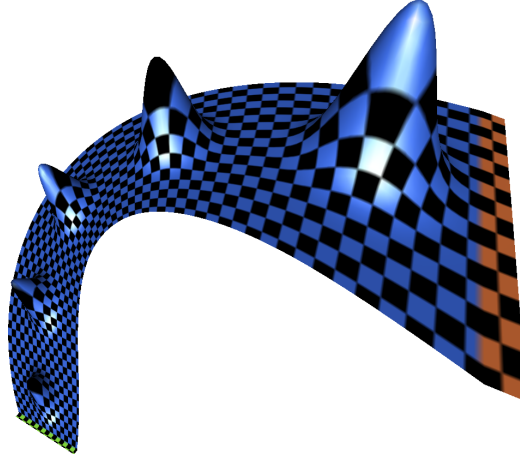


Figure 11.1: By enforcing conformal deformations local shapes in the texture are preserved even under extreme deformations.

the texture will therefore also keep their shape during editing as demonstrated in Figure 11.1.

Fortunately, rotations with isotropic scales can be represented by non-unit quaternions. More precisely, if the rotation  $R$  is represented by the unit-quaternion  $\mathbf{q}$ , the combined isotropic scale and rotation  $sR$  is given by the quaternion  $\sqrt{s}\mathbf{q}$ . This follows immediately from the quaternion-based representation of rotations. To incorporate isotropic scales in the above described frame reconstruction it is therefore sufficient, to drop the unit length requirement. This is advantageous since constraining local frames to unit-length quaternions would result in a non-linear optimization constraint.

### 11.1.3 Encoding Local Geometry

By now our representation is only based on local frames and does not include the actual geometry. However, as mentioned above frames should be consistent to the geometry to avoid the problem shown in Figure 10.1, e.g. the normal vector  $\mathbf{n}$  of each frame should be perpendicular to the reconstructed surface, and the relative position of frame vectors with respect to the 1-ring edges should not change.

To formalize this constraint we require that the coordinates of 1-ring edges in the local frames  $\bar{F}_v$  should not change in the reconstruction. For each directed edge  $(v, w)$  between vertices in  $\mathcal{M}$  these local coordinates are given by

$$\bar{\mathbf{c}}_v^w := \bar{F}_v^{-1}(\bar{\mathbf{x}}_w - \bar{\mathbf{x}}_v),$$

which we compute from the original undeformed mesh. Using quaternions, this

relation between edge vectors  $(\bar{\mathbf{x}}_w - \bar{\mathbf{x}}_v)$  and local frames  $\bar{F}_v$  can be written as

$$(1, \bar{\mathbf{c}}_v^w)^t = \bar{\mathbf{q}}_v^*(1, \bar{\mathbf{x}}_w - \bar{\mathbf{x}}_v)^t \bar{\mathbf{q}}_v \quad (11.5)$$

where  $(1, \mathbf{c})^t$  denotes the quaternion  $\mathbf{q}$  with

$$q_1 = 1 \quad \text{and} \quad q_i = c_{i-1} \text{ for } i = 2 \dots 4.$$

We are now able to define our quaternion-based differential coordinates: For an undeformed surface  $\bar{S}$  given as triangle mesh  $\mathcal{M}$ , we first fix orthogonal frames  $\bar{\mathbf{q}}_v$  at each vertex. Then we compute at each directed edge between vertices  $(v, w)$  adjacent in  $\mathcal{M}$  the quaternion  $\bar{\mathbf{q}}_v^w$  (as detailed in the last section) and the directed edge vector in local frame coordinates  $\bar{\mathbf{c}}_v^w$ . These entities comprise our differential representation. Please note, that this representation is specifically derived for triangle meshes. The corresponding coordinate transformation therefore takes the mesh connectivity  $\mathcal{M}$ , the vertex positions  $(\bar{\mathbf{x}}_v)_{v \in R}$ , and a set of frames  $(\bar{\mathbf{q}}_v)_{v \in R}$  as arguments instead of a continuous parameterization  $\bar{\mathbf{x}}$ . For the just defined coordinates it has the following form:

$$T_{\mathcal{M}}((\bar{\mathbf{x}}_v)_{v \in R}, (\bar{\mathbf{q}}_v)_{v \in R}) := (\bar{\mathbf{q}}_v^w, \bar{\mathbf{c}}_v^w)_{(v,w) \in \mathcal{M}}$$

The discrete coordinate transformation thus takes a mesh as input and returns a tuple  $(\bar{\mathbf{q}}_v^w, \bar{\mathbf{c}}_v^w)$  for each directed edge in  $\mathcal{M}$ .

## 11.2 Reconstruction Potentials

### 11.2.1 Reconstructing from Quaternion-based Coordinates

Our differential representation should contain enough information about the surface to reconstruct its shape. Let us assume that a coordinate representation  $\bar{\mathbf{c}} := (\bar{\mathbf{q}}_v^w, \bar{\mathbf{c}}_v^w)_{(v,w) \in \mathcal{M}} = T_{\mathcal{M}}((\bar{\mathbf{x}}_v)_{v \in R}, (\bar{\mathbf{q}}_v)_{v \in R})$  is given. The reconstruction problem can then be formulated as follows: Find a set of vertex positions  $(\mathbf{x}_v)_{v \in R}$  and frames  $(\mathbf{q}_v)_{v \in R}$  that satisfy

$$\bar{\mathbf{c}} = T_{\mathcal{M}}((\mathbf{x}_v)_{v \in R}, (\mathbf{q}_v)_{v \in R}) \quad (11.6)$$

With  $(\mathbf{q}_v^w, \mathbf{c}_v^w)_{(v,w) \in \mathcal{M}} := T((\mathbf{x}_v)_{v \in R}, \mathcal{M})$  and using the defining Equations 11.5 and 11.4 of the coordinate transformation  $T$  we can equivalently rewrite this as:

$$(\bar{\mathbf{q}}_v^w, \bar{\mathbf{c}}_v^w)_{(v,w) \in \mathcal{M}} = (\mathbf{q}_v^w, \mathbf{c}_v^w)_{(v,w) \in \mathcal{M}} \Leftrightarrow \quad (11.7)$$

$$\bar{\mathbf{q}}_v^w = \mathbf{q}_v^w \quad \wedge \quad \bar{\mathbf{c}}_v^w = \mathbf{c}_v^w \quad \forall (v, w) \in \mathcal{M} \Leftrightarrow \quad (11.8)$$

$$\bar{\mathbf{q}}_v^w = \mathbf{q}_v^{-1} \mathbf{q}_w \quad \wedge \quad (1, \bar{\mathbf{c}}_v^w)^t = \mathbf{q}_v^*(1, \mathbf{x}_w - \mathbf{x}_v)^t \mathbf{q}_v \quad \forall (v, w) \in \mathcal{M} \quad (11.9)$$

Reconstruction from coordinates  $\bar{\mathbf{c}}$  thus amounts to solving the equation system [11.9](#) for both vertex positions  $\mathbf{x}_v$  and frames  $\mathbf{q}_v$ . The question, whether this equation system possesses a solution is answered trivially, by choosing  $\mathbf{x}_v = \bar{\mathbf{x}}_v$  and  $\mathbf{q}_v = \bar{\mathbf{q}}_v$  for all vertices  $v \in R$ . Equations [11.10](#) and [11.11](#) are then satisfied by definition. (We defer a discussion on the uniqueness of this solution to the next section.) However, for interactive mesh editing we are not interested in a perfect reconstruction. We rather seek for a surface, that is locally similar in shape to the original surface, but that also satisfies user specified transformations at handle vertices. In our framework, we allow for two kinds of constraints: Positional constraints specify a desired position  $\mathbf{x}_v^{const}$  for a handle vertex  $v \in H$ . Orientation constraints specify a local coordinate system  $\mathbf{q}_v^{const}$  at a handle vertex. They can be used e.g. to fix surface normal and directional derivatives at a vertex while allowing it to translate freely. Instead of solving Equation [11.9](#) we therefore seek a solution to the following system:

$$\bar{\mathbf{q}}_v^w = \mathbf{q}_v^{-1} \mathbf{q}_w \quad \forall (v, w) \in \mathcal{M} \quad (11.10)$$

$$(1, \bar{\mathbf{c}}_v^w)^t = \mathbf{q}_v^*(1, \mathbf{x}_w - \mathbf{x}_v)^t \mathbf{q}_v \quad \forall (v, w) \in \mathcal{M} \quad (11.11)$$

$$\mathbf{q}_v = \mathbf{q}_v^{const} \quad \forall v \in H \quad (11.12)$$

$$\mathbf{x}_v = \mathbf{x}_v^{const} \quad \forall v \in H \quad (11.13)$$

In the presence of more than one position and orientation constraint, the system is overconstraint and no perfect solution exists. We therefore loosen requirements by resorting to solution that minimizes least squares of residuals. While a least squares solution is no longer a perfect reconstruction of the original surface, it keeps deviations in quaternion-based coordinates small at each vertex and thus results in preservation of local shape. In the next section, we derive a least-squares potential for Equations [11.10-11.13](#).

### 11.2.2 Deformation Potentials for Quaternion-based Coordinates

In analogy to Equation [10.1](#), a potential based on least-squares residuals for equations [11.10-11.13](#) is given by

$$\begin{aligned} E' &= \alpha_q \sum_{(v,w) \in \mathcal{M}} \|\bar{\mathbf{q}}_v^w - \mathbf{q}_v^{-1} \cdot \mathbf{q}_w\|^2 \\ &+ \alpha_x \sum_{(v,w) \in \mathcal{M}} \|(1, \bar{\mathbf{c}}_v^w)^t - \mathbf{q}_v^*(1, \mathbf{x}_w - \mathbf{x}_v)^t \mathbf{q}_v\|^2 \\ &+ \alpha_c E_{const} \end{aligned} \quad (11.14)$$

where we introduced scalar parameters  $\alpha_q$ ,  $\alpha_x$ ,  $\alpha_c$  that control the relative importance of the residual errors. Their choice will be discussed in Section 11.3.5. Please note, that in the first term a least squares norm on the difference of quaternions is used to measure the distance between quaternions. Although this choice arises naturally in this context, there are some important implications for the reconstruction algorithm. We defer a discussion of this choice to Section 11.3.4.

$E_{const}$  ensures that user specified handle transformations are respected. This can be done by the method of Lagrange multipliers but requires additional variables that increases dimensionality. Alternatively, handle transformation can be implemented as soft constraints by choosing:

$$E_{const} = \sum_{v \in H} \|\mathbf{q}_v - \mathbf{q}_v^{const}\|^2 + \sum_{v \in H} \|\mathbf{x}_v - \mathbf{x}_v^{const}\|^2 \quad (11.15)$$

Please note, that is easily possible to constraint position and local frames of a vertex  $v$  independently. We refer to vertices with constrained position as *position constraints* and to those with constrained local frame to *orientation constraints*. For implementation, it is sufficient to replace the set  $H$  by two sets  $H_x$  and  $H_q$  of position-constraint and orientation-constraint vertices in the above equation, i.e.

$$E_{const} = \sum_{v \in H_q} \|\mathbf{q}_v - \mathbf{q}_v^{const}\|^2 + \sum_{v \in H_x} \|\mathbf{x}_v - \mathbf{x}_v^{const}\|^2 \quad (11.16)$$

While the potential  $E'$  is a straightforward choice, it poses some difficulties in the optimization. In the next section we will describe an interactive optimization algorithm based on *alternating optimization* (AO) [Zan69]: In this approach the set of all unknowns is divided into multiple subsets, that correspond to local frames  $\mathbf{q}_v$  and vertex positions  $\mathbf{x}_v$ . In each step of the algorithm, the potential is minimized only with respect to one subset of all unknowns while keeping all other variables fixed. The set of active variables is alternated in each iteration.

Our basic idea for an efficient optimization is to keep each of these optimization steps as simple as possible. Preferably, optimizing the potential with respect to each of the variable subsets should either amount to solving a linear system or to solving a low dimensional non-linear problem. For the linear system, we also aim at a system matrix, that does not change between iterations. In this way, it is possible to precompute a matrix factorization. The actual solution of the linear system then becomes highly efficient.

In view of this optimization strategy, the above potential has two problems: First, it is not a quadratic function in the local frames  $\mathbf{q}_v$  (in particular, it is expressed in terms of the inverse  $\mathbf{q}_v^{-1}$ ). Thus, minimization with respect to this variable subset results into a non-linear system. Second, while the potential is quadratic in the vertex positions  $\mathbf{x}_v$ , the product of frames and positions in the

second term leads to a system matrix, that depends on the actual values of  $\mathbf{q}_v$  and is thus not constant.

To address these problems, we turn back to the equation system [11.10-11.13](#). We equivalently rewrite the system by multiplying  $\mathbf{q}_v$  from the left of the first equations. For the second we multiply by  $(\mathbf{q}_v^*)^{-1}$  and  $\mathbf{q}_v^{-1}$  from left and right respectively which results in:

$$\mathbf{q}_v \bar{\mathbf{q}}_v^w = \mathbf{q}_w \quad \forall (v, w) \in \mathcal{M} \quad (11.17)$$

$$\frac{1}{\|\mathbf{q}_v\|^2} \mathbf{q}_v (1, \bar{\mathbf{c}}_v^w)^t \mathbf{q}_v^{-1} = (1, \mathbf{x}_w - \mathbf{x}_v)^t \quad \forall (v, w) \in \mathcal{M} \quad (11.18)$$

$$\mathbf{q}_v = \mathbf{q}_v^{const} \quad \forall v \in H \quad (11.19)$$

$$\mathbf{x}_v = \mathbf{x}_v^{const} \quad \forall v \in H \quad (11.20)$$

Based on least-squares residuals, we again derive a potential  $E''$  from these equations:

$$\begin{aligned} E'' &= \alpha_q \sum_{(v,w) \in \mathcal{M}} \|\mathbf{q}_v \bar{\mathbf{q}}_v^w - \mathbf{q}_w\|^2 + \\ &+ \alpha_x \sum_{(v,w) \in \mathcal{M}} \left\| \frac{\mathbf{q}_v (1, \bar{\mathbf{c}}_v^w)^t \mathbf{q}_v^{-1}}{\|\mathbf{q}_v\|^2} - (1, \mathbf{x}_w - \mathbf{x}_v)^t \right\|^2 \\ &+ \alpha_c E_{const} \end{aligned} \quad (11.21)$$

As the equation systems [11.10-11.13](#) and [11.17-11.20](#) are equivalent, a perfect reconstruction is a solution to both systems and thus a minimizer for both potentials. The modified potential  $E''$  is also quadratic in the vertex positions  $\mathbf{x}_v$  but in contrast to  $E'$ , the product of frames and positions in the second term is avoided. As described in the next section, minimizing  $E''$  with respect to positions  $\mathbf{x}_v$  thus amounts to solving a linear system with constant system matrix.

More precisely, the system matrix is a uniformly weighted Laplacian matrix. Assuming local frames are known, solving for the coordinates  $\mathbf{x}_v$  thus comes down to reconstruction from Laplacian coordinates. In this respect there is a connection of our approach to linear methods like [[LSCO+04](#), [YZX+04](#), [ZRKS05](#), [LSLCO05](#)]. In contrast to those methods that estimate local frames independent of the geometry, our approach solves for frames and geometry simultaneously. Therefore, the frames  $\mathbf{q}_v$  are forced to be consistent with the actual geometry  $\mathbf{x}_v$ . A translation of handle vertices specified as constraint to positions will indirectly influence the frame field. Therefore, our approach is sensitive to translations as shown in [Figure 10.1](#).

Concerning optimization with respect to local frames, the first term in the modified potential  $E''$  and the constraint energy  $E_{const}$  are both quadratic in the

frames  $\mathbf{q}_v$ . Unfortunately, the second term is not. An optimization of  $E''$  with respect to  $\mathbf{q}_v$  is therefore still a high dimensional non-linear problem. As the second term ensures consistency of frames, it is exactly the coupling of frames and geometry that turns the reconstruction into a non-linear problem. Therefore, we cannot expect a quadratic residual potential.

According to the above described optimization strategy, we would like to separate high-dimensional linear and low-dimensional non-linear problems in the optimization. Our idea is to loosen the above mentioned coupling of frames and geometry during reconstruction allowing them to become temporarily inconsistent. To allow for this deviation in a controlled manner we introduce a set of auxiliary variables: For each vertex in the region of interest we add a quaternion  $\mathbf{q}_v^{consist}$  of which we think as a frame that is consistent with the actual geometry. We then substitute  $\mathbf{q}_v^{consist}$  for  $\mathbf{q}_v$  in the second term. To ensure, that both frame sets do not deviate too much, we add an additional term to the potential. Our final potential then takes the following form:

$$\begin{aligned}
 E = & \alpha_q \sum_{(v,w) \in \mathcal{M}} \|\mathbf{q}_v \bar{\mathbf{q}}_w - \mathbf{q}_w\|^2 + & (11.22) \\
 & + \alpha_x \sum_{(v,w) \in \mathcal{M}} \left\| \frac{\mathbf{q}_v^{consist} (1, \bar{\mathbf{c}}_v^w)^t (\mathbf{q}_v^{consist})^{-1}}{\|\mathbf{q}_v^{consist}\|^2} - (1, \mathbf{x}_w - \mathbf{x}_v)^t \right\|^2 + \\
 & + \alpha_{qc} \sum_{v \in R} \|\mathbf{q}_v - \mathbf{q}_v^{consist}\|^2 \\
 & + \alpha_c E^{const}
 \end{aligned}$$

The above potential is a quadratic function in both vertex positions  $\mathbf{x}_v$  and local frames  $\mathbf{q}_v$ . As will be shown in the next section, optimizing with respect to the geometry consistent frames  $\mathbf{q}_v^{consist}$  amounts to a low-dimensional problem, that can be solved for all vertices in parallel.

### 11.2.3 Derivatives

For the optimization algorithm described in the next section, we need to take derivatives with respect to vertex positions  $\mathbf{x}_v$  and quaternions  $\mathbf{q}_v$ . These are derived in this section. We begin by rewriting the potential to separate the terms that only depend on  $\mathbf{x}_v$ ,  $\mathbf{q}_v$  and  $\mathbf{q}_v^{consist}$ , respectively. To this extent, we first define three symbols that correspond to right hand side multiplication with the quater-

nions  $\bar{\mathbf{q}}_v^w$ ,  $(1, \bar{\mathbf{c}}_v^w)^t$  and  $(1, \mathbf{x}_w - \mathbf{x}_v)^t$  as follows:

$$\begin{aligned}\bar{Q}_{vwij} &:= \mathcal{M}_{vw} \rho_{ijk} \bar{q}_{vk}^w \\ \bar{C}_{vwij} &:= \mathcal{M}_{vw} \left( \rho_{ij1} + \sum_{k=2}^4 \rho_{ijk} \bar{c}_{v(k-1)}^w \right) \\ X_{vwij} &:= \mathcal{M}_{vw} \left( \rho_{ij1} + \sum_{k=2}^4 \rho_{ijk} (x_{w(k-1)} - x_{v(k-1)}) \right)\end{aligned}$$

Assuming that indices  $u, v, w$  take values in  $R$  and extending Einstein sum conventions accordingly, we can then rewrite the terms of the potential:

$$\begin{aligned}\sum_{(v,w) \in \mathcal{M}} \|\mathbf{q}_v \bar{\mathbf{q}}_v^w - \mathbf{q}_w\|^2 &= \sum_{(v,w) \in \mathcal{M}} (\|\mathbf{q}_v \bar{\mathbf{q}}_v^w\|^2 - 2 \langle \mathbf{q}_v \bar{\mathbf{q}}_v^w, \mathbf{q}_w \rangle + \|\mathbf{q}_w\|^2) \\ &= \bar{Q}_{vwki} \bar{Q}_{vwkj} q_{vi} q_{vj} - 2 \bar{Q}_{vwji} q_{vi} q_{wj} + \mathcal{M}_{vw} \mathcal{M}_{vw} q_{wj} q_{wj}\end{aligned}$$

$$\begin{aligned}\sum_{v \in R} \|\mathbf{q}_v - \mathbf{q}_v^{consist}\|^2 &= q_{vi} q_{vi} - 2 q_{vi} q_{vi}^{consist} + q_{vi}^{consist} q_{vi}^{consist} \\ \sum_{v \in H} \|\mathbf{q}_v - \mathbf{q}_v^{const}\|^2 &= H_v q_{vi} q_{vi} - 2 H_v q_{vi} q_{vi}^{const} + H_v q_{vi}^{const} q_{vi}^{const} \\ \sum_{v \in H} \|\mathbf{x}_v - \mathbf{x}_v^{const}\|^2 &= H_v x_{vi} x_{vi} - 2 H_v x_{vi} x_{vi}^{const} + H_v x_{vi}^{const} x_{vi}^{const}\end{aligned}$$

For the remaining consistency term we get

$$\begin{aligned}&\sum_{(v,w) \in \mathcal{M}} \left\| \frac{\mathbf{q}_v^{consist} (1, \bar{\mathbf{c}}_v^w)^t (\mathbf{q}_v^{consist})^{-1}}{\|\mathbf{q}_v^{consist}\|^2} - (1, \mathbf{x}_w - \mathbf{x}_v)^t \right\|^2 \\ &= \sum_{(v,w) \in \mathcal{M}} \frac{\|\mathbf{q}_v^{consist} (1, \bar{\mathbf{c}}_v^w)^t (\mathbf{q}_v^{consist})^{-1}\|^2}{\|\mathbf{q}_v^{consist}\|^4} \\ &\quad - 2 \sum_{(v,w) \in \mathcal{M}} \left\langle (1, \mathbf{x}_w - \mathbf{x}_v)^t, \frac{\mathbf{q}_v^{consist} (1, \bar{\mathbf{c}}_v^w)^t (\mathbf{q}_v^{consist})^{-1}}{\|\mathbf{q}_v^{consist}\|^2} \right\rangle \\ &\quad + \sum_{(v,w) \in \mathcal{M}} \|(1, \mathbf{x}_w - \mathbf{x}_v)^t\|^2\end{aligned}$$

To further simplify these expressions, we remark that for quaternions  $\mathbf{p}, \mathbf{q}$ , and  $\mathbf{r}$  the following basic relations between dot product, vector norm and quaternion

product hold (see e.g. [Hor87]):

$$\|\mathbf{pq}\| = \|\mathbf{p}\|\|\mathbf{q}\| \quad (11.23)$$

$$\langle \mathbf{pq}, \mathbf{r} \rangle = \langle \mathbf{p}, \mathbf{rq}^* \rangle \quad (11.24)$$

Using these relations we find

$$\|\mathbf{q}_v^{consist}(1, \bar{\mathbf{c}}_v^w)^t(\mathbf{q}_v^{consist})^{-1}\|^2 = \|\bar{\mathbf{c}}_v^w\|^2 + 1$$

and

$$\begin{aligned} & \langle (1, \mathbf{x}_w - \mathbf{x}_v)^t, \mathbf{q}_v^{consist}(1, \bar{\mathbf{c}}_v^w)^t(\mathbf{q}_v^{consist})^{-1} \rangle = \\ & = \frac{1}{\|\mathbf{q}_v^{consist}\|^2} \langle \mathbf{q}_v^{consist}(1, \bar{\mathbf{c}}_v^w)^t, (1, \mathbf{x}_w - \mathbf{x}_v)^t \mathbf{q}_v^{consist} \rangle \end{aligned}$$

The consistency term can then be written as

$$\begin{aligned} & \sum_{(v,w) \in \mathcal{M}} \|\mathbf{q}_v^{consist}(1, \bar{\mathbf{c}}_v^w)^t(\mathbf{q}_v^{consist})^{-1} - (1, \mathbf{x}_w - \mathbf{x}_v)^t\|^2 \\ & = \frac{\frac{1}{4}\bar{C}_{vwij}\bar{C}_{vwij} + \mathcal{M}_{vw}\mathcal{M}_{vw} - 2q_{vj}^{consist}X_{vwjk}\bar{C}_{vuki}q_{vi}^{consist}}{(q_{vl}^{consist}q_{vl}^{consist})^2}}{+ 2(\mathcal{M}_{vw}\mathcal{M}_{vw}x_{wj}x_{wj} - \mathcal{M}_{vw}x_{vi}x_{wi}) + \mathcal{M}_{vw}\mathcal{M}_{vw}} \end{aligned}$$

Finally, we define

$$\begin{aligned} A_{vwij}^1 & = \alpha_q (\delta_{vw}\bar{Q}_{vuki}\bar{Q}_{vukj} - 2\bar{Q}_{vwji} + \delta_{vw}\delta_{ij}\mathcal{M}_{uw}\mathcal{M}_{uw}) \\ & \quad + \alpha_{qc}\delta_{vw}\delta_{ij} + \alpha_c H_v \delta_{vw}\delta_{ij} \quad (\mathbf{v}, \mathbf{w}, \mathbf{i}, \mathbf{j} \text{ not summed}) \end{aligned}$$

$$A_{vwij}^2 = \alpha_{qc}\delta_{vw}\delta_{ij} \quad (\mathbf{v}, \mathbf{w}, \mathbf{i}, \mathbf{j} \text{ not summed})$$

$$\begin{aligned} A_{vwij}^3 & = 2\alpha_x (\delta_{vw}\delta_{ij}\mathcal{M}_{uw}\mathcal{M}_{uw} - \delta_{ij}\mathcal{M}_{vw}) \\ & \quad + \alpha_c H_v \delta_{vw}\delta_{ij} \quad (\mathbf{v}, \mathbf{w}, \mathbf{i}, \mathbf{j} \text{ not summed}) \end{aligned}$$

$$A_{vwij}^4 = -2\alpha_{qc}\delta_{vw}\delta_{ij} \quad (\mathbf{v}, \mathbf{i} \text{ not summed})$$

$$b_{vi}^1 = -2H_v \alpha_c q_{vi}^{const} \quad (\mathbf{v}, \mathbf{i} \text{ not summed})$$

$$b_{vi}^2 = -2H_v \alpha_c x_{vi}^{const} \quad (\mathbf{v}, \mathbf{i} \text{ not summed})$$



and obtain the potential  $E$  as

$$\begin{aligned}
 E = & \frac{\frac{1}{4}\bar{C}_{vwij}\bar{C}_{vwij} + \mathcal{M}_{vw}\mathcal{M}_{vw} - 2q_{vj}^{consist}X_{vwjk}\bar{C}_{vuki}q_{vi}^{consist}}{(q_{vl}^{consist}q_{vl}^{consist})^2} \\
 & + A_{vwij}^1q_{vi}q_{wj} + A_{vwij}^2q_{vi}^{consist}q_{wj}^{consist} + A_{vwij}^3x_{vi}x_{wj} + A_{vwij}^4q_{vi}^{consist}q_{wj}^{consist} \\
 & + b_{vi}^1q_{vi} + b_{vi}^2x_{vi} + const
 \end{aligned}$$

where  $A^1$ - $A^4$ ,  $b^1$ ,  $b^2$  and  $const$  are derived from the quaternion based coordinates only and thus constant with respect to the unknowns  $(\mathbf{q}_v, \mathbf{q}_v^{consist}, \mathbf{x}_v)$ . In this form it becomes immediately clear that  $E$  is a quadratic function in the local frames  $\mathbf{q}_v$  and the vertex positions  $\mathbf{x}_v$ .

Derivatives with respect to all unknowns are now easily found as:

$$\begin{aligned}
 \frac{\partial E}{\partial q_{vi}} &= (A_{vwij}^1 + A_{wvji}^1)q_{wj} + A_{vwij}^4q_{wj}^{consist} + b_{vi}^1 \\
 \frac{\partial E}{\partial x_{vi}} &= (A_{vwij}^3 + A_{wvji}^3)x_{wj} - 2\frac{q_{uj}^{consist}(\mathcal{M}_{uw} - \mathcal{M}_{vw})\rho_{jk(i+1)}\bar{C}_{uwkm}q_{um}^{consist}}{(q_{ul}^{consist}q_{ul}^{consist})^2} + b_{vi}^2 \\
 \frac{\partial E}{\partial q_{vi}^{consist}} &= -2\frac{X_{vwik}\bar{C}_{vukj} + X_{vwjk}\bar{C}_{vuki}q_{vj}^{consist}}{(q_{vl}^{consist}q_{vl}^{consist})^2} \\
 & - 4\frac{(\frac{1}{4}\bar{C}_{uwkj}\bar{C}_{uwkj} + \mathcal{M}_{uw}\mathcal{M}_{uw} - 2X_{vwjk}\bar{C}_{vukm}q_{vj}^{consist}q_{vm}^{consist})q_{vi}^{consist}}{(q_{vl}^{consist}q_{vl}^{consist})^3} \\
 & + (A_{vwij}^2 + A_{wvji}^2)q_{wj}^{consist} + A_{wvji}^4q_{wj}
 \end{aligned}$$

#### 11.2.4 Properties

In this section, we show some basic properties of the derived potential  $E$ . As quaternion-based coordinates have been designed to be rotation invariant, the following lemma comes as no surprise:

**Lemma 11.1.** *In the absence of constraints, the potential  $E$  defined in Equation 11.21 is invariant under rigid transformations, i.e. if  $\mathbf{q} \in \mathbb{H}$  denotes an arbitrary rotation represented as unit quaternion and  $t \in \mathbb{R}^3$  a translation, we have*

$$E(\mathbf{q}_v, \mathbf{q}_v^{consist}, \mathbf{x}_v) = E(\mathbf{q}'_v, \mathbf{q}_v^{consist'}, \mathbf{x}'_v)$$

for

$$\begin{aligned}
 \mathbf{q}'_v &:= \mathbf{q}\mathbf{q}_v \\
 \mathbf{q}_v^{consist'} &:= \mathbf{q}\mathbf{q}_v^{consist} \\
 (1, \mathbf{x}'_v)^t &:= \mathbf{q}(1, \mathbf{x}_v + t)^t\mathbf{q}^{-1}
 \end{aligned}$$

*Proof.* We simply verify the claim by plugging the rotated and translated quaternions and positions into  $E$ . For the first three terms in Equation 11.21 we obtain

$$\begin{aligned}\|\mathbf{q}'_v \bar{\mathbf{q}}_v^w - \mathbf{q}'_w\|^2 &= \|\mathbf{q}(\mathbf{q}_v \bar{\mathbf{q}}_v^w - \mathbf{q}_w)\|^2 \\ &= \|\mathbf{q}_v \bar{\mathbf{q}}_v^w - \mathbf{q}_w\|^2\end{aligned}$$

where we used property 11.23 and  $\|\mathbf{q}\| = 1$ . Using the same properties we obtain analogously

$$\begin{aligned}\|\mathbf{q}_v^{consist'}(1, \bar{\mathbf{c}}_v^w)^t (\mathbf{q}_v^{consist'})^{-1} - (1, \mathbf{x}'_w - \mathbf{x}'_v)^t\|^2 \\ &= \|\mathbf{q} \mathbf{q}_v^{consist} (1, \bar{\mathbf{c}}_v^w)^t (\mathbf{q}_v^{consist})^{-1} \mathbf{q}^{-1} - \mathbf{q}^{-1} (1, \mathbf{x}_w + t - \mathbf{x}_v + t)^t \mathbf{q}^{-1}\|^2 \\ &= \|\mathbf{q}_v^{consist} (1, \bar{\mathbf{c}}_v^w)^t (\mathbf{q}_v^{consist})^{-1} - (1, \mathbf{x}_w - \mathbf{x}_v)^t\|^2\end{aligned}$$

and

$$\begin{aligned}\|\mathbf{q}'_v - \mathbf{q}_v^{consist'}\|^2 &= \|\mathbf{q}(\mathbf{q}_v - \mathbf{q}_v^{consist'})\|^2 \\ &= \|\mathbf{q}_v - \mathbf{q}_v^{consist'}\|^2.\end{aligned}$$

As no constraints are given, the last term of the potential vanishes and thus the claim of the Lemma follows.  $\square$

Besides rotation and translation invariance, it is crucial that quaternion-based coordinates do in fact encode the complete shape information of a mesh. We therefore expect that the original surface can be reconstructed from the information encoded in our coordinates. As we reconstruct the surface by minimizing the potential  $E$ , the original mesh configuration should correspond to a unique global minimum. However, since we assume shape as invariant with respect to rigid transformations, we can only expect uniqueness up to a rotation and translation. This property of quaternion-based coordinates is asserted by the following

**Lemma 11.2.** *If any existing vertex constraints constrain vertices to their original position or orientation, the potential  $E$  vanishes for*

$$\mathbf{q}_v = \mathbf{q}_v^{consist} = \bar{\mathbf{q}}_v \quad \text{and} \quad \mathbf{x}_v = \bar{\mathbf{x}}_v$$

*This configuration is a global minimum of  $E$ . Moreover, if the mesh  $\mathcal{M}$  is connected, the constants  $\alpha_q, \alpha_{qc}, \alpha_x, \alpha_c > 0$  and at least one vertex position as well as one local frame orientation are constrained to their original value, this global minimum is unique.*

*Proof.* As the potential  $E$  is defined as residual of the Equations 11.17-11.20 and as the above choice of  $\mathbf{x}_v$  and  $\mathbf{q}_v$  satisfies these equations by definition the first

claim follows trivially. From the definition of the potential, also immediately follows  $E \geq 0$  so that this configuration is a global minimizer.

For uniqueness, we first remark, that any configuration that results in a vanishing potential  $E$  has to satisfy equations 11.17-11.20 as well as

$$\mathbf{q}_v = \mathbf{q}_v^{consist}$$

Otherwise, a positive residual in one of the terms in Equation 11.21 will remain and because of  $\alpha_q, \alpha_{qc}, \alpha_x, \alpha_c > 0$  will result in  $E > 0$ . By assumption, at least one vertex  $v_0 \in H$  exists that is constrained to its original local frame. From Equation 11.19 follows

$$\mathbf{q}_{v_0} = \bar{\mathbf{q}}_{v_0}.$$

For all vertices  $v$  with  $(v_0, v) \in \mathcal{M}$  we then obtain from Equations 11.17 and 11.4:

$$\mathbf{q}_v = \bar{\mathbf{q}}_{v_0} \bar{\mathbf{q}}_{v_0}^v = \bar{\mathbf{q}}_{v_0} \bar{\mathbf{q}}_{v_0}^{-1} \bar{\mathbf{q}}_v = \bar{\mathbf{q}}_v$$

By induction over the edges of the connected mesh  $\mathcal{M}$  we get  $\mathbf{q}_v = \bar{\mathbf{q}}_v$  for all vertices  $v \in R$ . From the local frames  $\bar{\mathbf{q}}_v$  and the local coordinates  $\bar{\mathbf{c}}_v^w$ , we obtain from Equations 11.18 and 11.5:

$$(1, \mathbf{x}_w - \mathbf{x}_v)^t = \frac{1}{\|\bar{\mathbf{q}}_v\|^2} \bar{\mathbf{q}}_v (1, \bar{\mathbf{c}}_v^w)^t \bar{\mathbf{q}}_v^{-1} = (1, \bar{\mathbf{x}}_w - \bar{\mathbf{x}}_v)^t \quad (11.25)$$

Again by assumption, at least one vertex  $v_1 \in H$  exists that is constrained to its original position, i.e.  $x_{v_1} = \bar{x}_{v_1}$ . From the relation 11.25 it now follows again by induction over the mesh edges, that all vertex positions are uniquely determined as

$$\mathbf{x}_v = \bar{\mathbf{x}}_v$$

which concludes the proof.  $\square$

Finally, we proof a property of the Hessian of the potential that will be needed in the optimization algorithm given in the following section.

**Lemma 11.3.** *The Hessians of the potential with respect to local frames  $\mathbf{q}_v$  and vertex positions  $\mathbf{x}_v$  are given by*

$$\frac{\partial^2 E}{\partial q_{vi} \partial q_{wj}} = A_{vwij}^1 + A_{wvji}^1$$

and

$$\frac{\partial^2 E}{\partial x_{vi} \partial x_{wj}} = A_{vwij}^3 + A_{wvji}^3.$$

*If the mesh  $\mathcal{M}$  is connected,  $\alpha_q, \alpha_{qc}, \alpha_x, \alpha_c > 0$  and at least one orientation constraint as well as one position constraint exists, both Hessians are positive definite.*

*Proof.* The equations for both Hessians follow easily from the derivatives  $\frac{\partial E}{\partial q_{vi}}$  and  $\frac{\partial E}{\partial x_{vi}}$  given in the last section. Both Hessians are constant in the unknowns  $(\mathbf{q}_v, \mathbf{q}_v^{consists}, \mathbf{x}_v)$ . Moreover, they depend only on the number of position and orientation constraints but not on their actual constraint value. We can thus assume without loss of generality, that the conditions of Lemma 11.2 are met. Positive definiteness then follows from the uniqueness of the global minimum.  $\square$

## 11.3 Efficient Non-linear Optimization

---

**Algorithm 2** The iterative reconstruction algorithm

---

Inputs:

- quaternion-based coordinates  $(\bar{\mathbf{q}}_v^w, \bar{\mathbf{c}}_v^w)_{(v,w) \in \mathcal{M}}$
- initial values  $\mathbf{q}_v^0, \mathbf{x}_v^0$  for  $v \in R$
- user specified constraints  $\mathbf{q}_v^{const}, \mathbf{x}_v^{const}$  for  $v \in H$

$\mathbf{q}_v \leftarrow \mathbf{q}_v^0$

$\mathbf{x}_v \leftarrow \mathbf{x}_v^0$

**repeat**

  step 1 (non-linear):  $\mathbf{q}_v^{consist} \leftarrow \arg \min_{\mathbf{q}_v^{consist}} E(\mathbf{q}_v^{consist}, \mathbf{q}_v, \mathbf{x}_v)$

  step 2 (linear):  $\mathbf{q}_v \leftarrow \arg \min_{\mathbf{q}_v} E(\mathbf{q}_v^{consist}, \mathbf{q}_v, \mathbf{x}_v)$

  step 3 (linear):  $\mathbf{x}_v \leftarrow \arg \min_{\mathbf{x}_v} E(\mathbf{q}_v^{consist}, \mathbf{q}_v, \mathbf{x}_v)$

**until** convergence

---

The potential  $E$  derived in the last section falls into the category of non-linear least squares problems. Such problems are usually solved by Gauss-Newton or Levenberg-Marquardt methods that proceed in an iterative manner solving a sparse linear system in each step. To solve these systems fast sparse direct solvers are commonly applied since they benefit from the fact, that the sparsity pattern of system matrices (in this case the Hessian) does not change. However, although the sparsity pattern is constant, the values of the Hessian do change, requiring a numerical refactorization in each step. In context of mesh editing, these iterated refactorizations becomes too costly to allow for interactive editing of medium-sized meshes.

As already lined out in the last section, we therefore take another approach to the minimization of the potential  $E$  that is based on alternating optimization. There, the deformation potential was already carefully chosen, to ensure that optimization with respect to each of the three sets of variables, namely the positions

$\mathbf{x}_v$ , frames  $\mathbf{q}_v$  and consistent frames  $\mathbf{q}_v^{consist}$  is computationally feasible. In particular, we will see that all large matrices involved in the optimization are constant over the iteration, which is in stark contrast to standard Newton or non-linear least squares solvers.

An overview of our reconstruction method is given in Algorithm 2. It alternates between three steps that correspond to the three above stated subsets of unknowns. Each step minimizes the potential energy  $E$  with respect to one of the variable subsets. The three optimization steps are iterated until eventually convergence is detected.

While the above given algorithm is conceptually very simple, it has two other important advantages: First it is guaranteed to converge, as each of the three steps lowers the overall energy  $E$  and the energy is bounded below by zero. Second, in a typical mesh editing scenario handle transformations usually change only little between two consecutive frames provided that the system runs at interactive frame rates. The simple iterative nature of our algorithm allows it to fully exploit this frame-to-frame coherence by reusing local frames and geometry of the last frame as initial values for the next. This is in contrast to multigrid methods like [BPGK06b] that need to optimize over a mesh hierarchy in each frame. In the following we will comment on the individual steps.

### 11.3.1 Linear Steps

We begin our discussion with steps two and three. In the second step we assume that for each vertex  $v$  both position  $\mathbf{x}_v$  and geometry consistent frame  $\mathbf{q}_v^{consist}$  are given and fixed and optimize the potential  $E$  for frames  $\mathbf{q}_v$ . Minimizing the potential with respect to local frames amounts to solving the linear system  $\frac{\partial E}{\partial q_{vi}} = 0$  which is equivalent to:

$$(A_{vwi j}^1 + A_{wvji}^1)q_{vj} = -A_{vwi j}^4 q_{wj}^{consist} - b_{vi}^1$$

Using an arbitrary one-to-one mapping  $\rho$  from vertex/coordinate index pairs to natural numbers  $1, 2, \dots, 4|R|$  the above linear system can be written as a matrix equation  $\mathbf{Ax} = \mathbf{b}$  with a  $4|R| \times 4|R|$  matrix  $\mathbf{A}$  and vector  $\mathbf{b}$  whose entries are given by

$$A_{\rho(v,i)\rho(w,j)} = A_{vwi j}^1 + A_{wvji}^1 \quad b_{\rho(v,i)} = -A_{vwi j}^4 q_{wj}^{consist} - b_{vi}^1$$

From the definition of  $A_{vwi j}^1$  we see, that the matrix  $\mathbf{A}$  is sparse. The number of non-zero entries per row is at most  $4val(v) + 1$  where  $val(v)$  denotes the number of edges at vertex  $v$ . As shown in Lemma 11.2.4, it corresponds to Hessian of the potential with respect to local frames  $\mathbf{q}_v$  and is symmetric positive definite which guarantees, that the above linear system thus has a unique solution.

Moreover, the matrix is constant with respect to all unknowns  $\mathbf{q}_v$ ,  $\mathbf{q}_v^{consist}$  and  $\mathbf{x}_v$ . This important property allows a factorization into two sparse triangular factors in a precomputation step using sparse LU or Cholesky factorizations methods. In our case we use the freely available SuperLU package [DGL97] for this factorization, which we found to perform slightly faster than the sparse Cholesky solver TAUCS [To103]. Once factored, the above linear system can be efficiently solved for varying right hand side vectors  $\mathbf{b}$  by a very fast triangular solver, which namely amounts to two simply back- and forward substitutions.

The third step of the algorithm is very similar two the second step. This time, we keep local frames  $\mathbf{q}_v$  and  $\mathbf{q}_v^{consist}$  fixed and minimize the potential with respect to optimal vertex positions. Again, Lemma 11.2.4 asserts that the potential is quadratic in these positions with a unique minimum (constant positive definite Hessian). It is obtained by solving  $\frac{\partial E}{\partial x_{vi}} = 0$  which amounts to:

$$(A_{vwi}^3 + A_{wvi}^3)x_{vj} = 2 \frac{q_{uj}^{consist} (\mathcal{M}_{uv} + \mathcal{M}_{vw}) \rho_{jk(i+1)} \bar{C}_{uwm} q_{um}^{consist}}{(q_{ul}^{consist} q_{ul}^{consist})^2}$$

In analogy two the second step, this linear system can be written in matrix form using the mapping  $\rho$  which results in a sparse square matrix of dimension  $3|R|$  with exactly  $val(v) + 1$  entries in each row and column. We also factor this matrix into two triangular matrix factors in the precomputation. In each iteration, only the right hand side vector has to be updated and the system is solved by a fast triangular solver.

### 11.3.2 Non-linear Step

As already pointed out, the potential  $E$  is not quadratic in the consistent frames  $\mathbf{q}_v^{consist}$  which is due to the second term in Equation 11.21. However, a closer observation of this term reveals, that the consistent local frame of each vertex  $v$  contributes to only one addend (This is in contrast to e.g. the frames  $\mathbf{q}_v$  which are coupled in the energy by the first term). As the same observation holds for the third term in Equation 11.21, the minimization of  $E$  with respect to  $\mathbf{q}_v^{consist}$  can be carried out for each vertex individually. This has two important consequences: First, the dimensionality of the optimization problem at each vertex is low. Namely, at each vertex  $v$  only the four components  $q_{vi}^{consist}$  have to be optimized. Second, we can make use of parallelization, i.e. solve for consistent frames at all vertices in parallel.

To actually solve for optimal consistent frames, a standard Levenberg-Marquardt based non-linear least squares solver can be used. Alternatively, an even more efficient and conceptually simpler solution to the problem is available if we neglect for a moment the third term in Equation 11.21, i.e. the coupling of  $\mathbf{q}_v$  and  $\mathbf{q}_v^{consist}$

within the non-linear step. Computing  $q_v^{consist}$  then amounts to the optimal solution of the equation

$$\frac{1}{\|\mathbf{q}_v^{consist}\|^2} \mathbf{q}_v^{consist} (1, \bar{\mathbf{c}}_v^w)^t (\mathbf{q}_v^{consist})^{-1} = (1, \mathbf{x}_w - \mathbf{x}_v)^t \quad (11.26)$$

in least squares sense for a fixed positions  $\mathbf{x}_v$ . An efficient solution to this problem was given by Horn in [Hor87] that essentially only requires to extract the eigenvector to the largest eigenvalue from a four by four matrix.

To account, at least approximately, for the third term of the potential, the result obtained by Horn's algorithm has to be averaged in some sense with the local frames  $\mathbf{q}_v$ . However, the second step in our algorithm does in fact correctly account for the third term and thus performs the desired averaging. A simple approximate solution is therefore, to set  $\mathbf{q}_v^{consist}$  to the result of the second step (namely  $\mathbf{q}_v$ ) after the second step. The simplified algorithm is summarized in Algorithm 3

---

**Algorithm 3** The simplified reconstruction algorithm

---

Inputs: same as in the exact Algorithm 2.

**repeat**

step 1 (non-linear): compute  $\mathbf{q}_v^{consist}$  from  $\mathbf{x}_v$  using Horn's method

step 2a (linear):  $\mathbf{q}_v \leftarrow \arg \min_{\mathbf{q}_v} E(\mathbf{q}_v^{consist}, \mathbf{q}_v, \mathbf{x}_v)$

step 2b:  $\mathbf{q}_v^{consist} \leftarrow \mathbf{q}_v$

step 3 (linear):  $\mathbf{x}_v \leftarrow \arg \min_{\mathbf{x}_v} E(\mathbf{q}_v^{consist}, \mathbf{q}_v, \mathbf{x}_v)$

**until** convergence

---

The simple convergence argument given above does not hold for the simplified algorithm, as theoretically, the overall potential energy  $E$  can increase in the first step. Even though it is difficult to formally proof convergence of the simplified algorithm, we observed in our experiments that the reconstruction converges after only a few iterations even for large handle transformations. Besides its higher performance and conceptual simplicity, the simplified algorithm can also be further modified, to simulate the deformation behaviour of different materials as explained in the following section.

### 11.3.3 Parameterizing Deformation

By construction the above described framework produces conformal deformation. In Figure 11.2 the two handles on both sides of the plate were pushed apart by a translation along the x-axis. Interestingly, the mesh also stretches in the perpendicular direction to preserve the local shape of each 1-ring and thus produces a conformal deformation.

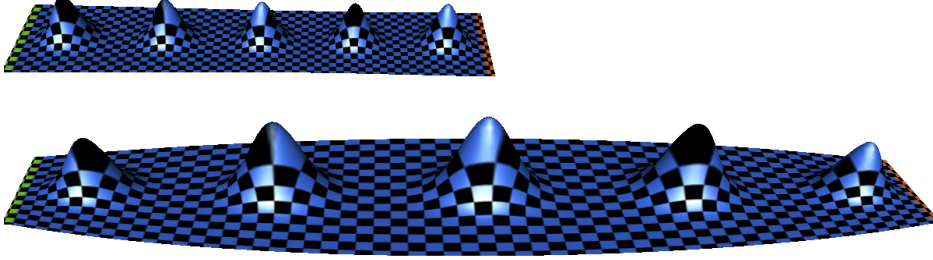


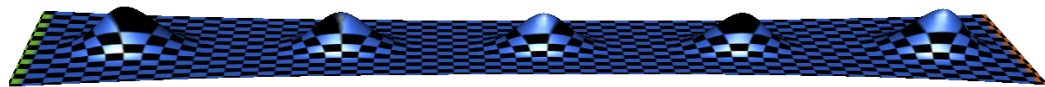
Figure 11.2: On top a undeformed surface strip is shown. The bottom shows the resulting deformation, if both handles (shown in green and red) are pushed apart along the horizontal direction.

In reality only few materials (so called auxetic materials) show a behavior as depicted in the figure. As explained in Section 3.1.2, linear elasticity characterizes these materials by a negative Poisson’s ratio which means that these materials when stretched in one direction also stretch in perpendicular directions. In contrast, most existing materials like metals, rubber or textiles have a positive Poisson’s ratio and correspondingly shrink along unconstrained directions when anisotropically stretched (see Figure 11.3a).

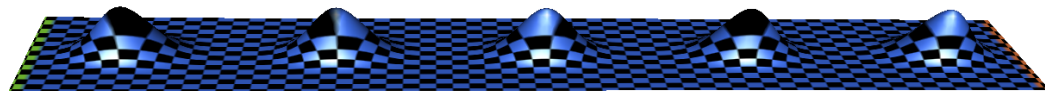
With a small modification, the simplified algorithm described in the last section can be extended to yield mesh deformations similar to those observed for these materials. To this extent we reconsider the update of  $\mathbf{q}_v^{consist}$  in step 1 of Algorithm 3: As described in the last section, Horn’s method is used to compute at each vertex a quaternion  $\mathbf{q}_v^{consist}$  that solves Equation 11.26 in least squares sense. This quaternion represents a rotation  $\mathbf{R}_v$  as well as a isotropic scaling by factor  $s_v = \|\mathbf{q}_v^{consist}\|^2$ . Horn shows in [Hor87] that these two components can be determined independently, so that the rotated and scaled original 1-ring edges (represents by  $\bar{\mathbf{c}}_v^w$ ) fit to the deformed 1-ring edges (given by  $\mathbf{x}_w - \mathbf{x}_v$ ) in least squares sense. Pushing the handles apart as shown in Figure 11.2 causes an anisotropic stretch along the x-axis in each 1-ring. Therefore the optimal isotropic scale  $s_v$  computed in step 1 of the algorithm will be larger than 1. As a consequence, all quaternions tend to have norm larger than one in step 2a and step 2b. This in turn causes a stretch in the unconstrained directions in step 3 and thus leads to the above described behavior resembling auxetic materials shown in the figure.

Since in the example of Figure 11.2 stretch along the x-axis is fixed by the handle constraints on both sides, the isotropic scale of  $\mathbf{q}_v^{consist}$  determines the stretch along the unconstrained y- and z-axis. To obtain a deformation behavior similar to those of materials with positive Poisson’s ratio we therefore replace the actual scale  $s_v$  of  $\mathbf{q}_v^{consist}$  by  $s_v^\alpha$  in step 1 of the simplified algorithm. For negative  $\alpha$  we

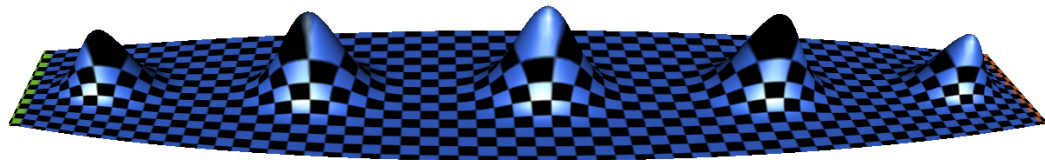




(a)



(b)



(c)

Figure 11.3: By varying the parameter  $\alpha$  the behaviour of materials with different Poisson's ratio can be simulated: For  $\alpha = -1$  the mesh deforms similar to materials with high Poisson's ratio like e.g. rubber. For  $\alpha = 0$  the deformation resembles materials like cork, whose Poisson's ratio is near zero. Choosing  $\alpha = 1$  yields the behaviour of auxetics which are characterized by negative Poisson's ratios.

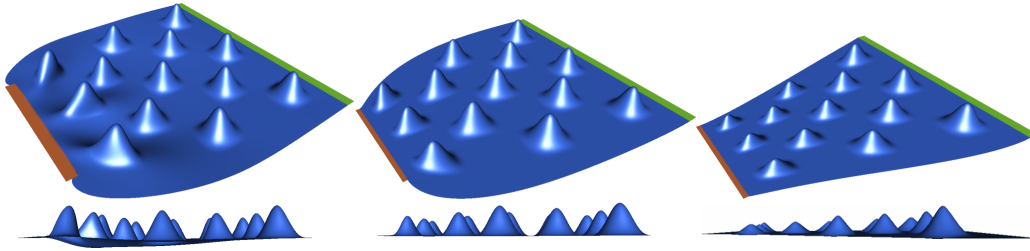


Figure 11.4: Deformations obtained for pure scaling by 0.5. From left to right: a) Primo, b) our method with  $\alpha = 0$  and c) our method in free-scale mode

then obtain the desired shrinking in unconstrained directions as shown in Figure 11.3a while  $\alpha = 1$  results in the original conformal behavior. Figure 11.3b shows the result for the special case of  $\alpha = 0$  that keeps  $\mathbf{q}_v^{consist}$  normalized so that stretching along the x-axis does not influence the other directions. Such behavior can e.g. be observed for cork which has a Poisson's ratio of nearly zero. As shown in the figure the parameter  $\alpha$  in our method has therefore a similar interpretation as the Poisson's ratio. Although our method is not physically motivated, with the above extension the resulting deformations appear physically plausible for different materials. Therefore, its deformation behavior is very intuitive as long as handle transformation does not include scaling.

For scaling, however, there exists no physical counterpart: While real materials can be pulled or bend they cannot be scaled. As demonstrated in Figure 11.4a physically plausible methods like Primo [BPGK06b] react to scaling in a way that is intuitive but might yet be undesired. In the figure the case of a pure scaling of the handle by 0.5 is shown. Both Primo and our approach result in a creasing near the scaled handle while the bumps approximately keep their height. In some cases, however, the user might desire that features as the bumps near the handle also scale down.

We therefore added a further editing mode that handles scaling differently. In this mode we give up scale consistency between frames and geometry to some extent. More precisely, we allow frames  $\mathbf{q}_v$  to scale independently of the actual geometry while still enforcing consistent orientation. However, we only break scale coupling in one direction: we suppress feedback from geometry scale to handle scale, while still encouraging the actual edges  $\mathbf{x}_w - \mathbf{x}_v$  to adapt the scale of the frames  $\mathbf{q}_v$ . This can easily be achieved by rescaling  $\mathbf{q}_v^{consist}$  after step 1 in Algorithm 3 so that it has the same length as  $\mathbf{q}_v$ . As shown in 11.4c this free-scale mode distributes changes in scale over the mesh evenly. Therefore, features near the scaled handle scale down as intended. By keeping frame orientations

consistent the algorithm still preserves curvature and tangential derivatives in this mode. This can be seen from the slight S-shape of the boundaries in the top of [11.4c](#).

Please note that, both the parameter  $\alpha$  and the editing mode do not affect the system matrices in the reconstruction algorithm and can therefore be changed interactively during editing without compromising the performance.

### 11.3.4 Quaternion Distance Measures

In the definition of our potential energy  $E$  in Equation [11.21](#) the first, third and the constraint term involve a distance measure on quaternions. In all three cases, the Euclidean norm in  $\mathbb{R}^4$  is used on the difference of quaternions to measure distances between quaternions, i.e.

$$d(\mathbf{q}_1, \mathbf{q}_2) := \|\mathbf{q}_1 - \mathbf{q}_2\|$$

for two quaternions  $\mathbf{q}_1$  and  $\mathbf{q}_2$ . While this distance measure enables an efficient least squares formulation, it is in general not suitable to measure differences in orientation [[BB06](#)]. In contrast, in case of pure rotations represented by unit quaternions the distance measure

$$d_{rot} := \arccos \langle \mathbf{q}_1 | \mathbf{q}_2 \rangle, \tag{11.27}$$

measuring the angle between the two quaternions in 4D-space, is well established in computer graphics [[Sho85](#)].

A subtle complication arises due to the fact, that unit quaternions represent a double cover of the rotation group  $SO(3)$ , i.e.  $\mathbf{q}$  and  $-\mathbf{q}$  describe the same rotation. Thus, the comparison of the rotations has to consider all possible combinations of negated and not negated quaternions. The according distance measure has the following form:

$$d_{rot} := \min \{ \arccos \langle \mathbf{q}_1 | \mathbf{q}_2 \rangle, \arccos \langle \mathbf{q}_1 | -\mathbf{q}_2 \rangle, \arccos \langle -\mathbf{q}_1 | \mathbf{q}_2 \rangle, \arccos \langle -\mathbf{q}_1 | -\mathbf{q}_2 \rangle \} \tag{11.28}$$

For symmetry reasons, this can be reduced to

$$d_{rot} = \min \{ \arccos \langle \mathbf{q}_1 | \mathbf{q}_2 \rangle, \arccos \langle \mathbf{q}_1 | -\mathbf{q}_2 \rangle \}. \tag{11.29}$$

In our case, we are concerned with combinations of rotations and isotropic scaling as represented by non-unit quaternions. An appropriate metric should certainly consider the angle of the rotation as well as the scaling factor of each transformation. The distance  $d_{rot}$  is not a good metric in this case, because the

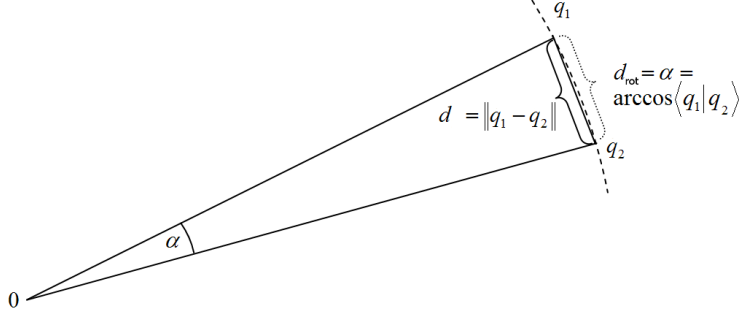


Figure 11.5: In case of small rotation angles, the simple Euclidean distance of quaternions provides a sufficient approximation of  $d_{rot}$ .

scaling factors of the transformations are not properly compared to each other. In contrast, the much simpler distance measure

$$\begin{aligned} d_{min}(\mathbf{q}_1, \mathbf{q}_2) &:= \min \{ \|\mathbf{q}_1 - \mathbf{q}_2\|, \|\mathbf{q}_1 - (-\mathbf{q}_2)\| \} \\ &= \min \{ \|\mathbf{q}_1 - \mathbf{q}_2\|, \|\mathbf{q}_1 + \mathbf{q}_2\| \} \end{aligned} \quad (11.30)$$

considers rotation as well as scaling. As illustrated in Figure 11.5,  $d_{min}$  is an approximation of  $d_{rot}$  for unit quaternions if angles between  $\mathbf{q}_1$  and  $\mathbf{q}_2$  are small.

However, using  $d_{min}$  instead of  $d$  in the definition of the potential  $E$  is not advisable as it breaks the linearity of the second and third step in the optimization algorithm. The following two Lemmas, which are taken from [Par07], examine under which conditions  $d$  and  $d_{min}$  are equivalent. We then propose a few modifications to the reconstruction algorithm that enforce these conditions.

**Lemma 11.4.** *If the angle  $\alpha/2 := \angle(\mathbf{q}_1, \mathbf{q}_2)$  between the non-unit quaternions  $\mathbf{q}_1$  and  $\mathbf{q}_2$  is less than  $\pi/2$ , the metrics  $d_{min}$  and  $d$  coincide.*

*Proof.* As shown in Figure 11.6, the law of cosines allows to rewrite the distances  $\|\mathbf{q}_1 - \mathbf{q}_2\|$  and  $\|\mathbf{q}_1 + \mathbf{q}_2\|$  in the following way:

$$\begin{aligned} \|\mathbf{q}_1 - \mathbf{q}_2\|^2 &= \|\mathbf{q}_1\|^2 + \|\mathbf{q}_2\|^2 - 2\|\mathbf{q}_1\|\|\mathbf{q}_2\|\cos\frac{\alpha}{2} \\ \|\mathbf{q}_1 + \mathbf{q}_2\|^2 &= \|\mathbf{q}_1\|^2 + \|\mathbf{q}_2\|^2 - 2\|\mathbf{q}_1\|\|\mathbf{q}_2\|\cos\left(\pi - \frac{\alpha}{2}\right) \end{aligned}$$

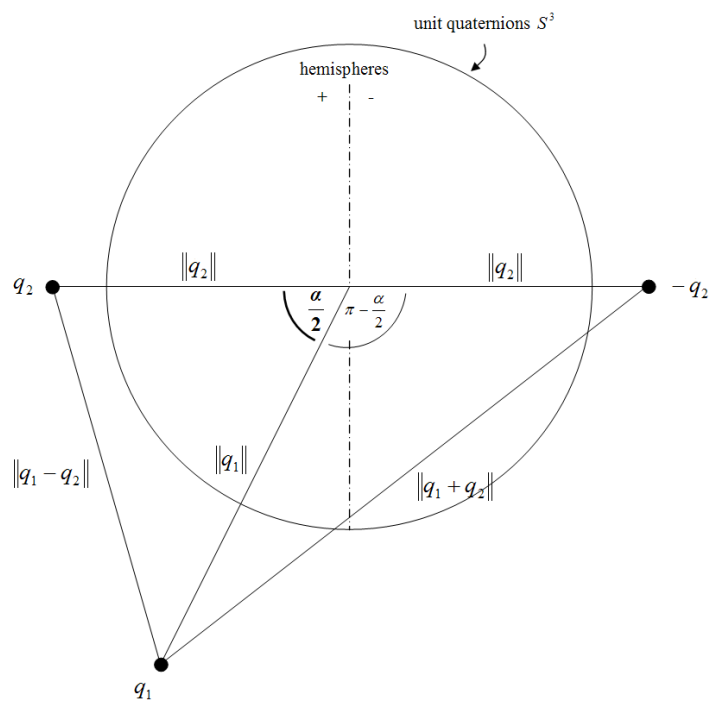


Figure 11.6: Notation for Lemma 11.4: The space of unit quaternions forms a unit sphere  $S^3$  in  $\mathbb{R}^4$ . A 2D-plane is defined by  $q_1$ ,  $q_2$ , and  $-q_2$  that contains the origin. Cutting the sphere  $S^3$  with this plane, we get a two-dimensional slice of the sphere, which is illustrated above.

then we have:

$$\begin{aligned}
 & |\alpha| < \pi \\
 \Rightarrow & \cos \frac{\alpha}{2} > \cos \left( \pi - \frac{\alpha}{2} \right) \\
 \Leftrightarrow & -2 \|\mathbf{q}_1\| \|\mathbf{q}_2\| \cos \frac{\alpha}{2} < -2 \|\mathbf{q}_1\| \|\mathbf{q}_2\| \cos \left( \pi - \frac{\alpha}{2} \right) \\
 \Leftrightarrow & \|\mathbf{q}_1 - \mathbf{q}_2\|^2 < \|\mathbf{q}_1 + \mathbf{q}_2\|^2 \\
 \Rightarrow & \min \{ \|\mathbf{q}_1 - \mathbf{q}_2\|, \|\mathbf{q}_1 + \mathbf{q}_2\| \} = \|\mathbf{q}_1 - \mathbf{q}_2\| \\
 \Leftrightarrow & d_{\min}(\mathbf{q}_1, \mathbf{q}_2) = d(\mathbf{q}_1, \mathbf{q}_2)
 \end{aligned}$$

□

Lemma 11.4 shows that  $d = d_{\min}$ , if the angle of between the quaternions  $\mathbf{q}_1$ ,  $\mathbf{q}_2$  is less than  $\pi/2$ . In the following Lemma, we will establish a connection to the rotation between the orientations described by  $\mathbf{q}_1$  and  $\mathbf{q}_2$ :

**Lemma 11.5.** *Let  $\mathbf{q}_1, \mathbf{q}_2$  be quaternions describing two orientations (with isotropic scale). We consider the transformation  $\mathbf{q}_1^{-1}\mathbf{q}_2$  that rotates and scales  $\mathbf{q}_1$  into  $\mathbf{q}_2$ . If the angle of the rotation of this transformation (around some axis  $\mathbf{v}$ ) is denoted by  $\alpha$ , then*

$$\arccos \frac{\langle \mathbf{q}_1 | \mathbf{q}_2 \rangle}{\|\mathbf{q}_1\| \|\mathbf{q}_2\|} = \alpha/2, \quad (11.31)$$

*i.e. the angle between the quaternions  $\mathbf{q}_1, \mathbf{q}_2$  is half the rotation angle between the according orientations.*

*Proof.* The angle of the rotation between the orientations is not changed by normalizing the two quaternions:

$$\begin{aligned}
 \mathbf{q}'_1 & := \mathbf{q}_1 / \|\mathbf{q}_1\| \\
 \mathbf{q}'_2 & := \mathbf{q}_2 / \|\mathbf{q}_2\|
 \end{aligned}$$

The rotation  $\mathbf{q}_3$  between the orientations of the unit quaternions  $\mathbf{q}'_1 = (q_0^1, q_x^1, q_y^1, q_z^1)^t$  and  $\mathbf{q}'_2 = (q_0^2, q_x^2, q_y^2, q_z^2)^t$  is given as  $\mathbf{q}_3 := (\mathbf{q}'_1)^{-1} \cdot \mathbf{q}'_2$ . This is again a unit quaternion and corresponds to a rotation of angle  $\alpha = 2 \arccos(q_{31})$  around the axis  $\mathbf{v} = (q_{32}, q_{33}, q_{34})^t$ . One easily verifies that

$$q_{31} = \langle \mathbf{q}'_1, \mathbf{q}'_2 \rangle$$

and thus

$$\alpha/2 = \arccos \frac{\langle \mathbf{q}_1 | \mathbf{q}_2 \rangle}{\|\mathbf{q}_1\| \|\mathbf{q}_2\|}$$

□

In summary, we conclude that the distances  $d(\mathbf{q}_1, \mathbf{q}_2)$  and  $d_{min}(\mathbf{q}_1, \mathbf{q}_2)$  coincide, as long as the angle  $\alpha$  of the rotation, that rotates the orientation represented by  $\mathbf{q}_1$  onto the orientation represented by  $\mathbf{q}_2$  is smaller than  $\pi$ . Note, that this corresponds to a very drastic difference in the orientation.

### Consequences for the mesh editing algorithm

As elaborated in the previous section, the approximation of  $d_{min}(\mathbf{q}_1, \mathbf{q}_2)$  by  $d(\mathbf{q}_1, \mathbf{q}_2)$  is only reasonable if the angle enclosed by the quaternions  $\mathbf{q}_1$  and  $\mathbf{q}_2$  in  $\mathbb{R}^4$  is smaller than  $\pi/2$ . If this condition is not met, it can be enforced by simply negating one of the two quaternions. We refer to this negations in the following as a flip.

We therefore suggest the following modifications to Algorithm 3: When setting up the system matrices for step 2a and 3 in the precomputation, care must be taken to select  $\bar{\mathbf{q}}_v^w$  from the two quaternions representing  $\bar{F}_v^w$  so that the angle between  $\bar{\mathbf{q}}_w$  and  $\bar{\mathbf{q}}_v * \bar{\mathbf{q}}_v^w$  is minimal. In theory, it may become necessary in the iterations to apply further flips to  $\bar{\mathbf{q}}_v^w$  at some edges to ensure this condition for the optimal frames  $\mathbf{q}_v$ . This would require a refactorization of the matrix. Fortunately, according to Lemma 11.5 such a flip is only necessary if the rotation angle of the transformation that rotates  $\mathbf{q}_w$  onto  $\mathbf{q}_v * \bar{\mathbf{q}}_v^w$  is greater than  $\pi$ . As this rotation angle is zero in the undeformed state for all edges, a violation of the above condition is an extraordinary event. Please note that sharp features or vertices with high curvature do not increase the probability of such a flip as the relevant distance is measured between  $\mathbf{q}_w$  and  $\mathbf{q}_v * \bar{\mathbf{q}}_v^w$  and not directly between  $\mathbf{q}_w$  and  $\mathbf{q}_v$ . Therefore, a flip becomes in practice only necessary for extreme deformations in conjunction with unreasonable low mesh resolution, that would anyway require mesh subdivision in order to adequately represent the deformed surface. Although our implementation detects the necessity of  $\bar{\mathbf{q}}_v^w$  flips, we never encountered this case in our experiments.

Similarly we flip  $\mathbf{q}_v^{consist}$  and  $\mathbf{q}_v^{const}$  in step 2a and 3 of each iteration so that the angular distance to  $\mathbf{q}_v$  is minimal. These flips are not problematic since both variables appear only in the right hand side vectors and not in the system matrices.

### 11.3.5 Implementation

In this section we give some noteworthy details on our GPU-based implementation of the simplified algorithm for constrained reconstruction and comment on some technical details. As described in Section 11.3 the estimation of consistent frames  $\mathbf{q}_v^{consist}$  amounts to the solution of a set of independent low-dimensional problems and thus naturally lends itself to a parallel implementation. The method

of [Hor87] essentially requires to setup covariance matrices for vertex neighborhoods and to compute the eigenvector to the largest eigenvalue of a four by four matrix. For an easy implementation on graphics hardware we chose the NVIDIA CUDA Framework and adapted an off-the-shelf eigenvalue decomposition which required only minor modifications. Although we did not optimize this code for parallelism or memory access it is fast enough for our purpose and only amounts for less than 8 percent of the overall reconstruction time. Our parallel implementation of the eigenvalue and the right hand side vector calculation runs about 80 times faster compared to an equivalent CPU implementation.

With a fast parallel estimation of consistent frames at hand, we strove for an efficient linear solver that is easy to implement on graphics hardware. To this extent we tried a conjugate gradient method that naturally lends itself to sparse matrices and parallel execution. However, to our knowledge by now only simple diagonal preconditioners have been implemented on graphics hardware [BFGS03] that result in a high number of iterations for our matrices.

In contrast, sparse direct solvers are particularly suited for our method as they exploit the fact, that our system matrices are constant. The mayor computational effort is spent in a precomputation step. During interactive editing they only solve two triangular systems by backsubstitution, which is far more efficient than iterative methods. Although backsubstitution on a single CPU is very fast, an efficient parallel implementation is challenging and we are currently not aware of an implementation on graphics hardware. Our implementation therefore uses a CPU based backsubstitution from the SuperLU package [DGL97] to solve the linear systems in step 2 and 3 of the reconstruction algorithm. Taken together the backsubstitutions for the two linear systems currently make up nearly 90 percent of the reconstruction time.

Due to its iterative nature, our algorithm naturally benefits from frame-to-frame coherence during interactive mesh editing. To exploit frame-to-frame coherence to its full extend, our actual implementation differs slightly from Algorithm 3. Instead of fixing the handle transformation in the iterative optimization, our implementation uses an updated handle transformation in each step. In addition, the mesh is also rendered in each step to provide an immediate feedback. In this way the responsiveness of the interactive reconstruction is further increased. The user can change or correct handle transformation in each step and does not have to wait until the iteration fully converges.



## 12.1 Experimental Results

To demonstrate the quality of the deformations obtained with our method we applied it to the test cases from [BS08]. The examples in that survey were chosen to reveal limitations of linear approaches, i.e. existing linear methods show gross artifacts on at least one of these examples. Our results are shown in Figure 12.1 in direct comparison to the results of the non-linear Primo method [BPGK06b]. As it can be seen the deformations are physically plausible and comparable in quality to those of Primo. For these test cases we set  $\alpha = 0$ .

On the cylinder example the two methods show slightly different bending. While for our method the bending is more smooth the Primo result seems to preserve the boundary frames better. As described in Section 11.2.2, we used soft constraints to express the orientation and positional constraints in our potential energy. Increasing or decreasing the influence  $\alpha_c$  in Equation 11.21 yields an additional intuitive way to change deformation behavior. If desired, our method can adapt a similar behavior to Primo by either increasing the constraint weighting or by using hard constraints.

Even though our interactive reconstruction algorithm is relatively simple compared to the considerably more complex GPU based multi resolution optimizations of Primo [BPGK06a] or subspace domain methods like [HSL<sup>+</sup>06], it is fast enough to enable interactive editing of large models: Figure 12.2 shows the result of two single step editings of the crouching dragon model that each took only about 2 minutes of user time. Please note, that in this editing the region of interest included the whole mesh with more than 100k triangles.

Figure 12.3 shows a step-by-step deformation of an even larger model with more than 300k triangles. First, the leg is bent by moving and rotating the handle,

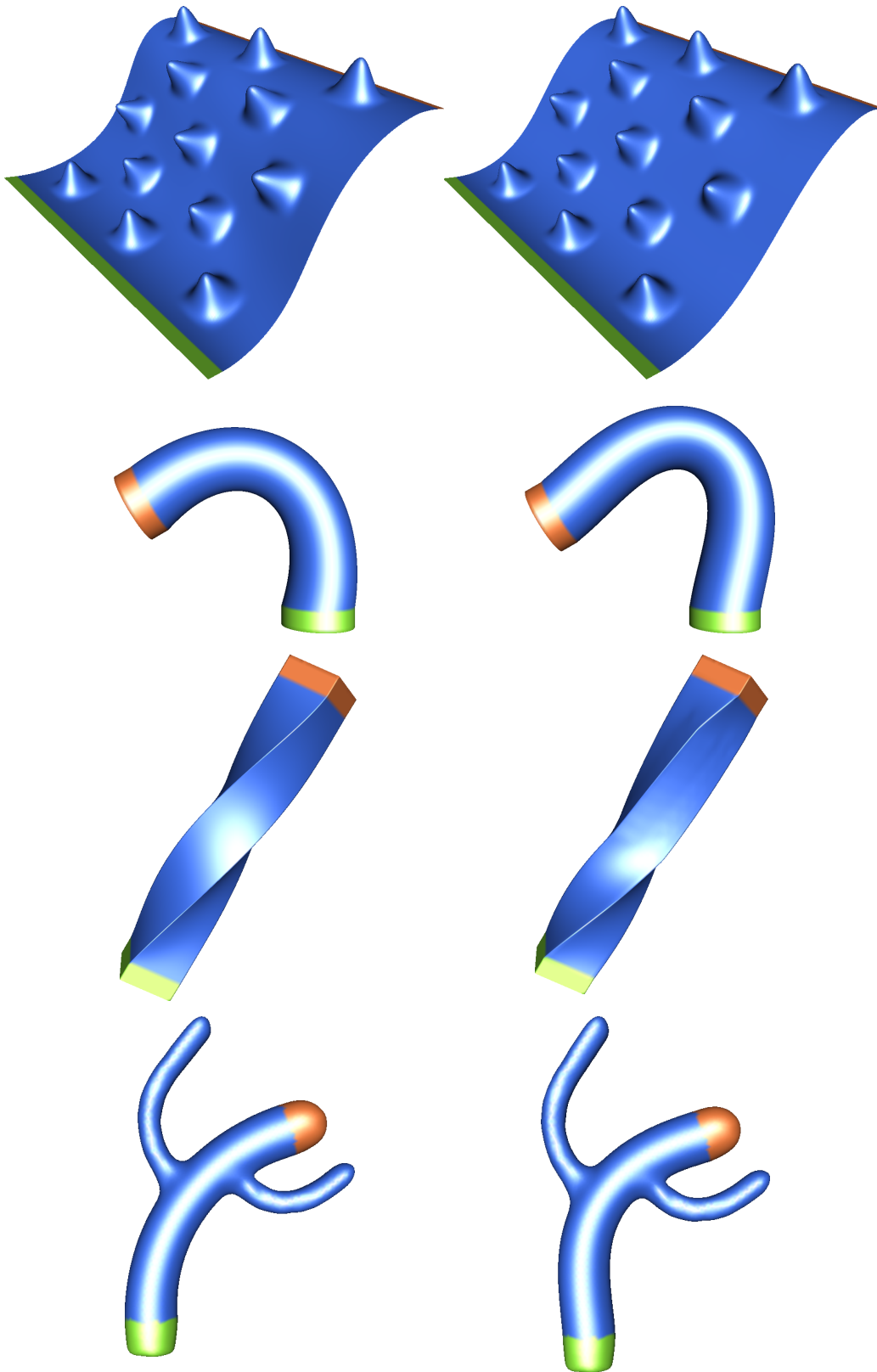


Figure 12.1: Our results (left) compared to results of Primo[BPGK06b] (right) for pure translation, a  $120^\circ$  bend, a  $135^\circ$  twist and a  $70^\circ$  bend of different objects

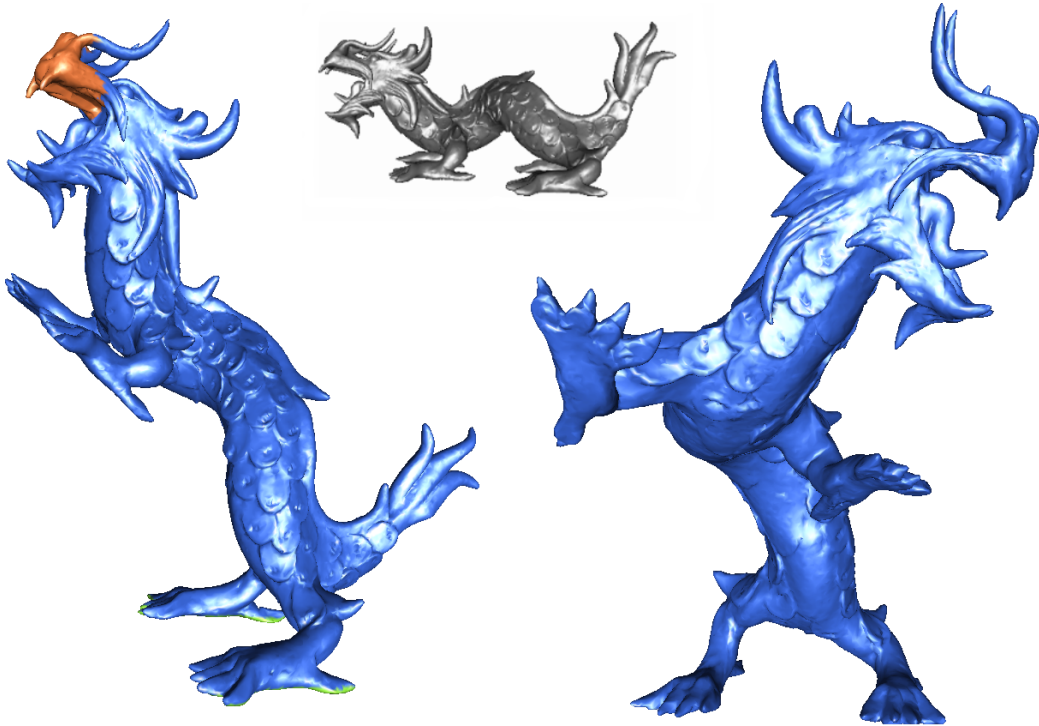


Figure 12.2: Two mesh editing operations on the crouching dragon model. The original model is shown on the top in gray.

which is located at one of the Armadillo's toes, drawn in red. This is repeated with his left claw, and finally the tip of his nose is used to rotate his head. In every step, a small handle consisting of only a few vertices is sufficient to achieve intuitive results. The user time for such a stepwise modeling operation is certainly higher because the user has to select the region of interest and handles anew after each step. On the other hand, there are two advantages of such a procedure: First, by selecting only a part of the mesh in each step, the region of interest is comparatively small allowing for interactive mesh editing of huge models. Second, the modeler has more control over the deformation as parts outside the region of interest do not change. On both the Dragon as well as the Armadillo model our method runs at interactive frame rates as demonstrated in the accompanying video.

The dinosaur model shown in Figure 12.4 is not as complex as the crouching dragon or the Armadillo. Nevertheless, it is a good example for showing how details on the surface of the model are preserved. The example on the lower right demonstrates this rather clearly; the ripples of the dinosaur are preserved even

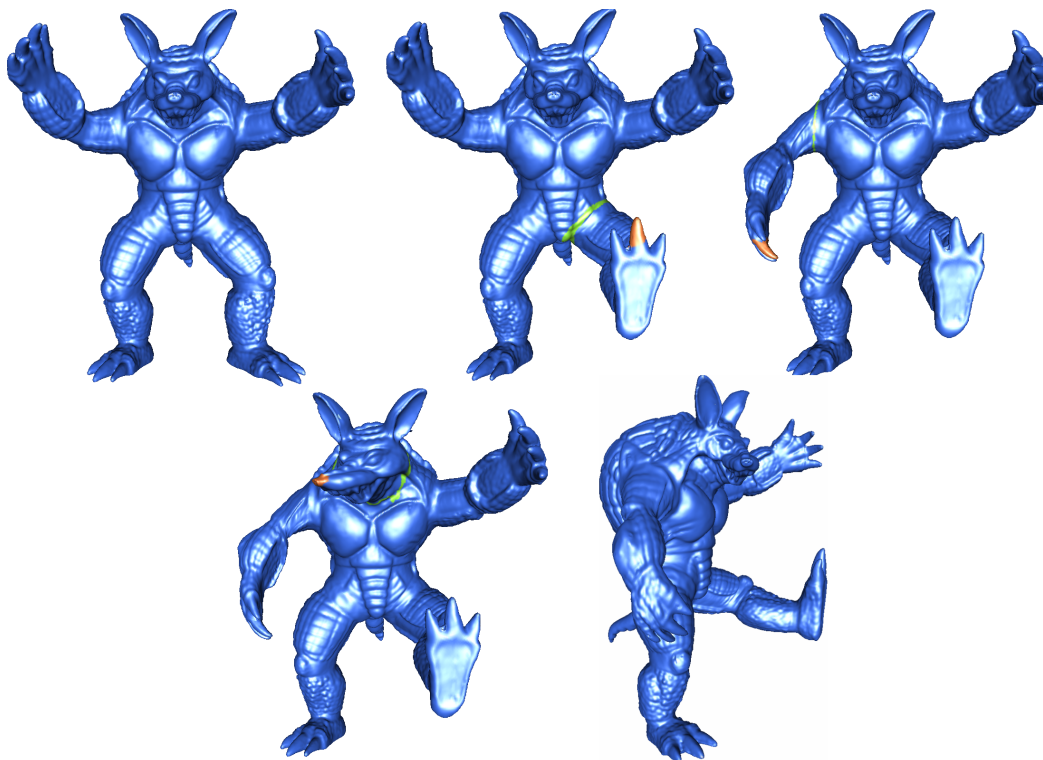


Figure 12.3: An editing of the Armadillo model in 3 steps: moving and rotating one toe, one claw and the tip of his nose.

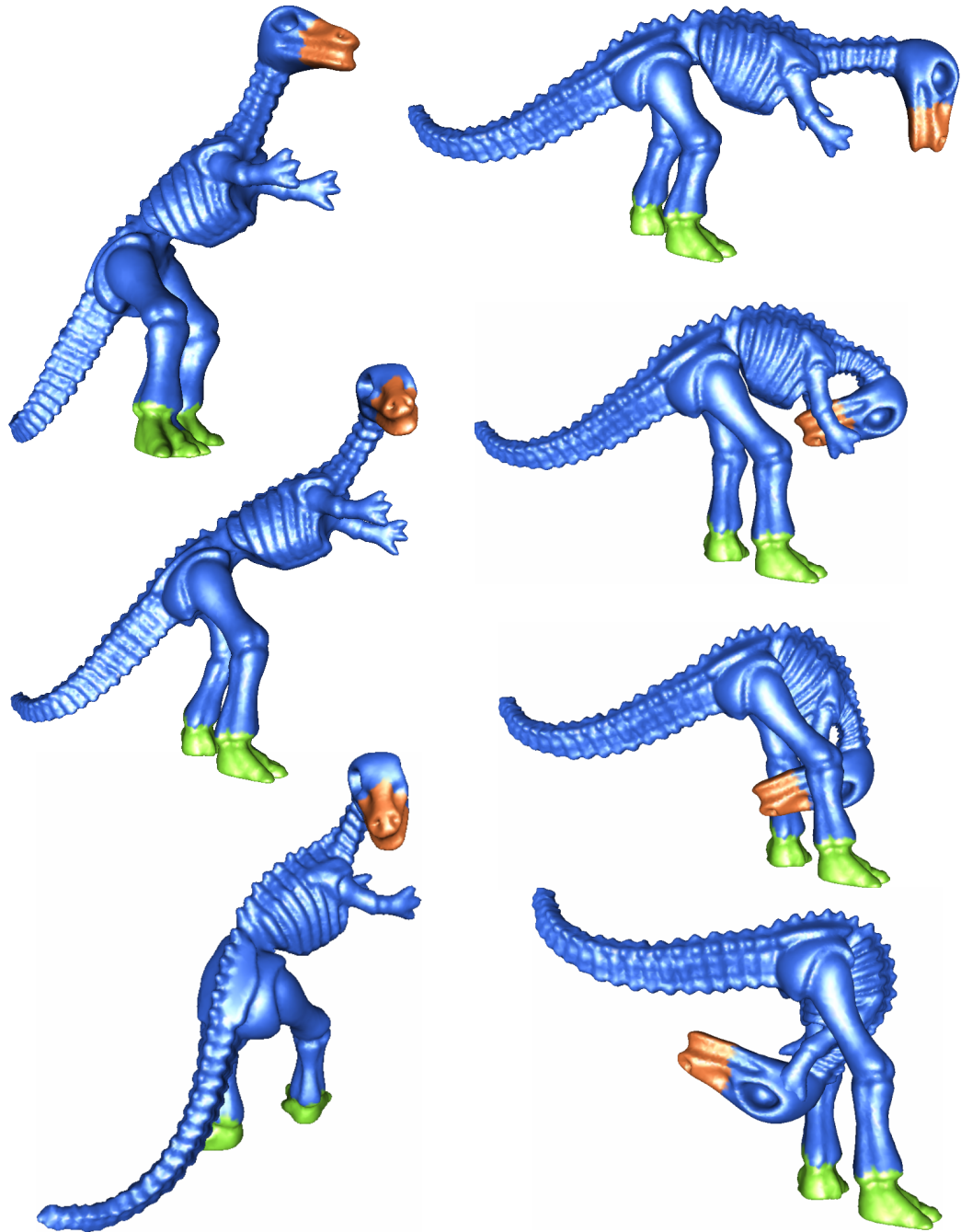


Figure 12.4: Various deformations of the dinosaur model. The original model is the one on the upper left.

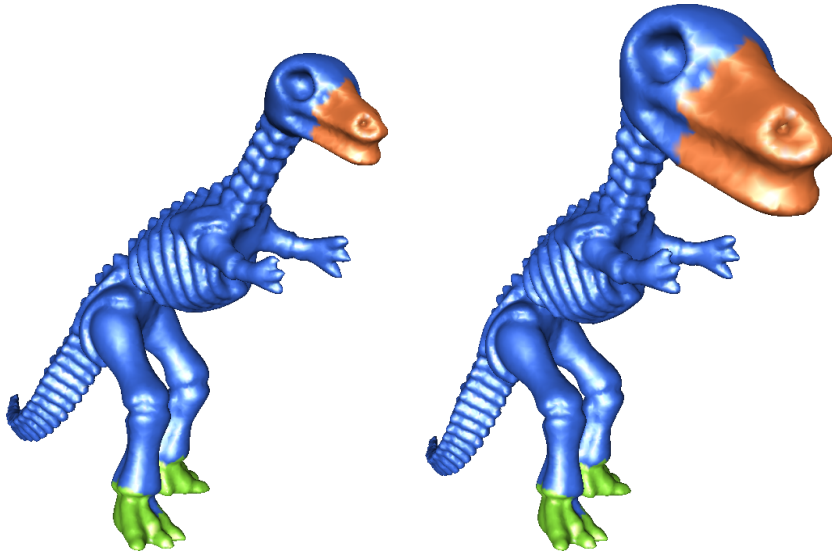


Figure 12.5: A simple deformation obtained by scaling the head of the dinosaur.

under extreme deformations. In addition, unconstrained parts of the model, such as the tail and the arms, behave in a plausible way. Figure 12.5 shows the result of a mesh editing operation involving only the scaling of the handle.

As lined out in Section 11.3.5, the computation time for a single iteration is largely dominated by the triangular solver, i.e. the forward- and backsubstitution algorithms. The complexity of these algorithms is linear in the number of non-zero entries in the triangular factors. This in turn seems to be roughly proportional to the number of vertices in the region of interest. The total number of iterations needed for convergence is hard to derive analytically. While this number theoretically depends on the shape of the region of interest, mesh connectivity as well as on the shape of handles, in our experiments convergence was detected after three to at most seven iterations.

## 12.2 Discussion

In this part we presented a simple and fast mesh editing algorithm based on a novel differential representation and associated reconstruction potential. The potential was carefully chosen to enforce consistent and non-degenerated frames and yet allow for an efficient optimization. Although our potential is non-linear, we described an efficient reconstruction algorithm that breaks down the high-dimensional non-linear optimization problem into a sequence of two linear systems with constant system matrices and a set of independent low-dimensional



eigenvector computations.

The resulting algorithm is considerably simpler than previous non-linear methods and yields physically plausible deformations. Moreover, it can be parameterized to emulate physical deformation behavior of different materials. In particular, it can be tuned to enforce conformal deformations that locally preserve texture shapes. An additional extra editing mode allows to handle scaling correctly, which is not possible with physically-plausible methods.

The here proposed method also has some limitations that point to directions for future work. First, our formulation is completely surface based. It therefore has no notion of the enclosed volume (if case the surface is closed) and no check is done for volume preservation or self-intersections. Experimental results show that the algorithm tends to avoid gross volume changes and self-intersections for reasonable handle deformations. However, self-intersections can certainly occur for large or drastic handle transformations. Additional residual terms enforcing volume preservation and absence of self-intersections can in principle be added to our potential. However, an efficient optimization of such extended potentials is challenging. A possible solution are force field constraints as will be described in Section 13.2.

Like most editing approaches, our method is currently restricted to handle rotations of less than  $\pi$ . This is due to the representation of rotations by quaternions that do not allow to represent larger rotations. To obtain larger rotations, the deformation has therefore to be split into multiple smaller parts. However, rotations larger than  $\pi$  are rarely necessary in typical editing sessions. Currently, rotations larger than  $2\pi$  are only supported by the method of Lipman et al.[[LCOGL07](#)].

As demonstrated in the previous section, interactive editing of large models is possible with our method. Nevertheless, the efficiency could be improved vastly, if the involved systems of equations were solved completely on graphics hardware in parallel. As described in Section 11.3.5, about 90 percent of the overall computation time is currently spent for sequential backsubstitution. As the triangular factors are usually very sparse with only a few non-zero entries per row, parallelization can only give limited speedup. A more promising alternative are parallel iterative solvers in conjunction with a suitable preconditioning in the pre-computation phase. However, preliminary experiments with incomplete Cholesky factorizations did not show noteworthy speedups.

From the theoretical point of view, a more concise statement about the convergence of the simplified algorithm would be desirable. Another open question is whether and under what conditions the potential  $E$  has a unique global minimum: While some terms of the potential are clearly convex, others are definitely not (most of all the second term, as it is invariant under quaternion flips). Nevertheless, we conjecture that the potential is convex if quaternion constraints are specified.

## 12.3 More recent work

In parallel with our work, an energy for interactive mesh editing was published by Sorkine and Alexa in [SA07] that bears similarities with the here proposed potential. In fact, their energy can be regarded a special case of ours, obtained when setting  $\alpha_q = 0$  and  $\alpha_{qc} = 0$ . In contrast to our coordinates, that explicitly capture the original surface’s curvatures, their approach only enforces rigid transformations of the original one ring geometry  $\bar{c}_v^w$  and thus only implicitly encodes surface curvatures. They also use alternating optimization to decompose the non-linear optimization problem into a linear and a low-dimensional non-linear part and the resulting algorithm is similar to our simplified reconstruction. However, the iteration must be initialized e.g. by the Laplacian editing method [SCL<sup>+</sup>04]. Orientation constraints and parameterizable material as proposed here are not originally supported but it should be possible to adapt their algorithm accordingly.

Also in parallel with our work Xu et al. proposed a mesh editing algorithm in [XZY<sup>+</sup>07] that is based on the very same potential as that of Sorkine and Alexa. Similarly, they also used alternating optimization for an efficient minimization. For initialization, they use the subspace gradient method of Huang et al. [HSL<sup>+</sup>06]. Xu et al. extended the algorithm to sequences of deforming meshes.

Shi et al. [SZT<sup>+</sup>07] proposed an interactive mesh deformations technique that they term “mesh puppetry”. The principle idea of their approach is to combine traditional skeleton-based editing with recent mesh deformation methods. They start with the potential of Huang et al. [HSL<sup>+</sup>06] but add soft constraints, that e.g. ensure consistency with the skeleton. While they also apply alternating optimization, the resulting system matrices after linearization are not constant. To efficiently optimize this potential, Shi et al. use a complex cascading optimization algorithm that exploits the relatively small dimensionality of the mesh skeleton.



---

## EXTENSIONS AND APPLICATIONS

---

In this chapter we present three extensions to the original interactive reconstruction algorithm. The first extension allows to specify an arbitrary changing “force” vector at any vertex during editing. We therefore term this new kind of constraint *force field constraint*. (The term “force” is slightly incorrect since our approach is not physically correct. However, the deformation resulting in presence of force field constraints are again physically plausible. This motivated the name.) We will see that force field constraints are very versatile. In Section 13.3 we show how they can be used to fit a template mesh to a point cloud. The third extension allows to define continuity constraints between two geometrically adjacent but topologically disconnected patches during mesh editing.

These extensions might be interesting in context of mesh editing but were originally motivated by another application, namely the rapid visualization of upholstery for early marketing. Before going into the details of the extensions, we first briefly describe how they can be put to work in this application to give a motivation.

### 13.1 Application: Modeling Upholstery for Early Product Visualization

As a motivating application for the proposed extensions we consider again industrial upholstery production. In Chapter 7 we assumed that the surface shape of the upholstered product is known and computed sewing patterns. This assumption is usually justified for pattern inference where the desired surface has been specified by the designer. In this chapter we take a different view and consider the visualization of upholstery for which only pattern and upholstery frame are known, but not the shape of the upholstered product. In fact, this situation occurs rather often for seats in automotive visualization, e.g. if the shape of the original design prototype is property of the designing company. In addition, for early visualization the



Figure 13.1: A visualization of a seat model retrieved with our method.

design prototype is still subject to frequent changes and might be unavailable by the time required.

In either case, the modeler is left with the task to visualize the upholstered product from preliminary pattern shapes, frame and point measurement data or sketches. The goal of such visualizations is timely product marketing but also to reveal errors and to support decisions during the design phase. To this extent, visualization tools must be fast and flexible so that designers can quickly evaluate different options or modifications.

In this chapter we describe an interactive modeling tool for upholstery, that allows to incorporate partial or incomplete construction data but yet gives the modeler enough freedom for rapid modifications. Starting from an approximate sewing pattern it estimates an initial surface that can be interactively edited. Using the proposed extensions, this initial estimate can be further refined to comply with a given upholstery frame or (partial) point measurements, if such data is available. Figure 13.1 shows the result of an interactive modeling session. A detailed description of the upholstery editing tool can be found in [SDK09].

As lined out in Chapter 7, the actual shape of the upholstered product is determined in the interaction of frame, cushioning and fabric. An alternative to our approach is thus a physical simulation of the deformation to infer the final shape. In fact, the simulation of textiles or cloth has a long tradition in computer graphics.

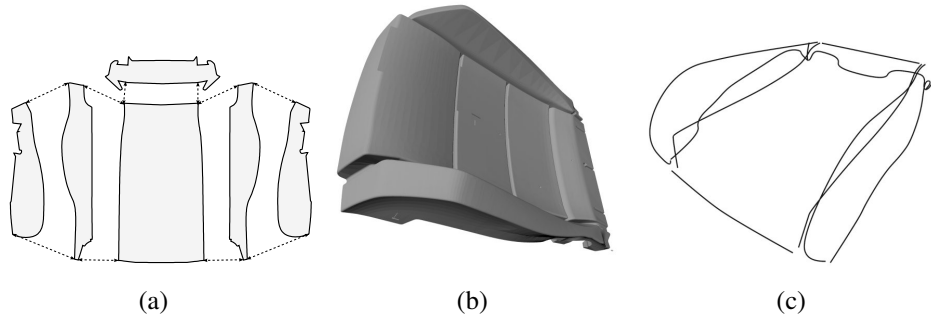


Figure 13.2: (a) sewing pattern with sewing instructions (b) upholstery frame (c) fixations curves on the upholstery frame.

However, for the above described visualization application an accurate simulation is less suited for two reasons: First, a simulation of the interaction is involved and requires precise knowledge of patterns, frame, and material parameters. Second, small design modifications like adding a fancy seam must be actually modelled and implemented in a physically correct manner. This is far more complex than directly editing the surface. For our interactive modeling tool we do not strive for physical accuracy. Nevertheless the results of the reconstruction should be plausible and mimic natural effects like folds or wrinkles.

### 13.1.1 Basic Reconstruction and Editing

We assume that for a piece of upholstered furniture the pattern (or at least an approximation) and (optionally) an upholstery frame are given (see Figure 13.2). The pattern is given as a set of closed planar curves that define the outlines of patches. Individual patches are further annotated with sewing instructions or seams illustrated in Figure 13.2a. For fixed seams, we assume that corresponding fixation curves on the upholstery frame are provided. Figure 13.2c shows an example of a set of fixation curves on a frame. Given this data the problem consists in finding a visibly plausible approximation of the fabric's surface.

For simplicity, we neglect cushioning effects of the upholstery frame as well as continuity at seams for a moment. Our idea to derive a first approximation of the shape is to simulate an elastic sheet cut from the sewing pattern that adheres to the sewing and fixations. Thinking of this sheet as an elastic shell, a first approximation of the fabric's surface is obtained as the configuration of minimal potential energy subject to the constraints defined by sewing and fixation curves. To allow interactive modifications, we resort to the potential proposed in the last chapter.

By using the shell metaphor, the sewing pattern and fixation curves can be regarded as an editable representation for the upholstered shape. In our system, the

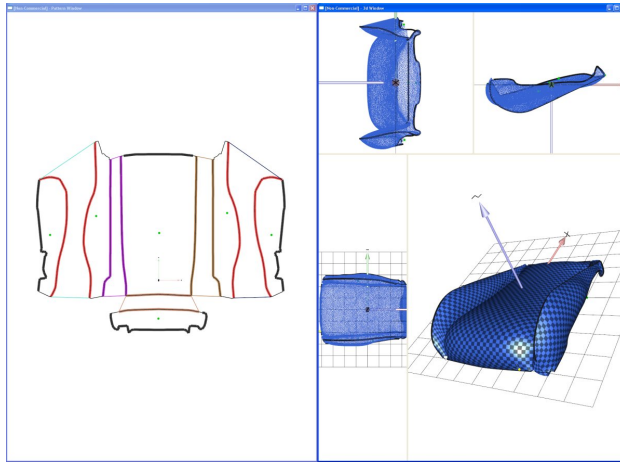


Figure 13.3: The user interface of the interactive upholstery modeling tool.

user can also interactively define additional curves within the pattern. These are automatically mapped onto the reconstructed surface and become fixation curves. Manipulating the control points of these fixations constitutes an intuitive way to make additional interactive changes to the shape of the upholstery.

To reconstruct upholstery with the mesh reconstruction algorithm, we first triangulate the interior of each sewing pattern curve using a fast Delaunay triangulator [She96]. Care is taken, that curve segments associated by seams are triangulated consistently. Associated boundary vertex pairs are stored in a map. A further map keeps track of vertices associated with fixation curves. From the planar triangulations of all patches the quaternion-based coordinates are computed. The mesh reconstruction algorithm will thus minimize bending in the surface as much as possible. In the absence of further constraints the patches will therefore remain flat.

Fixation curves can now be realized in a straight forward manner by adding according vertices to the set of position constrained vertices  $H_x$  and setting their position to the corresponding point on the fixation curve. Sewing constraints between patches, however, require a slight extension of the reconstruction potential: For each pair of vertices  $(v, w)$  associated by a seam, we add the quadratic term

$$\|\mathbf{x}_v - \mathbf{x}_w\|^2$$

to the original constraint energy  $E_{const}$ . In the reconstruction algorithm, this modification results in a few additional constant non-zero entries in the linear system solved in step 3 of the simplified algorithm.

In our implementation the user interface is divided into two windows (see Figure 13.3). The pattern window shows patterns laid-out in the plane with seams

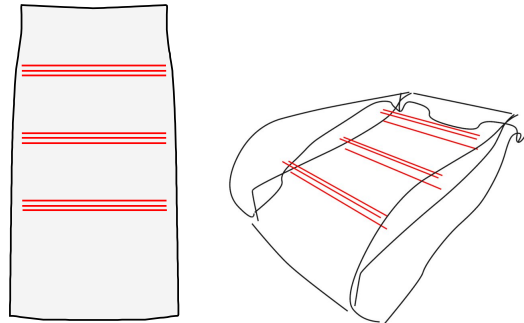


Figure 13.4: Adding fancy seams. Left: additional handle curves in the pattern. Right: corresponding surface handle curves.

indicated by lines. Pattern curves can be modified and rearranged. Moreover, seams between patches can be defined. In the  $3D$  window the reconstructed model and the fixation curves  $C$  are shown. In both windows, the user interacts with the reconstructed model only by adding or changing curves represented as B-splines. As a guidance when adding fixation curves, it is possible to display a polygon model of the upholstery frame. The user interface and its functions are exemplified in the accompanying video.

Modifying a fixation curve in the  $3D$  window only affects the position constraints of the associated vertices. As a result, the shape of the reconstructed surface updates interactively at about 10 – 15 fps. Modifications on a pattern curve in the pattern window results in instant re-triangulation of the patches and re-initialization of the mesh optimizations. Although this is computationally more expensive than changes of fixation curves, patch boundaries can be edited at interactive frame rates for moderate mesh resolutions.

Although results of the basic reconstruction algorithm are plausible, the information encoded by fixation curves are rather sparse and there might be the need to add surface details in some parts of the upholstered object. To provide additional flexibility, we therefore allow the user to draw addition handle curves into the interior of the planar pattern patches after a surface has been reconstructed. In contrast to patch boundaries these handle curves can be open.

Handle curves are triangulated into the respective patches and the corresponding edges in the reconstructed surface are collected into a polygon. We then add an additional fixation seam mapping the handle curve on the corresponding surface polygon. The user can then manipulate both curves interactively. In most cases it is convenient to approximate the surface polygon by a spline for editing.

Figure 13.4 shows an important application of handle curves. Fancy seams are not actually seams between patches, but are of purely decorative nature. Nonethe-

less these seams have a great impact on the object's appearance. Using handle curves, a fancy seam can be easily realized by adding triples of parallel curves as shown in the figure and translating the  $3D$  handle curve in the middle.

## 13.2 Force Field Constraints

The basic reconstruction algorithm relies only on the sewing pattern and fixation curves. The actual shape of the upholstery frame has not been considered so far. However, frame and cushioning certainly have significant influence on the actual shape of upholstered furniture. A physically accurate simulation of the cushioning not only requires an accurate shell model but also a solid body simulation of the frame deformation. Moreover, collision and friction between fabric and frame have to be respected. Such a simulation is clearly not feasible in interactive applications and assumes precise knowledge of material parameters.

As argued in the introduction, physical accuracy is of minor concern in the envisaged application. We therefore propose a far simpler approach based on a force field: From the upholstery frame geometry a force field  $\mathbf{f} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$  is computed, that assign each point on the inside of the frame an outward pointing force vector. For details on the construction of the force field we refer to [SDK09]. For cushioning simulation we first construct an initial surface using the basic reconstruction algorithm. Then a force to each vertex is applied that is equivalent to the value of  $\mathbf{f}$  at its location. In this way vertices inside the cushion are pushed outwards, while no force is applied to vertices on the outside.

To integrate the force field  $\mathbf{f}$  into the mesh optimization without compromising efficiency, we propose a simple extension that seamlessly integrates with the simplified reconstruction algorithm. In analogy to the physical elastic shell model, we interpret the derivative of the reconstruction energy in Equation 11.21 with respect to coordinates  $\mathbf{x}$  as internal restoring forces. In the presence of external forces, at each point of the surface internal and external forces must sum to zero in the equilibrium configuration, i.e.

$$\frac{\partial E}{\partial \mathbf{x}} = -\mathbf{f}(\mathbf{x})$$

If  $E_{forcefield}$  denotes a potential energy with

$$\frac{\partial E_{forcefield}}{\partial \mathbf{x}} = \mathbf{f}(\mathbf{x}) \quad (13.1)$$

an equilibrium configuration is equivalent to a local extremum of the sum of these potentials  $E + E_{forcefield}$ .

For simplicity, we strive for a potential satisfying Equation 13.1 that can be efficiently minimized with minor modifications of the reconstruction algorithm. To this extent, we first assume that forces act only at vertices. If a force  $\mathbf{f}_v = \mathbf{f}(\mathbf{x}_v)$  should be imposed to a vertex  $v$  located at  $\mathbf{x}_v$ , we compute in each iteration a ghost position

$$\mathbf{x}'_v = \mathbf{x}_v + \frac{1}{2}\mathbf{f}_v.$$

We then add a force field energy to the potential that is very similar to the position constraints term of the constraint energy 11.16.

$$E_{forcefield} = \sum_{v \in R} \|\mathbf{x}_v - \mathbf{x}'_v\|^2$$

. In the optimization, we then assume in step 3 of Algorithm 3 that the derivative of the force field  $\partial\mathbf{f}/\partial\mathbf{x}_v$  is small and treat  $\mathbf{f}_v$  as constant. The gradient of the force field energy is then given as

$$\frac{\partial E_{forcefield}}{\partial \mathbf{x}_v} = 2(\mathbf{x}_v - \mathbf{x}'_v) = \mathbf{f}_v$$

, i.e. it satisfies Equation 13.1. With these simplifying assumptions, it is possible to handle force field constraints exactly like positional constraints in the algorithm. The only difference is that ghost positions  $\mathbf{x}'_v$  have to be updated in each iteration. The system matrix of the linear systems remains constant and thus no refactorization is required.

Using the above described force field constraints, the force field  $\mathbf{f}$  can be trivially integrated with the reconstruction by evaluating the field at all vertex positions in each iteration. Concerning the cushioning simulation, it turns out, that results can be improved by applying only the fraction of the force  $\mathbf{f}$  that is orthogonal to the surface. Furthermore, we want the cushioning to pull the surface only outwards, i.e. in the direction of its normal. We thus use the following force assignment

$$\mathbf{f}_v = \max(0, \langle \mathbf{n}_v | \mathbf{f}(\mathbf{x}_v) \rangle) \mathbf{n}_v$$

where  $\mathbf{n}_v$  denotes the vertex normal at  $v$  obtained by averaging adjacent face normals. The effect of the cushioning simulation based on force field constraints is shown in Figure 13.5.

### 13.3 Fitting Template Surfaces to Point Measurements

Using force field constraints proposed in the last section, it is also possible to fit a given surface template to an unstructured point cloud. This is a problem



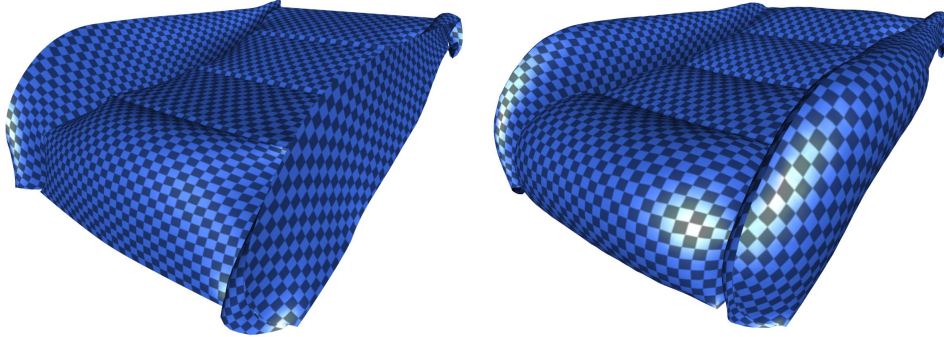


Figure 13.5: Left: result of the basic reconstruction without cushioning simulation. Right: same object with cushioning simulation.

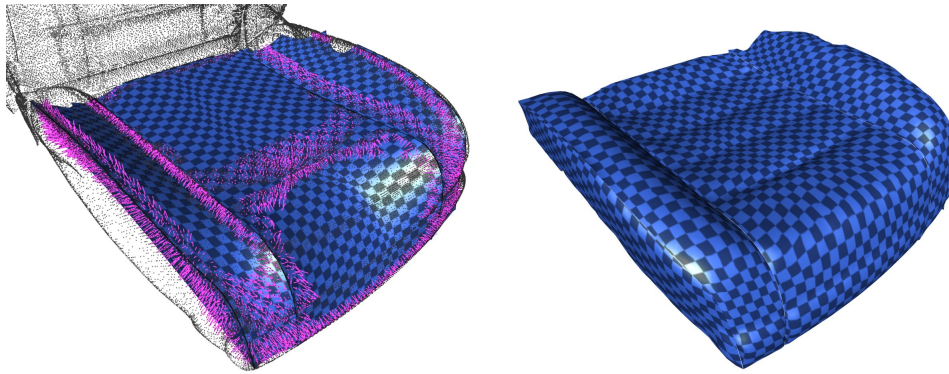


Figure 13.6: Point cloud fitting: Left: initialized mesh with closest pairs. Right: results after fitting.

that arises in numerous applications. Typical examples are consistent meshing of human body scans [ACP03] or registration of time-varying point sampled surfaces [dAST<sup>+</sup>08]. In the context of upholstery reconstruction, this technique will allow us to further refine an initial reconstruction based on sewing pattern and fixation curves by fitting it to 3D point measurements of the actual seat (if available). Such measurements can be obtained e.g. from range scans. However, they are usually incomplete due to scanning restrictions (some parts might not be accessible to the scanner) or show holes. Fitting a template mesh is beneficial in this situation as it fills holes and missing parts by extrapolating the template’s shape into these regions. Moreover, the reconstructed surface can inherit desirable mesh properties from the template such as triangle quality or texture coordinates.

Figure 13.6a shows the output of the basic reconstruction. To fit this initial surface to a sparse point set  $\hat{S} \subset \mathbb{R}^3$  we use an ICP-like approach. The original



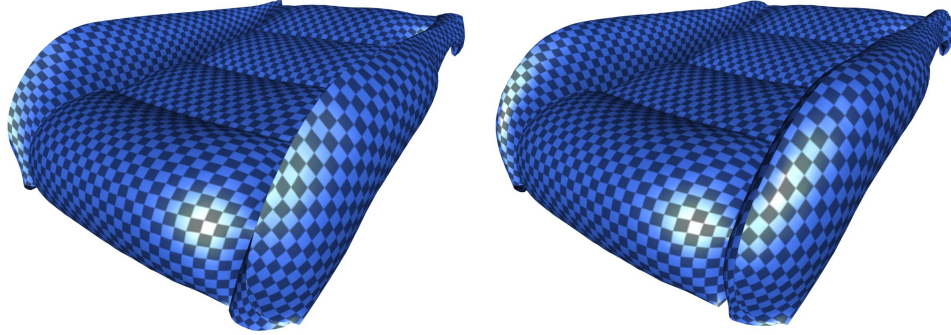


Figure 13.7: Left: results with cushioning simulation but no seam continuity. Right: result after selecting  $G^1$  smoothness for two seams at the top of the side parts.

iterative closest point (ICP) algorithm [BM92] searches a single rigid transformation that aligns two point sets so that their distance is minimized. It iterates between finding pairs of closest points in the two point sets and minimizing the distances between these point pairs until it eventually converges. Generalizations of the ICP algorithm to non-rigid deformations have been used earlier by Marschner et al. [MGR00], Allen et al. [ACP03], Amberg et al. [ARV07] and Stoll et al. [SZR<sup>+</sup>06] which mostly differ in the employed deformation model. We will now show, how the generalized ICP approach can be combined with our non-linear deformation model without compromising its performance.

Similarly to the ICP algorithm, we start by computing for every vertex  $v$  positioned at  $\mathbf{x}_v$  its nearest neighbor  $\mathbf{p}_v \in \hat{S}$  using a fast kd-tree indexing structure. The distance  $\|\mathbf{x}_v - \mathbf{p}_v\|$  is evaluated and all vertices whose values are below a certain threshold are marked. This is illustrated in Figure 13.6 where pairs of marked vertices and closest point samples are connected by red lines.

To minimize the distance between vertices and selected samples  $\mathbf{s}_v$  we use the force field constraint mechanism: To each marked vertex  $v$  we apply the force

$$\mathbf{f}_v = \mathbf{p}_v - \mathbf{x}_v$$

pointing towards the point sample  $\mathbf{p}_v$ . For all unmarked vertices we set  $\mathbf{f}_v = 0$ . Finally, a mesh optimization step is computed. Closest point search and mesh optimization are iterated until the surface converges.

Alternatively, it is possible to use positional constraints to minimize the distance between closest point pairs. However, as the set of vertices associated with some closest point  $\mathbf{p}_v$  can change significantly in the iteration, this would require a refactorization of the matrix in step 3 of the reconstruction algorithm.

Contrary to the original ICP our method does not solve for a single rigid body transformation but a non-rigid deformation of the surface. The resulting mesh approximates the point cloud while still minimizing shape distortion. The results of the fitting algorithm is shown in Figure 13.6b.

## 13.4 Geometric Continuity

So far, inter patch seams constrain only the positions of adjacent patches, i.e. the resulting surface is  $C^0$  continuous at such seams. In some cases, however, higher orders of smoothness are desirable. In our systems, the user can therefore choose  $C^1$  or  $G^1$  smoothness for individual seams between patches.

$C^1$  continuity at a seam can be imposed analogously to  $C^0$  continuity: For each pair of vertices  $(v, w)$  connected by a seam, we add the quadratic term

$$\|\mathbf{q}_v - \mathbf{q}_w\|^2$$

to the original constraint energy  $E_{const}$ . Again, only minor modification of step 2 of the reconstruction algorithm are necessary. These result only in a few additional constant non-zero entries in the matrix.

Let us consider a pair of vertices  $(v, w)$  associated by a seam with  $C^1$  continuity. As a seam connects two otherwise disconnected mesh components, there is originally no edge between these vertices. The above described construction can now be interpreted as inserting a virtual edge  $(v, w)$  with zero length (i.e.  $\bar{\mathbf{c}}_v^w = 0$ ) and identical frame transformation ( $\bar{\mathbf{q}}_v^w = (1, 0, 0, 0)^t$ ).

In upholstery seams are often placed to reduce stretch of the fabric. This is most effective if discontinuities in the texture or fabric pattern are allowed at the seam, i.e. discontinuous tangent vectors at the seam. Nevertheless, the surface shape should be smooth. In other words, the surface should be  $G^1$  continuous at the seam. To allow this weaker form of continuity, we have to enforce identical normal vectors at corresponding vertices along the seam. More precisely, for corresponding vertices  $v$  and  $w$  with frames  $\mathbf{q}_v, \mathbf{q}_w$  and original vertex normals  $\bar{\mathbf{n}}_v, \bar{\mathbf{n}}_w$  we require

$$\mathbf{q}_v(1, \bar{\mathbf{n}}_v)^t \mathbf{q}_v^{-1} = \mathbf{q}_w(1, \bar{\mathbf{n}}_w)^t \mathbf{q}_w^{-1} \quad (13.2)$$

i.e. we ensure that the original normals coincide after the deformation has been applied.

Unfortunately, equation 13.2 is nonlinear in the quaternion components. We therefore resort to an approximation to allow an efficient optimization. Similar to  $C^1$  continuous seams, virtual edges are added with zero edge length. For geometric continuity, however, the frame transformation  $\bar{\mathbf{q}}_v^w$  is not set to identity. Instead, we introduce it as a new auxiliary variable. Consequently we drop the

bar and write  $\hat{\mathbf{q}}_v^w$  instead of  $\bar{\mathbf{q}}_v^w$  to distinguish it from frame transformation of ordinary edges. Nevertheless, in the optimization we treat the virtual edge just like ordinary edges. In particular, the virtual (half)edge  $(v, w)$  will add

$$\|\mathbf{q}_v \hat{\mathbf{q}}_v^w - \mathbf{q}_w\|^2$$

to the first term of the potential 11.21. We chose  $\hat{\mathbf{q}}_v^w$  in such a way that the condition 13.2 is satisfied, when  $\mathbf{q}_w$  is replaced by  $\mathbf{q}_v \hat{\mathbf{q}}_v^w$ . In general, there is more than one valid choice. We use quaternion slerps [Sho85] to find the minimal rotation that aligns the frame normals while minimally perturbing tangential vectors.

Although this choice makes  $\hat{\mathbf{q}}_v^w$  a function of  $\mathbf{q}_v$  and  $\mathbf{q}_w$ , we can again use alternating optimizing to minimize the potential with respect to  $\hat{\mathbf{q}}_v^w$  for all edges separately from the remaining unknowns. In fact, the optimal slerps can be determined for all virtual edges in parallel after step 2b of the simplified reconstruction algorithm 3. By treating virtual edges just like ordinary edges, no further changes to the algorithm are required. Figure 13.7 shows a results of the reconstruction with  $G^1$  continuous seams.



## **Part IV**

# **Visibility Driven Deformation for Panorama Maps**



# CHAPTER 14

---

## INTRODUCTION

---

Besides their various and wide spread uses in modeling and geometry processing, deformation of solids and surfaces have also put to work in visualization for illustrative purposes. In fact, deformation techniques are a well-known and frequently used technique in the toolbox of visualization. Introduced to visualization by Barr [Bar84a], space deformation have subsequently been applied to various visualization problems. Examples include survey images showing objects from multiple sides in a single continuous view [Sin02], visualization of relativistic effects or charged particle movements [Grö95], cartograms [GN04] or illustrative volume visualization [CSC07].

In this part of the thesis we concentrate on a further visualization problem: the automatic generation of panorama maps. Panorama maps sometimes also called panoramic, pictorial or illustrated maps are a fascinating blend between traditional landscape painting and geographic map (see Figure 14.1). Like landscape paintings they contain depictions of details shown from a familiar, earth bound perspective. At the same time, prominent geographic features are visible and not occluded. Panorama's therefore convey spatial survey knowledge much like geographic maps.

We will see, that in the generation of panorama maps, deformation techniques based on suitable potentials can be put to work: Starting from the original geometry of the terrain as well as a user specified viewpoint, we apply a carefully chosen deformation that ensures visibility of all important features from the given viewpoint. The deformation itself is again found by minimizing an appropriate potential. In contrast to potentials considered so far in this thesis, this potential not only measures shape deformation but also visibility of features. We therefore call the resulting deformation *visibility driven deformation*.

Moreover, for panoramic map design, we will also revisit the way, shape deformation is quantified. While the deformation potential developed in Part III is in principle applicable, for panoramic map design it is beneficial to use potentials





Figure 14.1: A panoramic map painted by Heinrich Berann in 1962 showing the Yellowstone National Park, Wyoming.



that given the designer or cartographer explicit access to differential properties of the surface such as mean curvature, angles or surface area while an interactive optimization is of minor importance.

After giving a short introduction to the art of panorama drawing and reviewing related work in visualization in the remainder of this chapter, Chapter 15 will present suitable potentials for visibility and shape preservation for panoramic maps design. An automatic design method for panoramas based on these potentials is then described in Chapter 16. That chapter will also give experimental results and a discussion.

## 14.1 Panorama Maps

Traditionally, panorama maps are manually and artfully painted to ensure that all important features are well visible at the same time. As this is in general not possible with a correct perspective projection, the artist has to rearrange the landscape or, equivalently, modify the projection, e.g. by seamlessly blending between several perspectives as demonstrated in Figure 14.2. However, at the same time, the artist takes care that the depicted landscape still remains recognizable to the beholder. Heinrich Berann, an Austrian cartographer who is deemed the father of the modern panorama map archived a particular high perfection in his art: In his panoramas blending of perspectives or modifications of the original landscape, which are in some cases quite drastic, are hardly noticeable even under close inspection.

Panorama maps combine the advantages of two types of depictions: Like geographic maps, they provide a good overview and avoid occlusion in regions of interest. For example in ski maps, such regions of interest are ski slopes, lifts or restaurants which are drawn at a multitude of their actual scale. If a feature is occluded, the artist carefully rearranges terrain to disclose it. At the same time they inherit the familiar perspective of landscape images drawn from earth bound view point. In ski maps, the observer usually assumes the perspective of a person within the resort. Because of this property panoramas are thought to be less abstract and easier to visualize than ordinary maps [Pat00] and the visitor can easily orient a panorama by matching prominent landmarks as for example mountain vistas. This assumption is also supported by recent results in the field of human computer interaction: In a user study conducted by Chittaro and Burigat [CB05] comparing traditional 2D map depiction to maps augmented by photographs taken from a pedestrian perspective it was found that the latter representation clearly helps people to orient themselves.

Due to these properties, panoramas have many popular applications ranging



(a)



(b)



(c)



(d)



(e)

Figure 14.2: a) A panorama of the Zugspitze ski resort drawn by H. Berann. (under copyright by Panorama Studio Vielkind) b) A perspective image of the same region. While the foreground matches quite well, differences become apparent in the background (see closeups in c and d). e) the panorama perspective for the enlarged region rather matches a perspective image taken from a far higher view point.

from city and topographic visualizations to navigation aids in national parks and ski resorts, where they are handed out to visitors. An impression of the versatility of panoramic maps can be obtained by considering the extensive work of Heinrich Berann [[wwwa](#)] who alone has drawn over 500 panoramas or the archives of the popular panorama studio Vielkind [[wwwc](#)].

## 14.2 Contributions

As lined out in the previous section, the design of panoramic maps is up to today still a manual process. Combining the often contradictory view points into a single image requires not only great skill and art, but also tedious and painstaking work. In this process the designer must move, enlarge and rearrange features to ensure their visibility while keeping the landscape recognizable.

This part of the thesis proposes an automatized design method for panoramic maps. It starts from conventional elevation data and aerial images and computes images from arbitrary viewpoints that maximize the visibility of a user specified set of features while minimizing deformation of shapes. Compared to traditional panorama design, our method greatly simplifies creation of panorama maps: The user only provides an initial view point as well as regions of interest. Occlusions of features are then resolved automatically and regions of interest are scaled to enhance visibility (see Figure 14.3).

At the heart of the method is a set of novel potentials measuring visibility and occlusion of features. Both potentials are smooth by design and can be minimized by standard optimization methods. While evaluation of the visibility potential is linear in surface complexity, the occlusion potential is quadratic as in theory an occlusion relation can persist between any two points on the surface. To speed up evaluation of occlusions and to make optimization feasible, we propose a custom tailored acceleration scheme.

To control shape deformation, we also generalize the metric deformation potential from Part II to maps between two surfaces. This potential is complemented with an established curvature potential to quantify shape deformation. While the proposed method is automatic, the user can tune the influence of individual potentials to influence the final result. This tuning is very intuitive as all potentials are directly related to geometric quantities as angle, lengths or curvature.

Besides applications as touristic or ski maps, we also see potential application for our algorithm in mobile navigation systems: Since our method is fully automatic it is possible to provide drivers or hikers with panoramic maps customized to their current location, route, and personal preference, showing features that would be occluded in an ordinary perspective image. Last but not least, together

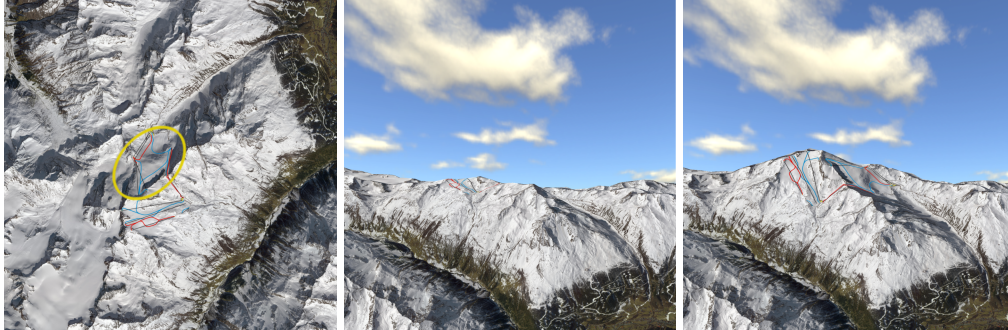


Figure 14.3: A ski map of the glacier area in the Sölden ski resort (Austria) computed with our method (slopes in blue and red, lifts in black). On the left an aerial image and an ordinary rendering are shown for comparison. Please note, that slopes on the Rettenbach glacier (marked in yellow) are completely occluded on the undeformed surface shown in the middle.

with the growing amount and higher availability of digital terrain and city data we recognize an increasing request for consumer cartographic tools. Even today, most GPS systems come with a desktop software, that allows to visualize e.g. bike or ski trips. With the here proposed automatic design method the user will be able to create a personalized panorama based on his own data with a few mouse clicks.

Apart from the projection there are many more aspects that make a panoramic map different from an ordinary photo as e.g. draw style, lighting or labeling. In this work, we concentrate on panoramic projections and leave other aspect to future research. Our method therefore results in maps that are photo realistic except for the projection.

### 14.3 Previous Work

As an alternative to the standard perspective pinhole camera numerous non-standard projection models have been proposed in the past. Although many of these are in principle suitable to model the effects observed in panoramas, their original intention was to give an artist a maximum of control over the projection. The problem of proper feature arrangement and minimal occlusion is still left to the designer. In contrast, our goal lies in a panorama design that automates these tasks. In the following we will briefly discuss the most important approaches to non-standard perspectives:

For artistic and overview renderings composition or blending of multiple views was proposed by Agrawala et al. [AZM00] and Singh [Sim02]. In the latter approach the matrices of several exploratory cameras are combined into a single



camera transformation, which can also be regarded as a deformation of the surrounding space. A general model for such space deformation was introduced by [Bar84b] and [SP86] and later applied in the context of non-linear cameras by [SGS05] and [CBGS05]. This approach was also recently pursued by Brosz et al. in [BSSS07] where a space deformation is defined via a set of control surfaces. While space deformations can be evaluated interactively [BNG06], defining a deformation that maximizes feature visibility requires tedious manual work.

Other generalizations of the standard pinhole camera model include projections onto non-planar surfaces as e.g. proposed by Löffelmann [LG96], Levene [Lev98] or Inakage [Ina91]. Yonggao et al. [YCB05] extended this approach with 3D lenses that are manually placed in the scene. The generalized linear cameras model by Yu and McMillan [YM04] unifies many of the above mentioned camera models by allowing for arbitrary projection surfaces and directions. Even more general is the concept of non-linear ray tracing as introduced by Gröller [Grö95] where lines of sight are traced through a predefined force field by numeric integration. In the same spirit the occlusion camera model [MPS05] bends rays of sight at silhouettes to reduce disocclusion errors in image based rendering. Again, these approaches can in principle be applied to produce a panorama but they demand significant user intervention. The modification of the projection plane or the specification of force field that maximizes visibility is clearly counter intuitive and requires lengthy and painstaking manual adjustments. The same is certainly true for 3D lenses: The choice, placement and orientation of a system of lenses that optimizes feature visibility and occlusion is clearly not trivial and also requires lengthy manual intervention.

Apart from these general non-standard camera models, only a few approaches address the problem of computer assisted panoramic map design: In [Pre02] Premoze proposes a interactive height field editor to perform vertical scaling and rotations in the parameter plane. A similar approach is taken by Takahasi et al. [TON<sup>+</sup>02] who suggest an automatic segmentation algorithm. The user can then manually rotate or move the extracted segments. The manipulated segments are subsequently merged using a mesh smoothing algorithm. In [FSWE07] Falk et al. use non-linear ray tracing to render panoramic maps. They define a force field that bends rays towards the terrain to reduce the overall degree of occlusion. However, the algorithm does not account for features and the user must manually edit the 3D force field to rearrange or disclose occluded features. Recently and parallel to our work, Grabler et al. [GASP08] developed an approach to generate city tourist maps, that automatically infers buildings and features of interest using web queries. Buildings are individually deformed and rearranged on a street map. The underlying terrain is assumed to be planar and is not altered.

The art of selecting the relevant information and its appropriate representation in a map drawn for a certain purpose and at a certain scale is referred to

by cartographers as “generalization” [Mac95, HGM02]. The generalization techniques used in the manual design are numerous and up to today no comprehensive classification exists. Nevertheless, efforts have been made to automatize certain well-established techniques that adapt level of detail or improve readability of automatically generated maps. E.g. in [Ses00], Sester proposes a least-squares approach to simplification of polygonal  $2D$  building outlines. Simplification is one of the most common generalization techniques, which reduces visual clutter by omitting unnecessary details. The approach is also extended to displacement, another well-known generalization technique, that rearranges map entities to widen important streets so that they appear more clearly and well visible. Agrawala and Stolte develop in [AS01] a set of generalization techniques specifically targeted to  $2D$  route map design. Their approach simplifies routes to sets of straight line segments that intersect at turning points. Furthermore, long route segments are scaled in length to provide a better overview. Closer related to our work is the generalization technique described by Takahasi et al. [TYSN06] as it operates on a  $3D$  panorama. In their approach the height of hills and mountains is adjusted to reduce occlusion of upcoming parts of the route in car navigation systems. In contrast to their method, the approach presented here allows for more general surface deformations than vertical terrain scaling.

---

## POTENTIALS FOR PANORAMA MAPS

---

### 15.1 Variational Approach

#### 15.1.1 Why Surface Deformation ? — Techniques in Manual Design

Comparing a panoramic map to an ordinary photo, the most striking difference is definitely the non-standard projection: Depending on his intention the artist changes size, orientation and visibility of certain terrain features at will. The techniques of Heinrich Berann concerning the projection have been studied by the American cartographer Tom Patterson, who identified three principle techniques [Pat00]:

**Airplane Perspective** Starting from a flat, earth bound perspective, Berann tilted terrain in the foreground towards the observer so that the surface normal points towards the observer (see Figure 15.1). In this way objects in the front are drawn from a steep angle almost like in an ordinary map, while far away objects are still shown from a flat angle and appear at the horizon which results in an airplane-like perspective. The map like perspective in the foreground usually already resolves some occlusions.

**Vertical exaggeration** A vertical exaggeration and resizing is applied selectively to features as e.g. certain mountain peaks to make them appear larger or more impressive.

**Rearrangement** While airplane perspective and exaggeration can resolve some occlusion issues, they are by far not sufficient to ensure the visibility of all features. Berann therefore rearranged whole parts of the terrain e.g. moving or rotating geographic features as like mountain peaks or even whole mountain ranges.

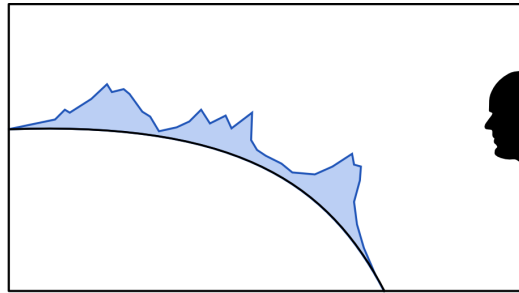


Figure 15.1: An airplane-like perspective can be obtained by tilting the terrain in the foreground towards the observer.

To model non-standard projection several concepts have been proposed in the past, ranging from non-linear ray tracing and special camera models to space and surface deformation. While in theory any of the above techniques is flexible enough to render images in the style of Berann we decided to base our approach on surface deformation as the above described techniques suggest that in manual design the cartographer mentally changes the terrain itself rather than changing his perspective. While all of the techniques map easily to deformations, it is less obvious how to reproduce their effects using other projection models. For example, a geographic rearrangement can be easily described as surface deformation. In contrast, the specification of an appropriate force field to guide rays of sight in non-linear ray tracing or the design of a  $3D$  lens system is not straightforward. We therefore think that surface deformation correspond most closely to the way panoramas are created and perceived. This conceptual proximity of surface deformation and manual mapping techniques will facilitate the formulation of appropriate deformation measures in the next section.

### 15.1.2 Problem Statement

Beside the depicted landscape the most discriminating aspects of a panoramic map are the observer's position and orientation. Usually these parameters are carefully chosen by the artist to achieve a particular effect. Ski maps, for example, are usually drawn from the valley looking into the opposite direction of the slopes, while for a tourist maps we might want to have some important or noteworthy building in the center. Furthermore, panoramic maps uncover and emphasize certain features in the depicted area. In ski maps, for example, slopes and lifts appear larger and clearly visible to facilitate navigation.

As the choice of the observer's position as well as the set of features is clearly subjective, we leave their specification to the user and assume that both are given



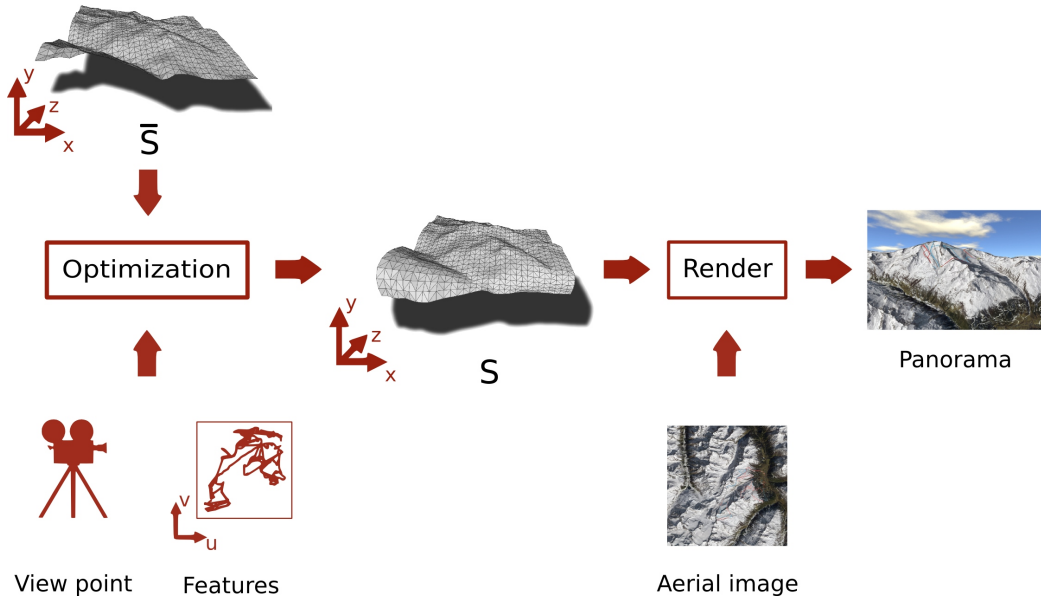


Figure 15.2: Overview of our approach: Starting from an initial terrain geometry  $\bar{S}$  and user specified features and viewpoint, our method computes a deformed surface geometry  $S$ . When rendering this deformed geometry from the specified viewpoint features become visible. For the final panorama we use a high resolution aerial image as texture.

as inputs to our algorithm. To assist the user during the specification process and to avoid a tiring selection of features, we offer the possibility to create an appropriate feature set from a query to a geographic database that stores semantic entities like roads, buildings, ski slopes, or lifts.

The problem that we tackle in this chapter is illustrated in Figure 15.2 and can be informally put as follows: Starting with some terrain surface  $\bar{S}$  and given a point of view, line of sight, as well as a set of features on the surface, we search for a deformed surface  $S$  so that all features are unobstructed and well visible, while  $S$  still resembles  $\bar{S}$ . To generate the final panorama, we render the deformed surface  $S$  using a high resolution aerial image as texture from the specified viewpoint.

### 15.1.3 Variational Formulation

We start with a formalization of the above informal problem statement: In the following we make our usual assumption that the undeformed surface  $\bar{S}$  is parameterized by a map  $\bar{x}$  over a parameter domain  $\omega \subset \mathbb{R}^2$ . Since terrain data is usually

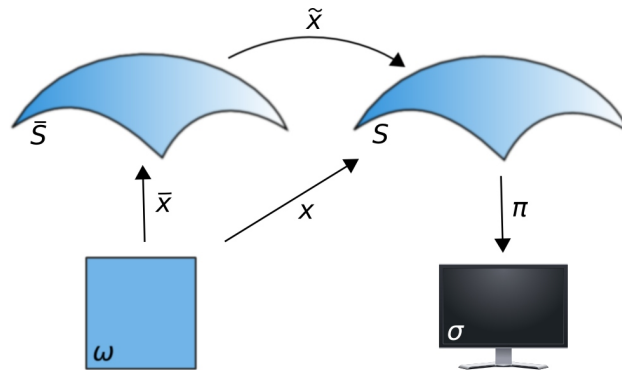


Figure 15.3: The notation used in the following: The original surface  $\bar{S}$  is deformed by a map  $\tilde{x}$  into a surface  $S$  that is mapped onto the screen  $\sigma$  using the projection  $\pi$ . Both original and deformed surface are moreover parameterized over a common planar domain  $\omega$ .

given as a 2.5D height field, the domain  $\omega$  and a corresponding parameterization  $\bar{x}$  can be obtained e.g. by simple projection. (If the surface is textured the texture map is another possible choice for  $\bar{x}$ .)

We further assume that features are specified as a map  $f : \bar{S} \rightarrow [0, 1]$  that assigns each point on the undeformed surface a feature value, where a value of one corresponds to most important feature points and zero to non-feature points. Feature values in between these two extremes correspond to the relative importance of the feature. The given observer parameters define a standard camera projection  $\pi$  that maps world coordinates onto the screen  $\sigma$ . In this chapter, we will use an orthographic camera for the sake of simplicity but a generalization to perspective projections is not difficult. Figure 15.3 gives an overview of our notation. As shown in the figure, the deformed surface  $S$  is again defined via a surface deformation  $\tilde{x}$ . By simple concatenation of  $\bar{x}$  and  $\tilde{x}$  we further obtain a parameterization  $x$  for  $S$  that will be handy in the following.

With the above, the problem of panoramic map design is reduced to the choice of a suitable deformation  $\tilde{x}$ . To find such a deformation we take a variational approach: In the next section we first define appropriate measures that formalize the notions of “resemblance” and “visibility” in the informal problem statement given in the last section. The deformation  $\tilde{x}$  can then be determined as extremal function of these measures.

Panoramic maps as drawn by Heinrich Berann are doubtlessly pieces of art. As such their quality is purely subjective and impossible to quantify. However, as with ordinary maps, panoramas are also designed to provide information on the

existence and location of certain features. In contrast to ordinary maps, that are naturally free of occlusion, silhouettes and occlusions are – just like in landscape images — desirable stylistic elements in panoramic maps. However, while occlusion is desired to some extent, features should obviously be not occluded and well visible. One reasonable measure of a panorama’s quality is therefore the visibility of important features.

A second quality of panoramas is shape preservation: Although not drawn to scale, panoramic maps do not exhibit more terrain deformation than necessary. When looking at Berann’s panoramas shape deformation is hardly noticeable. This is due to the fact, that shapes appear locally intact everywhere. Only when comparing local shapes across the panorama relative differences in size, position and orientation are noticeable. By preservation of local shape the map designer ensures that the deformed terrain remains recognizable to the user. A second measure of panorama quality is therefore the extent to which local shapes are preserved. In the following sections we will derive formal measures for shape preservation and for feature visibility.

## 15.2 Deformation Potential

As argued above, in panorama maps it is primarily local shape that is to be preserved. According to the elastic shell model (see Chapter 3.2.1) on surfaces there are two types of local shape deformation: membrane strain and bending strain. For geographic maps in general and panorama maps in particular, small membrane strain, i.e. tangential shape deformation is of special importance as it directly influence the extent to which areas, angles or length are distorted in the map and thus the usability of the map. While bending strain is no issue in tradition geographic maps, it can deteriorate the recognizability of panoramas as demonstrated in Figure 15.4.

To measure both types of deformation two different potentials have been discussed so far in this thesis: The physical elastic shell model (Chapter 3.1) and the potential for reconstruction from quaternion-based coordinates (Chapter 11.2.2) account for both membrane and bending strain. They can thus be directly applied to the setting of panorama maps. However, there are disadvantages: The elastic shell potential, while physically correct, requires the specification of an elasticity tensor and it is not obvious, how to choose these parameters in context of panorama maps: For a cartographer, it makes no sense to create a map with e.g. rubber or textile like deformation behaviour. He rather likes to know and control how much geometric quantities like length, area or angles are distorted. The elastic shell model does not give direct access to these properties. The same

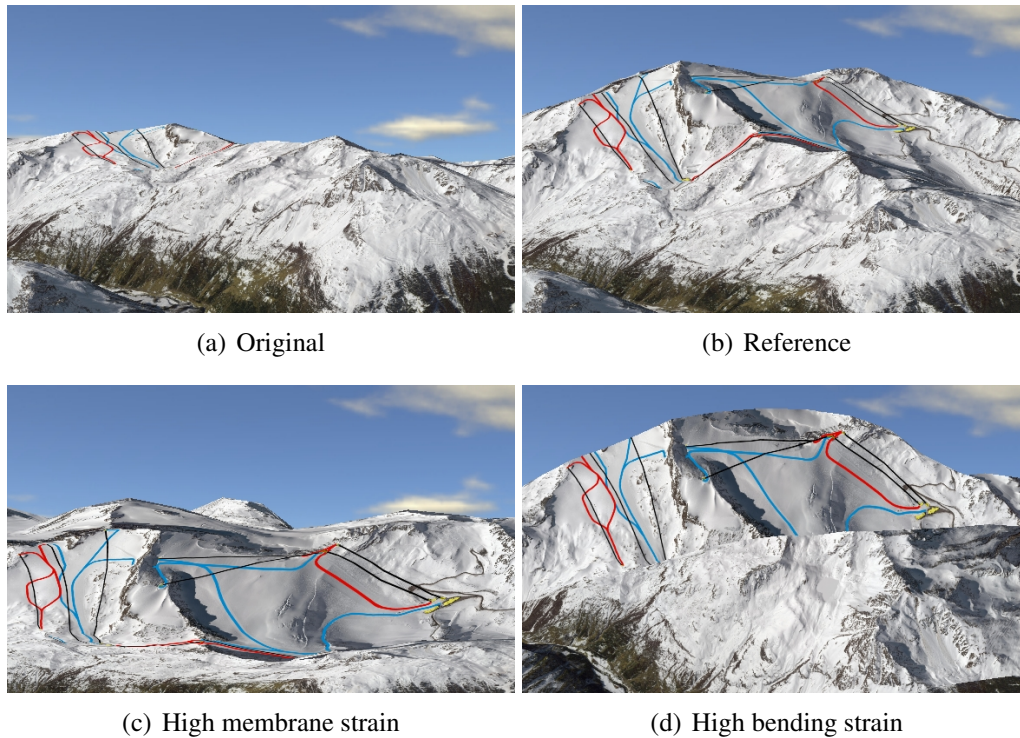


Figure 15.4: The effect of membrane and bending strain in a panorama: a) the original view from the user specified view point. Many features (slopes and lifts) lie in a occluded valley and are invisible b) a reference panorama computed with our method. Note, how slopes in the neighbouring valley become visible. c) The result of our method if more membrane strain is allowed in favor of better slope visibility. While features increase in size, they appear highly distorted so that their are hard to recognize. d) A result with bending strain increased in favor for better feature visibility. In particular silhouettes become unrecognizable. (Note: For this result our method did not succeed in resolving all occlusions. See discussion in Section 16.5.2)

holds, albeit in an alleviated sense, for the quaternion-based reconstruction potential which is designed to give access to frame orientations and vertex positions. While for panorama design access to these properties would enable easy manual intervention, our emphasis lies on a mostly automatized design, where the user interacts at most by allowing or penalizing deformation of geographic structures. At the same time, the interactivity of the quaternion-based potential is of minor importance in context of panorama design.

For these reasons we devise a new shape potential for panoramic maps that gives direct access to angle, length, area and curvature deformation. It is composed of two sub potentials one of which is based on the surface parameterization potential from Part II. The choice of both sub potentials is discussed in the following sections.

The shape potential derived in the following can be regarded as a special case of the shape matching energy proposed by Litke et al. [LDRS05]. But in contrast to Litke et al. who optimize the potential to find a matching between the planar parameter domains of two fixed surfaces, we vary the potential with respect to the deformed surface  $S$ . Moreover, our approach bears great similarity with the recently proposed constrained-based shape deformation approach of Eigensatz and Pauly [EP09, ESP08]. They present a variational framework that, similar to our method, gives the user direct access to curvature and metric properties for shape editing. While in our application the deformation is driven by feature visibility, their user interface allows to specify those properties manually either point wise or as quantities integrated along curves or over surfaces. Their metric energy can be regarded as a special case of the membrane potential described in Section 15.2.1.

### 15.2.1 Membrane Strain Potential

Just as with ordinary maps, deformation of geographic entities depicted in the panorama map should be as small as possible to give the observer a realistic impression of the shown area. Put aside the unavoidable deformation caused by the perspective, shape deformation is to a large extent due to deformation of the texture, i.e. membrane strain. It is therefore very important to control tangential shape deformations in addition to curvature changes. If for example two roads on the surface meet at a right angle in the original surface, we would like to preserve that angle to keep the crossroad easily recognizable. Moreover, other geometric quantities as lengths and areas of shapes in the texture should also be preserved whenever possible.

A potential that captures distortion of such geometric quantities has been derived in Section 5.4 for surface parameterizations. However, intended to measure deformations between a surface and a planar texture domain, that potential is conveniently expressed in terms of the metric tensor  $\tilde{g}$ . In order to apply it to our

setting, we need to generalize the metric tensor  $\tilde{\mathbf{g}}$  to maps between two curved surfaces  $\bar{S}$  and  $S$ . We can then substitute this generalization in the definition of the potential from Section 5.4 to obtain a suitable membrane potential.

To find this generalization we first turn back to the case of a parameterization  $\mathbf{x}$  over a planar domain  $\omega \subset \mathbb{R}^2$ . From its metric tensor  $\mathbf{g}$ , we define in each point  $\mathbf{u} \in \omega$  an inner product on the tangent space  $T_{\mathbf{u}}\omega$  (which is isomorphic to  $\omega$ ) in the following way: For points  $\mathbf{v}, \mathbf{w} \in T_{\mathbf{u}}\omega$  we set

$$\langle \mathbf{v}, \mathbf{w} \rangle_{\mathbf{x}} := \langle \nabla \mathbf{x} \mathbf{v}, \nabla \mathbf{x} \mathbf{w} \rangle = x_{i,\alpha} v_{\alpha} x_{i,\beta} w_{\beta} = v_{\alpha} g_{\alpha\beta} w_{\beta} \quad (15.1)$$

The last expression appeared already in Equation 2.1 from Section 2.1.1 where it was used to compute length, angles and areas of shapes on the surface  $S$  via the parameterization  $\mathbf{x}$ . Measuring length, angle and area using the inner product  $\langle \cdot, \cdot \rangle_{\mathbf{x}}$  is thus equivalent to measuring these properties after mapping the shapes onto the surface  $S$ . The metric tensor can now also be characterized as the unique positive definite symmetric matrix  $\mathbf{g}$  associated with the inner product  $\langle \cdot, \cdot \rangle_{\mathbf{x}}$  that satisfies Equation 15.1 for any  $\mathbf{v}, \mathbf{w} \in T_{\mathbf{u}}\omega$ .

Let us now come back to the more general case of a mapping  $\tilde{\mathbf{x}}$  that maps between two curved surfaces  $\bar{S}, S \subset \mathbb{R}^3$ : Analogous to the planar case, we define in each point  $\bar{\mathbf{p}} \in \bar{S}$  an inner product on  $T_{\bar{\mathbf{p}}}\bar{S}$  as

$$\langle \mathbf{v}, \mathbf{w} \rangle_{\tilde{\mathbf{x}}} := \langle D_{\bar{\mathbf{p}}}\tilde{\mathbf{x}}(\mathbf{v}), D_{\bar{\mathbf{p}}}\tilde{\mathbf{x}}(\mathbf{w}) \rangle \quad (15.2)$$

where  $D_{\bar{\mathbf{p}}}\tilde{\mathbf{x}}$  is the derivative of  $\tilde{\mathbf{x}}$  introduced in Section 2.1.3. As in the definition of the metric tensor,  $\langle \cdot, \cdot \rangle$  on the right hand side in the above equation denotes the Euclidean dot product in  $\mathbb{R}^3$ . The considerations for curves made in Section 2.1.1 easily carry over to the case of a map between surfaces. Length, area, and angles of shapes on the surface  $\bar{S}$  can be measured using the metric  $\langle \mathbf{v}, \mathbf{w} \rangle_{\tilde{\mathbf{x}}}$  and, similar to the planar case, these measurements are equivalent to the corresponding measures on the image of the shape under  $\tilde{\mathbf{x}}$ . Therefore, the inner product  $\langle \mathbf{v}, \mathbf{w} \rangle_{\tilde{\mathbf{x}}}$  is said to be a pull back of the Euclidean dot product on  $S$  via the map  $\tilde{\mathbf{x}}$ .

Now, a natural generalization of the metric tensor can be defined as the positive definite symmetric matrix associated with the metric tensor  $\langle \mathbf{v}, \mathbf{w} \rangle_{\tilde{\mathbf{x}}}$  analogous to Equation 15.1. However, this definition is dependent on the choice of the coordinate frame for the tangent plane at each point  $\bar{\mathbf{p}} \in \bar{S}$ .

Fortunately, the potentials from Part II are functions of the eigenvalues of the metric tensor only and thus is invariant with respect to similarity transformations of the metric tensor. As an orthonormal change of coordinates in the tangent plane corresponds to a similarity transformation of the matrix associated with  $\langle \mathbf{v}, \mathbf{w} \rangle_{\tilde{\mathbf{x}}}$  it is sufficient for our needs to choose an arbitrary orthonormal coordinate frame for the tangent space. Using the fact that  $D\tilde{\mathbf{x}} = D\mathbf{x} \circ D\bar{\mathbf{x}}^{-1}$  it can be shown that independent of the chosen frame the resulting symmetric matrix is similar to

$$\mathbf{G}_{\tilde{\mathbf{x}}} := \mathbf{g}\bar{\mathbf{g}}^{-1} \quad (15.3)$$



From this definition it is clear that  $\mathbf{G}_{\tilde{\mathbf{x}}}$  is a natural generalization of the metric tensor. Whereas the metric tensor describes how lengths, angles and areas of shapes are affected by a parametrization, i.e. a mapping from a flat domain onto a surface in three-space, the matrix  $\mathbf{G}_{\tilde{\mathbf{x}}}$  similarly characterizes deformations of shapes on the surface  $\bar{S}$  when mapped by  $\tilde{\mathbf{x}}$ . The potential developed for parametrizations in Section 5.4 can now be applied to the surface to surface map  $\tilde{\mathbf{x}}$  by simply replacing  $\tilde{\mathbf{g}}$  by  $\mathbf{G}_{\tilde{\mathbf{x}}}$ .

As an alternative to the potential from Section 5.4 it is also possible to use several other energies originally developed for surface parameterization for our purpose as long as these are formulated in terms of the metric tensor. In fact, in our implementation we use the slightly more general energy of Clarenz et al. [CLR04b] as it allows to control deformation in length in addition to angle and area. Citing from [CLR04b], the resulting measure of tangential shape deformation for a map  $\tilde{\mathbf{x}}$  is then given as:

$$E_{\text{tangential}}[\tilde{\mathbf{x}}] = \int_{\bar{S}} \alpha_l t + \alpha_a d + \frac{\alpha_l + \alpha_a}{d} + \alpha_c \left( \frac{t^2}{d} - 4 \right) d\bar{S} \quad (15.4)$$

where the real valued constants  $\alpha_l, \alpha_a, \alpha_c$  control the relative importance of length, area and angle preservation respectively. The functions  $t = \text{tr}(\mathbf{G}_{\tilde{\mathbf{x}}})$  and  $d = \det(\mathbf{G}_{\tilde{\mathbf{x}}})$  denote the trace and determinant of the substituted matrix  $\mathbf{G}_{\tilde{\mathbf{x}}}$ .

### 15.2.2 Bending Potential

The membrane potential derived in the previous section captures differences in the metric tensor, i.e. the first fundamental form. The bending potential should complement the membrane potential in that it measures differences in the second fundamental form, i.e. in the shape operators of surfaces  $\bar{S}$  and  $S$ . While in theory, both principal curvatures and associated directions are necessary to completely describe the shape operator, we adopt the bending energy proposed by Grinspun et al. [GHDS03] that measures only differences in mean curvature which is sufficient for our needs. For a given deformation  $\tilde{\mathbf{x}}$  the deformation of curvatures is given as:

$$E_{\text{bending}}[\tilde{\mathbf{x}}] = \int_{\bar{S}} (\bar{\kappa}_{\text{mean}} - \kappa_{\text{mean}}(\tilde{\mathbf{x}}))^2 d\bar{S} \quad (15.5)$$

where  $\bar{\kappa}_{\text{mean}}, \kappa_{\text{mean}}$  denote the mean curvature of  $\bar{S}$  and  $S$ , respectively.

## 15.3 Visibility Potentials

The visibility of surface features in a panorama can be impaired by two different factors (see Figure 15.5): First, the feature might appear too small on the screen

to be clearly visible, i.e. their projected screen size is too small or they might even lie on a back facing part of the surface. But even if the projected screen size is large enough, a feature might still be not visible because it is occluded by other parts of the terrain. Corresponding to these two sources of impaired visibility, we will derive two potentials measuring visibility: A screen size potential and an occlusion potential. Both will be discussed in the upcoming sections.

### 15.3.1 Screen Size Potential

To quantify the screen size of a feature we consider the concatenation of surface deformation  $\tilde{\mathbf{x}}$  and camera projection  $\pi$  which maps from the original surface  $\bar{S}$  to the screen  $\sigma$  (see Figure 15.3). Again, we can put the pull back metric to work to find the increase in screen size caused by the deformation  $\tilde{\mathbf{x}}$ . From the pull back metric  $\mathbf{G}_{\pi \circ \tilde{\mathbf{x}}}$  of the combined map we can derive changes in the length, angle and area of infinitesimal shapes on the original surface  $\bar{S}$  that are mapped by  $\pi \circ \tilde{\mathbf{x}}$  onto the screen. In particular, applying the generalized integral substitution rule we obtain the change in area as the area element

$$d\sigma = \sqrt{\det(G_{\pi \circ \tilde{\mathbf{x}}})} d\bar{S} = \sqrt{\frac{\det(\nabla(\pi \circ \mathbf{x}))^2}{\det(\bar{\mathbf{g}})}} d\bar{S}$$

As the fundamental form  $\bar{\mathbf{g}}$  is positive, the area element is also always greater than zero and evaluates to zero only for points on the silhouette whose normal is perpendicular to the viewing direction. Moreover, it does not distinguish front facing from back facing points on the surface. However, a reasonable measure of visibility should clearly reward front facing, and penalize back facing feature. We therefore formulate our screen size measure in terms of slightly modified signed version of the area element

$$d\sigma' = \frac{\det(\nabla(\pi \circ \mathbf{x}))}{\sqrt{\det(\bar{\mathbf{g}})}} d\bar{S}$$

While its absolute value equals that of the original area element, it has the same sign as  $\det(\nabla(\pi \circ \mathbf{x}))$ . Thus, it is positive if the map  $\pi \circ \mathbf{x}$  is orientation preserving and negative otherwise. Making the usual assumption of regularity on the parameterization  $\mathbf{x}$ , changes in the orientation of the combined map  $\pi \circ \mathbf{x}$  can only be due to back facing areas, i.e. points where the surface normal points away from the observer.

To obtain a zero bounded measure for screen size of features, we compute the signed increase in screen size as  $d\sigma'/d\bar{S}$  and plug it into the simple hull function



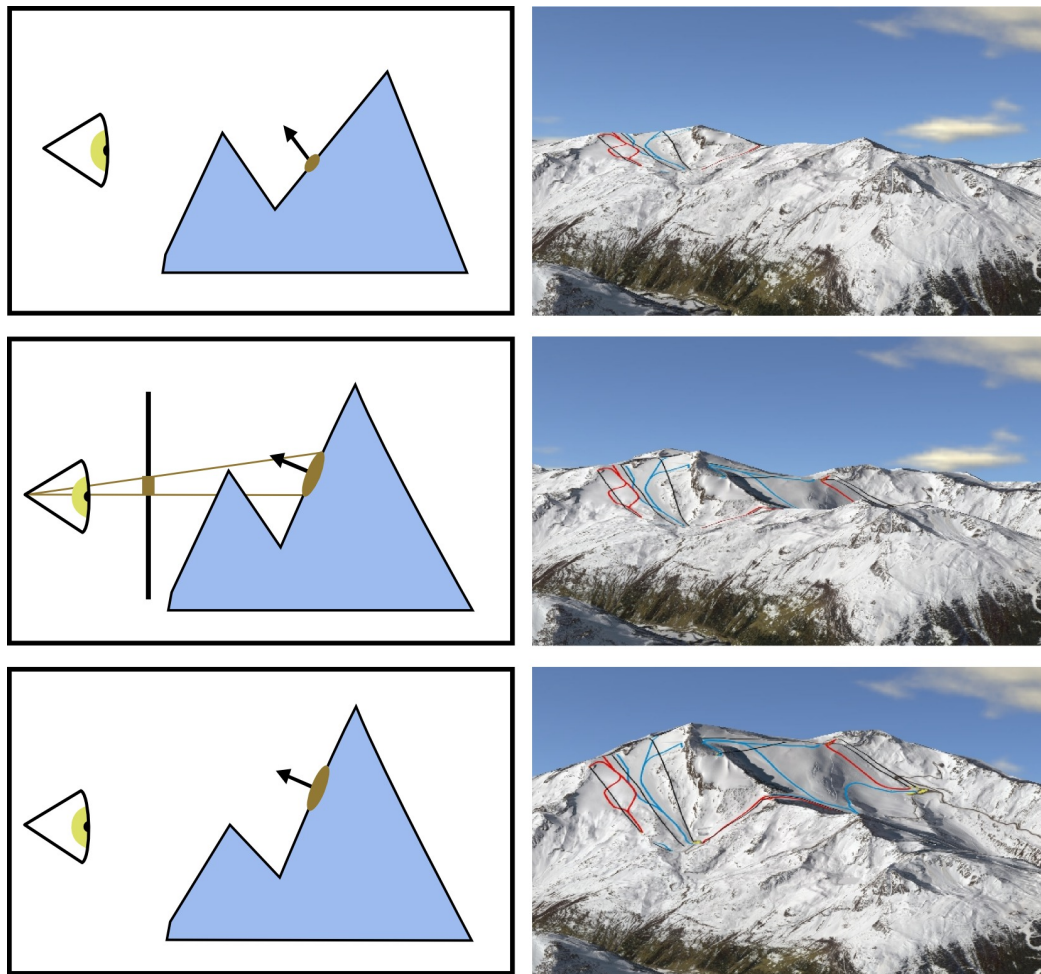


Figure 15.5: Causes of impaired feature visibility. First row: Original undeformed perspective. Features (ski slopes and lifts) are both small and to some extent occluded. Second row: The projected screen size of features has been increased by scaling features and turning their surface normal towards the observer. Still features can be invisible due to occlusion. Third row: Occlusion of features has been resolved by rearranging both occluders and occluded features.

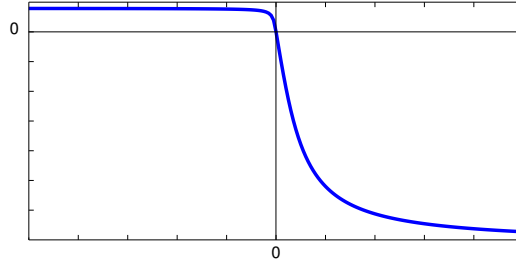


Figure 15.6: The hull function  $h_{screen}$  used in the definition of the screen size potential.

$h_{screen}$  defined as

$$h_{screen}(t) = \frac{\arctan(-st)}{s} \quad \text{with} \quad s = \frac{11 - 9 \operatorname{sign}(t)}{2} \quad (15.6)$$

which is shown in Figure 15.6. The exact choice of this function  $h_{screen}$  is of minor importance. It was primarily chosen to be smooth, bounded and monotonically decreasing as large increases in screen size should result in low energy values. Furthermore, its gradient nearly vanishes for negative values of  $d\sigma'$ , i.e. for points in back facing areas. In this way, minimizing the resulting potential will primarily influence front facing features while clearly back facing points are not affected. This is important to ensure that silhouettes, i.e. the curves on the surface where the orientation changes are mostly preserved. While it is theoretically possible to optimize both front and back facing features, we found that silhouettes are salient features of the landscape and thus have to be preserved to keep the panorama recognizable. The correct handling of back facing features will be further discussed in Section 16.5.2. By weighting with the feature map  $f$  and integrating over  $\bar{S}$  we obtain our screen size potential as:

$$E_{screen}[\tilde{\mathbf{x}}] = \int_{\bar{S}} f \cdot h_{screen}\left(\frac{d\sigma'}{d\bar{S}}\right) d\bar{S} \quad (15.7)$$

### 15.3.2 Occlusion Potential

To derive a measure of occlusion we start with the simple two-dimensional scenario and an orthographic projection as shown in top of Figure 15.7. For a feature point  $\mathbf{p}$  on  $S$  we find two direct occluding points  $\mathbf{q}'$  and  $\mathbf{q}''$  by following the projection ray from  $\mathbf{p}$  back to the camera. However, apart from those two, we can also consider each point  $\mathbf{q}$  on the surface between them as an occluder of  $\mathbf{p}$ , since it clearly has to be moved to make  $\mathbf{p}$  visible (middle of Figure 15.7). In this way

we define the set of points on the surface marked in dark blue the figure as occluders of  $\mathbf{p}$ . Denoting normalized up vector and viewing direction by  $\mathbf{s}$  and  $\mathbf{t}$  and setting

$$d_s := \mathbf{s}^t(\mathbf{q} - \mathbf{p}) \text{ and } d_t := \mathbf{t}^t(\mathbf{q} - \mathbf{p})$$

(see bottom of Figure 15.7) we find this set as

$$\{\mathbf{q} \in S \mid d_s > 0 \wedge d_t < 0\} \quad (15.8)$$

To quantify the degree to which  $\mathbf{q}$  occludes  $\mathbf{p}$  it is natural to choose either  $d_s$  or some function  $h_{occ}(d_s)$  that is monotonically increasing in  $d_s$ . As with the hull function  $h_{screen}$  defined in the last section, we require  $h_{occ}$  to be smooth and bounded from below in order to obtain a bounded occlusion measure. Furthermore, it should converge quickly to zero if  $\mathbf{q}$  lies below the dotted line ( i.e. for  $d_s < 0$  ) since in this case  $\mathbf{q}$  is not an occluder for  $\mathbf{p}$  anymore. However, there are ample reasons to extent the above set of occluders and to also penalize an occluder  $\mathbf{q}'$  that coincides with the dotted line or even if it lies below the line but not far from it. While such a point is not an occluder in the strict sense, we usually want features to lie clearly apart from silhouettes in the final image so that they are more noticeable. Technically, this is achieved by requiring  $h_{occ}(d_s)$  to be positive for  $d_s = 0$  and dropping the condition  $d_s > 0$  from the definition of the occluder set in Eq. 15.8.

Choosing  $h_{occ}$  as integral over a Gaussian

$$h_{occ}(t) = \int_{-\infty}^t g(s)ds \text{ with } g(s) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{s^2}{2\sigma^2}}$$

(see Figure 15.8) satisfies all the above requirements and facilitates the implementation as will become obvious in Section 16.3. Integrating over all occluders we find a measure for the occlusion of  $\mathbf{p}$  in the 2d case as

$$e_{2d}^{occ}(\mathbf{p}) = \int_{\{\mathbf{q} \mid d_t(\mathbf{q}) < 0\}} h_{occ}(d_s) dS \quad (15.9)$$

When looking at Figure 15.7, it might occur to the reader that  $e_{2d}^{occ}(\mathbf{p})$  can be chosen in a much simpler way as the area shown in light blue. While this choice is tempting, it is not advisable since minimizing this area will “narrow” the occluding hills and thus lead to unnecessary deformation. In the optimization hills therefore tend to get thinner or even collapse which is clearly not desirable. By formulating occlusion as a surface integral we avoid this problem: While a

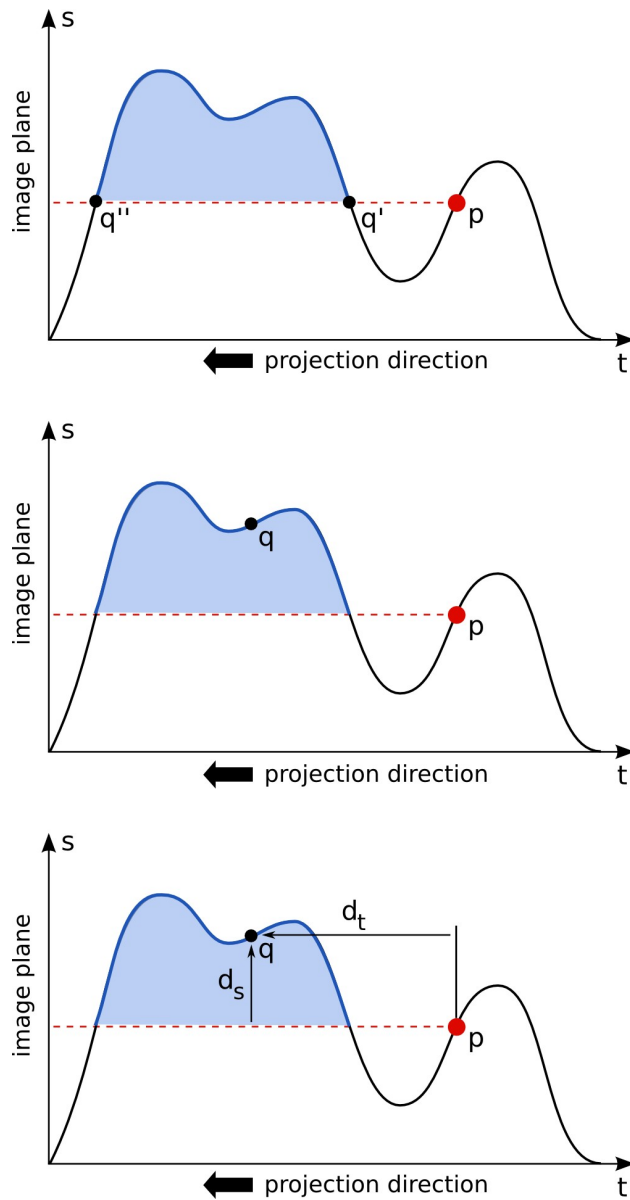


Figure 15.7: Simple 2d occlusion scenario. An orthographic camera is assumed to be on the left. Top: Projecting point  $p$  on the screen we find only two direct occluders  $q'$  and  $q''$ . Middle: However, point  $q$  can also be considered an occluder since it has to be moved to make  $p$  visible. Bottom: Definition of the signed distances  $d_s$  and  $d_t$ .

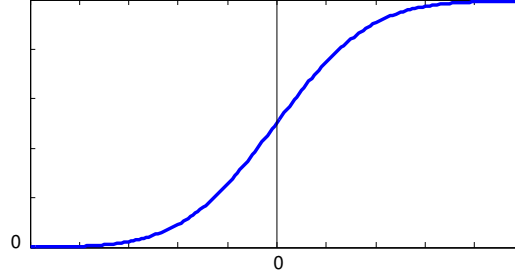


Figure 15.8: The hull function  $h_{occ}$  used in the definition of the occlusion potential.

compression of the occluding peak in Figure 15.7 decreases the area marked light blue, it does not alter the arc length of the occluding segment marked in dark blue and thus the surface integral in Equation 15.9 does not change.

To generalize the above measure to three dimensions, we first cut a slice from the surface  $S$  by intersecting it with the plane through  $\mathbf{p}$  spanned by  $\mathbf{s}$  and  $\mathbf{t}$  (see Figure 15.9a). We observe that within this slice we get the same situation as in the 2d case. For points  $\mathbf{q}$  within this plane we can therefore use  $e_{2d}^{occ}$  as defined above to measure their occlusion of  $\mathbf{p}$ .

However, by a similar argument as for the extension of the occluder set in the 2d case, it is reasonable to consider not only points within this plane as occluders but to extent it to points in neighbouring slices. Again, this is necessary to ensure that features will lie clear off silhouettes and are thus noticeable. We therefore further broaden the set of occluders by adding close points from both sides of the plane (see Figure 15.9b). For each of these additional points  $\mathbf{q}$  we compute the occlusion imposed on  $\mathbf{p}$  by evaluating  $e_{2d}^{occ}$  in the corresponding slice. Finally, we integrate over all slices weighting the contribution of each slice with the distance to the plane containing  $\mathbf{p}$ . To formalize this, we set

$$\mathbf{r} = \mathbf{s} \times \mathbf{t} \text{ and } d_r := \mathbf{r}^t(\mathbf{q} - \mathbf{p})$$

and write  $e_{2d}^{occ}(d_r) = e_{2d}^{occ}(\mathbf{p}, \mathbf{q})$  to denote the 2d occlusion measure evaluated in the slice at distance  $d_r$ . The extended occlusion measure can then be defined as

$$\begin{aligned} e_{3d}^{occ}(\mathbf{p}) &:= \int_{-\infty}^{\infty} e_{2d}^{occ}(d_r)g(d_r)d_r \\ &= \int_{\{\mathbf{q}|d_t(\mathbf{q})<0\}} h_{occ}(d_s)g(d_r)dS \end{aligned}$$

where the Gaussian  $g$  weights contributions of occluders outside the plane.

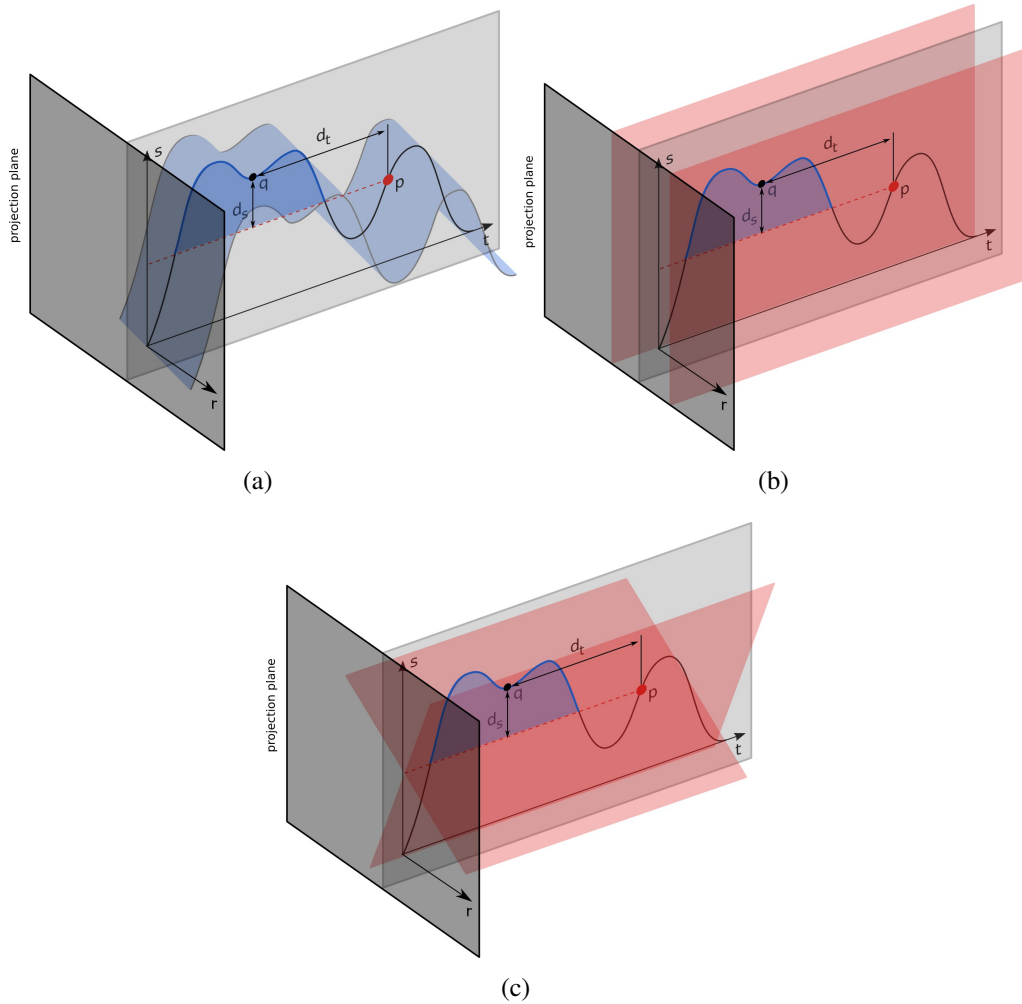


Figure 15.9: Generalization of the occlusion measure to 3d surfaces in three steps: (a) intersecting the surface with a plane spanned by  $s$  and  $t$  yields a similar situation as in the 2d example above. For points within this slice occlusion can thus be evaluated using  $e_{2d}^{occ}$ . (b) To have features lying clear off silhouettes, points  $q$  in neighbouring slices have also to be considered as occluders. In each of these slices occlusion can be evaluated using  $e_{2d}^{occ}$ . The overall occlusion is then obtained by integrating over all slices and weighting with a Gaussian. (c) In the 3d case occlusion can be resolved along multiple directions  $s$ . To account for this we evaluate  $e_{3d}^{occ}$  for a discrete set of directions  $s$  and sum all contributions.

So far, the above measure considers only occlusion relations along the up direction  $s$ . However, in three space occlusion can be resolved along any direction  $s$  in the image plane. Theoretically, the occlusion measure  $e_{3d}^{occ}$  must be minimized over all these directions  $s$  that are orthogonal on  $t$  (see Figure 15.9c). This is clearly prohibitive in practice as, first, taking a minimum yields a non-smooth function and thus severely complicates optimization and second requires extensive computation effort for each evaluation. As an approximation, we suggest to sum the above measure over a discrete set of directions  $D$ :

$$e_{3d}^{occ}(\mathbf{p}) = \sum_{s \in D} \int_{\{\mathbf{q} | d_t(\mathbf{q}) < 0\}} h_{occ}(d_s) \cdot g(d_r) dS \quad (15.10)$$

Fortunately, we found that for typical terrain geometry the direction of minimal occlusion is usually close to the up direction and thus a small set of directions distributed along the up direction is sufficient. This is because even in alpine terrain normals vary only little around the upward direction with only a very small percentage of surface area exhibiting slopes of more than 45 degrees.

Finally, we need to integrate the above defined occlusion measure over all feature points  $\mathbf{p}$  to obtain the occlusion potential as

$$E_{occ}[\tilde{\mathbf{x}}] = \int_{\tilde{S}} f \cdot e_{3d}^{occ}(\tilde{\mathbf{x}}) d\tilde{S} \quad (15.11)$$

where  $f$  is again the feature map introduced in Section 15.1.3.





---

## GENERATING PANORAMIC MAPS AUTOMATICALLY

---

At the heart of the automatic panorama generation is the optimization of the surface deformation  $\tilde{\mathbf{x}}$  which takes care of visible features and low shape deformation. Once this surface deformation is found, the final panorama is obtained by standard rendering and texture mapping methods. In this chapter we describe details of the optimization and implementation details of our method. In particular the optimization of the occlusion potential is computationally expensive as its evaluation involves a double integral over all surface points. To make optimization feasible, we present an approximation scheme that avoids the expensive evaluation of this integral in each optimization step. We also discuss the choice of parameters and finally present an experimental evaluation of our method on three different data sets.

### 16.1 Combining Shape and Visibility Potentials

The goal of the optimization is to find an at least locally optimal surface deformation  $\tilde{\mathbf{x}}^*$  that minimizes the four above defined shape and visibility potentials. As perfect shape preservation and optimal feature visibility are contradictory goals there is necessarily some trade off to be made. Consequently, we require  $\tilde{\mathbf{x}}^*$  to minimize a weighted average  $E$  of shape and visibility potentials defined as

$$E[\tilde{\mathbf{x}}] = (E_{bending}[\tilde{\mathbf{x}}] \quad E_{tangential}[\tilde{\mathbf{x}}] \quad E_{screen}[\tilde{\mathbf{x}}] \quad E_{occ}[\tilde{\mathbf{x}}]) * \beta \quad (16.1)$$

where  $\beta$  is the constant four dimensional vector

$$\beta = (\beta_{bending} \quad \beta_{tangential} \quad \beta_{screen} \quad \beta_{occ})^t \quad (16.2)$$

that controls the relative importance of the individual energies in this trade off. As we are only interested in the optimizing deformation  $\tilde{\mathbf{x}}^*$  and not in the actual value of  $E[\tilde{\mathbf{x}}^*]$  the vector  $\beta$  can be normalized which leaves three degrees of freedom. The actual choice of these parameters is a matter of personal preference and we

leave their choice to the user. As Section 16.5.2 will demonstrate the effect of these parameters on the panorama is very intuitive as they directly correspond to the four potentials measuring clearly defined properties of the panorama.

## 16.2 Discretization and Optimization

For numerical optimization we triangulate the parameter domain  $\omega$  using a regular triangular mesh  $\mathcal{M} = (V, \mathcal{T})$ . We discretize all involved maps by piecewise affine functions on this triangulation as described in Section 2.2. In particular, the piecewise affine parameterization  $\mathbf{x}$  and  $\bar{\mathbf{x}}$  are given as linear combinations

$$\begin{aligned} x_i &= \sum_{v \in V} x_i^v \Psi_v \\ \bar{x}_i &= \sum_{v \in V} \bar{x}_i^v \Psi_v \end{aligned}$$

for vertex coordinates  $\mathbf{x}^v, \bar{\mathbf{x}}^v \in \mathbb{R}^3$ . Both surfaces  $\bar{S}$  and  $S$  are then again triangular meshes with the same connectivity  $\mathcal{M}$ . The surface deformation  $\tilde{\mathbf{x}}$  is then also a piecewise affine map uniquely determined by the sets of coefficients  $\mathbf{x}^v, \bar{\mathbf{x}}^v \in \mathbb{R}^3$ . As the surface  $\bar{S}$  and  $\bar{\mathbf{x}}$  are assumed to be fixed, optimizing the surface deformation  $\tilde{\mathbf{x}}$  comes down to finding optimal vertex coefficients  $\mathbf{x}^v$ .

For the combined energy  $E$  we derive the analytic gradient with respect to  $\tilde{\mathbf{x}}$  and use the non-linear conjugate gradient optimization from the MATLAB optimization toolbox to find the (local) minimum  $\tilde{\mathbf{x}}^*$ . To speed up the optimization and to circumvent local minima we use a multi-grid approach: From the initial input surface we first compute a hierarchy of levels of detail with each level having half the resolution of its preceding level. As a coarsening operator, rather than simple sub sampling, spline patches on the coarse grid are fitted to the fine data to suppress aliasing. Starting from the coarsest level, the optimization is then run on each level of detail until convergence of the conjugate gradient solver and the resolution is subsequently doubled. The resulting solution is then used to initialize the optimization on the next level.

Both the membrane and screen size potentials are expressed in terms of metric tensors and thus involve only first order derivatives. As we assume piecewise affine maps, the metric tensors are constant within each triangle  $T \in \mathcal{T}$ . Expressions for the discrete metric tensor are given in Section 2.2.2.

In contrast, the bending potential is formulated in terms of mean curvatures and thus depends on second order derivatives of  $\bar{\mathbf{x}}$  and  $\mathbf{x}$  which are not available in our linear element based discretization. Resorting to higher order elements certainly solves this problem but the discretization is more involved. For simplicity, we therefore use the discretization of [GHDS03] which is based on a discrete

differential geometry approach. However, Grinspun et al. did not derive analytic derivatives for their energy but used comparatively slow automatic differentiation. More details on the discretization and analytic derivatives for individual potentials including  $E_{bending}$  are given in Section 16.4.

## 16.3 Approximation Scheme for the Occlusion Potential

The occlusion potential as defined above involves a double integral over the surface: one integral over all occluders  $\mathbf{q}$  and one over all occludees  $\mathbf{p}$ . A straightforward discretization would therefore require to sum over all triangle pairs of the discrete surface and thus has a complexity  $O(|\mathcal{T}|^2)$  in the number of triangles  $|\mathcal{T}|$ . Since minimization involves several thousand evaluations, this is clearly prohibitive. We therefore propose here an approximation with linear complexity.

To this extent, we first slightly rewrite the occlusion potential in coordinates of  $\omega$ . With  $\mathbf{p}', \mathbf{q}' \in \omega$  such that

$$\mathbf{p} = \mathbf{x}(\mathbf{p}') \text{ and } \mathbf{q} = \mathbf{x}(\mathbf{q}')$$

and using the notation  $1_{d_t < 0}$  to denote the indicator function that evaluates to 1 if  $d_t < 0$  and vanishes elsewhere we can write

$$\begin{aligned} E_{occ} &= \int_{\bar{S}} f \cdot e_{3d}^{occ}(\tilde{\mathbf{x}}) d\bar{S} & (16.3) \\ &= \sum_{s \in D} \int_{\bar{S}} \int_{\{\mathbf{q} \in S \mid d_t < 0\}} f h_{occ}(d_s) g(d_r) dS d\bar{S} \\ &= \sum_{s \in D} \int_{\bar{S}} \int_S 1_{d_t < 0} f h_{occ}(d_s) g(d_r) dS d\bar{S} \\ &= \sum_{s \in D} \int_{\omega} \int_{\omega} 1_{d_t(\mathbf{p}, \mathbf{q}) < 0} f(\bar{\mathbf{x}}(\mathbf{p}')) h_{occ}(d_s(\mathbf{p}, \mathbf{q})) \cdot \\ &\quad \cdot g(d_r((\mathbf{p}, \mathbf{q}))) (\det \mathbf{g}(\mathbf{q}'))^{\frac{1}{2}} (\det \bar{\mathbf{g}}(\mathbf{p}'))^{\frac{1}{2}} d\mathbf{q}' d\mathbf{p}' & (16.4) \end{aligned}$$

where in the last equality we made all dependencies on the integration variables  $\mathbf{p}'$  and  $\mathbf{q}'$  explicit. Now, for a fixed point  $\mathbf{o} \in \mathbb{R}^3$  we define two auxiliary functions

$$\begin{aligned} s_{occ}(\mathbf{o}) &= \sum_{s \in S} \int_{\omega} 1_{d_t(\mathbf{o}, \mathbf{q}) < 0} g(d_r(\mathbf{o}, \mathbf{q})) (\det \mathbf{g}(\mathbf{q}'))^{\frac{1}{2}} d\mathbf{q}' \\ s_{ocd}(\mathbf{o}) &= \sum_{s \in S} \int_{\omega} f(\bar{\mathbf{x}}(\mathbf{p}')) 1_{d_t(\mathbf{p}, \mathbf{o}) < 0} h_{occ}(d_s(\mathbf{p}, \mathbf{o})) g(d_r(\mathbf{p}, \mathbf{o})) (\det \bar{\mathbf{g}}(\mathbf{p}'))^{\frac{1}{2}} d\mathbf{p}' \end{aligned}$$

These two functions are actually derived from the right hand side of Equation 16.4 by omitting the integral over occluders  $\mathbf{q}$  and occludees  $\mathbf{p}$  respectively and substituting  $\mathbf{o}$  for the omitted integration variable. The first function  $s_{occ}$  is in fact very similar to the per-point occlusion measure  $e_{3d}^{occ}$  defined in Section 15.3.2. The only difference is that  $s_{occ}$  is defined for arbitrary points  $\mathbf{o}$  in three-space while  $e_{3d}^{occ}$  is restricted to points  $\mathbf{p}$  on the surface  $S$ . Analogously to  $e_{3d}^{occ}$ , it thus quantifies how much a point  $\mathbf{o}$  in space is occluded by the surface  $S$ . Using the function  $s_{occ}$ , Equation 16.4 can thus be equivalently written as:

$$E_{occ} = \int_{\omega} f(\bar{\mathbf{x}}(\mathbf{p}')) s_{occ}(\mathbf{p}) (\det \bar{\mathbf{g}}(\mathbf{p}'))^{\frac{1}{2}} d\mathbf{p}' \quad (16.5)$$

Likewise, in the definition of the function  $s_{ocd}$  the occluding surface point  $\mathbf{q}$  is replaced by an arbitrary point  $\mathbf{o}$  in three space. It complements  $s_{occ}$  in the sense that it measures how much a point  $\mathbf{o}$  occludes important feature points on the surface  $S$ . Yet another way to write the occlusion potential is:

$$E_{occ} = \int_{\omega} s_{ocd}(\mathbf{q}) (\det \mathbf{g}(\mathbf{q}'))^{\frac{1}{2}} d\mathbf{q}' \quad (16.6)$$

which can be easily verified by substituting the definition of  $s_{occ}$  and comparing with Equation 16.4.

For discretization, we replace the two functions  $s_{\{occ,ocd\}}$  by two approximations  $\hat{s}_{\{occ,ocd\}}$  sampled on a discrete space grid as will be explained below. By Equations 16.5 and 16.6 we then in turn find two approximations of  $E_{occ}$  substituting  $\hat{s}_{\{occ,ocd\}}$  for  $s_{\{occ,ocd\}}$ . Defining  $\hat{E}_{occ}$  in terms of the approximation  $\hat{s}_{occ}$  by Eq. 16.5 accounts well for the case that a surface point  $\mathbf{p}$  is occluded by surface parts lying before it but only indirectly accounts for the occlusion imposed by  $\mathbf{p}$  onto subjacent regions. Likewise, an approximation based on Eq. 16.6 accounts for the occlusion imposed by  $\mathbf{q}$  onto other parts but neglects the occlusion of point  $\mathbf{q}$  itself. To obtain a better approximation we therefore combine both these approximations

$$\hat{E}_{occ} = \frac{1}{2} \int_{\omega} f(\bar{\mathbf{x}}(\mathbf{p}')) \hat{s}_{occ}(\mathbf{p}) (\det \bar{\mathbf{g}}(\mathbf{p}'))^{\frac{1}{2}} d\mathbf{p}' + \frac{1}{2} \int_{\omega} \hat{s}_{ocd}(\mathbf{q}) (\det \mathbf{g}(\mathbf{q}'))^{\frac{1}{2}} d\mathbf{q}'$$

The approximations  $\hat{s}_{occ}$  and  $\hat{s}_{ocd}$  themselves are computed by sampling the functions  $s_{\{occ,ocd\}}$  on a discrete regular voxel grid in screen space and trilinear interpolation of these samples. In practice, the two grids can be computed quite efficiently: First, the surface  $S$  is rasterized into a 3d voxel grid. On modern graphics hardware this operation is supported in hardware which makes it very fast. For  $\hat{s}_{ocd}$  the surface is textured with the function  $f \cdot \frac{d\bar{S}}{dS}$  to account for the feature map and the area element of  $\bar{S}$ . For  $\hat{s}_{occ}$ , as it is defined as integral over  $S$  no such correction is necessary and we set the texture to 1.

Now, for an efficient computation of  $\hat{s}_{\{occ,ocd\}}$  it is important to observe, that the three remaining factors  $1_{d_t(\mathbf{p},\mathbf{q})<0}$ ,  $h_{occ}(d_s(\mathbf{p},\mathbf{q}))$  and  $g(d_r(\mathbf{p},\mathbf{q}))$  in the definition of  $s_{\{occ,ocd\}}$  depend only on one of the distances  $d_s$ ,  $d_t$  and  $d_r$  of  $\mathbf{p}$  and  $\mathbf{q}$  along the orthogonal directions  $\mathbf{r}$ ,  $\mathbf{s}$  and  $\mathbf{t}$  respectively. The integral is therefore equivalent to three one dimensional convolutions of the rasterized surface: convolution with the Gaussian  $g$  along  $\mathbf{r}$ , convolution with  $h_{occ}$  along  $\mathbf{s}$ , and convolution with  $1_{<0}$  along  $\mathbf{t}$ .

On a discrete grid aligned with the direction  $\mathbf{r}$ ,  $\mathbf{s}$ ,  $\mathbf{t}$ , the latter simplifies to a cumulative sum along the  $\mathbf{t}$  direction. Moreover, convolving with  $h_{occ}$  is by its definition also equivalent to a convolution with the Gaussian followed by a convolution with  $1_{<0}$ . After rasterizing the textured surfaces, computing  $\hat{s}_{\{occ,ocd\}}$  thus comes down to two convolutions with the Gaussian along  $\mathbf{s}$  and  $\mathbf{r}$  directions followed by two cumulative sums in  $\mathbf{t}$  and  $\mathbf{s}$  direction. In our implementation we use an efficient one dimensional convolution operator that exploits the sparsity of the rasterized surface.

Even with the above described improvements a computation of  $\hat{s}_{\{occ,ocd\}}$  for every evaluation of  $\hat{E}_{occ}$  is still too costly to be practicable. However, since  $s_{ocd}$  and  $s_{occ}$  integrate over the set of all occludees and occluders, they usually change quite slowly. We found it therefore sufficient to defer the update of the grids  $\hat{s}_{\{occ,ocd\}}$  over several evaluations. In our implementation grids are only updated when the deformed surface  $S$  has moved significantly, but at least once per level of detail.

More details on the computation of  $\hat{E}_{occ}$  and its derivative can be found in Section 16.4.

## 16.4 Details on Discretizations and Derivatives

### 16.4.1 Discretization of $E_{tangential}$

The membrane potential is nearly identical to the parameterization potential described in Section 5.4. As the only difference lies in the generalized metric tensor, the discretization and derivatives from Part II can be used and the generalized metric tensor  $\mathbf{G}_{\tilde{x}} = \mathbf{g}\tilde{\mathbf{g}}^{-1}$  substituted for the metric tensor  $\tilde{\mathbf{g}}$ .

For the alternative potential by Clarenz et al. used in our implementation, a detailed description of its discretization as well as derivatives can be found in the original paper [CLR04b].

## 16.4.2 Derivatives of $E_{bending}$

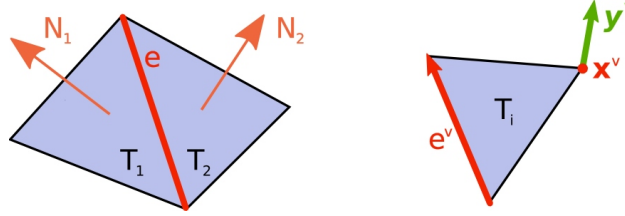
As derived in [GHDS03], the discrete curvature energy is given for a pair of adjacent triangles by:

$$e_{bending} = (\alpha - \bar{\alpha})^2 \|\mathbf{e}\| \quad (16.7)$$

where  $\alpha$  and  $\bar{\alpha}$  denote the signed dihedral angle in  $S$  and  $\bar{S}$ , respectively and  $\mathbf{e}$  the edge vector of the common edge in  $S$ . A complication in the derivation arises if the dihedral angle is computed using the dot product due to missing smoothness of arccos at  $\alpha = 0$ . To circumvent this issue, we compute  $\alpha$  as

$$\alpha = 2 \arcsin((\mathbf{e}/\|\mathbf{e}\|)^t (\mathbf{H} \times \mathbf{N}_1)) \quad (16.8)$$

where  $\mathbf{H}$  denotes the normalized halfway vector between  $\mathbf{N}_1$  and  $\mathbf{N}_2$  (see figure).



Using the notation from the above figure and  $\hat{\mathbf{e}} := \mathbf{e}/\|\mathbf{e}\|$ , the derivative of  $\alpha$  with respect to a variation  $\mathbf{y}^v$  of a vertex  $\mathbf{x}^v$  incident with  $T_1$  or  $T_2$  is now given by

$$\begin{aligned} \frac{\partial \alpha}{\partial \mathbf{x}^v}[\mathbf{y}^v] &= (\|\mathbf{N}_1 + \mathbf{N}_2\|)^{-1} (1 - (\hat{\mathbf{e}})^t (\mathbf{H} \times \mathbf{N}_1)^2)^{\frac{1}{2}} \cdot \\ &\left( \left\langle \frac{\partial \mathbf{N}_2}{\partial \mathbf{x}^v} \times \mathbf{N}_1 \middle| \hat{\mathbf{e}} \right\rangle + \left\langle \mathbf{N}_2 \times \frac{\partial \mathbf{N}_1}{\partial \mathbf{x}^v} \middle| \hat{\mathbf{e}} \right\rangle + \left\langle \mathbf{N}_2 \times \mathbf{N}_1 \middle| \frac{\partial \hat{\mathbf{e}}}{\partial \mathbf{x}^v} \right\rangle - \right. \\ &\left. - \frac{\langle \mathbf{N}_2 \times \mathbf{N}_1 | \hat{\mathbf{e}} \rangle}{\|\mathbf{N}_1 + \mathbf{N}_2\|^2} \left\langle \mathbf{N}_1 + \mathbf{N}_2 \middle| \frac{\partial \mathbf{N}_1}{\partial \mathbf{x}^v} + \frac{\partial \mathbf{N}_2}{\partial \mathbf{x}^v} \right\rangle \right). \end{aligned}$$

In this expression, the derivatives of the normals  $\mathbf{N}_i$  for  $i = 1, 2$  evaluate to

$$\frac{\partial \mathbf{N}_i}{\partial \mathbf{x}^v}[\mathbf{y}^v] = \frac{1}{2|T_i|} (\text{id} - \mathbf{N}_i \mathbf{N}_i^t) (\mathbf{y}^v \times \mathbf{e}^v)$$

if  $v$  is incident to the triangle  $T_i$  and zero otherwise, with  $\mathbf{e}^v$  denoting the edge opposite to  $v$  in  $T_i$ . If the vertex  $v$  is incident to both triangles, the derivative of the normalized shared edge  $\hat{\mathbf{e}}$  is also non-zero and evaluates to

$$\frac{\partial \hat{\mathbf{e}}}{\partial \mathbf{x}^v}[\mathbf{y}^v] = \frac{1}{\|\mathbf{e}\|} (\text{id} - \hat{\mathbf{e}} \hat{\mathbf{e}}^t) \mathbf{y}^v$$

Finally, we can write down the derivative of  $e_{bending}$  with respect to a variation  $\mathbf{y}^v$  of a vertex  $v$  as

$$\frac{\partial e_{bending}}{\partial \mathbf{x}^v}[\mathbf{y}^v] = 2(\alpha - \bar{\alpha}) \|\mathbf{e}\| \frac{\partial \alpha}{\partial \mathbf{x}^v}[\mathbf{y}^v] - (\alpha - \bar{\alpha})^2 \langle \hat{\mathbf{e}} | \mathbf{y}^v \rangle$$

### 16.4.3 Discretization of $E_{screen}$

Using rules of the calculus of variation the derivative of  $E_{screen}$  with respect to  $\mathbf{x}$  in the direction  $\mathbf{y}$  is found as

$$\partial_{\mathbf{x}} E_{screen}[\mathbf{x}](\mathbf{y}) = - \int_{\omega} \frac{f \circ \bar{\mathbf{x}}}{1 + (sd\sigma')^2} \partial_{\mathbf{x}} d\sigma' d\theta$$

with the function  $s$  defined as in Equation 15.6 and

$$\partial_{\mathbf{x}} d\sigma'[\mathbf{x}](\mathbf{y}) = \mathbf{r}^t(\mathbf{y}_{,1} \times \mathbf{x}_{,2} + \mathbf{x}_{,1} \times \mathbf{y}_{,2})$$

where  $\mathbf{r}$  is the view direction of the orthographic projection  $\pi$ . For discretization, we remark that the area element  $d\sigma'$  as defined in Equation 15.3.1 is a function of the first order derivatives of  $\mathbf{x}$  and the affine map  $\pi$  only. It is therefore constant within each triangle  $T \in \mathcal{T}$  and evaluates to

$$d\sigma'_T = \mathbf{r}^t(\mathbf{x}_{,1} \times \mathbf{x}_{,2}) |T_{\omega}| / |T_{\bar{S}}|$$

Setting  $F_T := \int_{T_{\bar{S}}} f d\bar{S}$  the discrete screen size energy can be written as

$$E_{screen}[\mathbf{x}] = \sum_{T \in \mathcal{T}} F_T \cdot h(d\sigma'_T)$$

The derivative with respect to a variation  $\mathbf{y}^v$  of the coefficients  $\mathbf{x}^v$  can easily be obtained by evaluating the above given derivative for the special choice of  $\mathbf{y} = \mathbf{y}^v \Psi_v$  (where  $v$  is not summed). This gives

$$\begin{aligned} \partial_{\mathbf{x}} E_{screen}[\mathbf{x}](\mathbf{y}^v \Psi_v) = \\ \sum_{T \text{ adjacent } v} \frac{F_T}{1 + (sd\sigma'_T)^2} \sum_{w \in T} \mathbf{r} \times \mathbf{y}^v \nabla \Psi_w \mathbf{R}^{\frac{\pi}{2}} \nabla \Psi_v \quad (v \text{ not summed}) \end{aligned}$$

where  $R^{\frac{\pi}{2}}$  is clockwise rotation in  $\omega$  by  $\frac{\pi}{2}$  degree.

### 16.4.4 Discretization of $E_{occ}$

The evaluation of  $\hat{E}_{occ}$  requires the integration of  $\hat{s}_{\{occ, ocd\}}$  over the surface  $S$ . For a piecewise affine map  $\mathbf{x}$  this is equivalent to integrating the functions over each triangle  $T_S$  of  $S$  and summing over all triangles.

As in our implementation the functions  $\hat{s}_{\{occ, ocd\}}$  are discretized on a regular grid, we have to perform some kind of numerical integration. For efficiency, we suggest the following simple scheme: For each triangle, we fit a linear function to

the values of  $\hat{s}_{\{occ,ocd\}}$  at the vertices of  $T_S$  such that its gradient equals the average of  $\nabla \hat{s}_{\{occ,ocd\}}$  at the three vertices. This gives the simple approximations:

$$\int_{T_S} \hat{s}_{\{occ,ocd\}} dS = (1/3)|T_S| \sum_{v \in T} \hat{s}_{\{occ,ocd\}}(\mathbf{x}^v)$$

$$\int_{T_S} \nabla \hat{s}_{\{occ,ocd\}} dS = (1/3)|T_S| \sum_{v \in T} \nabla \hat{s}_{\{occ,ocd\}}(\mathbf{x}^v)$$

where we use finite difference approximations for the gradients  $\nabla \hat{s}_{\{occ,ocd\}}$ . Setting

$$\hat{s}_{\{occ,ocd\}}^T := (1/3) \sum_{v \in T} \hat{s}_{\{occ,ocd\}}(\mathbf{x}^v) \text{ and}$$

$$\nabla \hat{s}_{\{occ,ocd\}}^T := (1/3) \sum_{v \in T} \nabla \hat{s}_{\{occ,ocd\}}(\mathbf{x}^v)$$

the discrete occlusion potential is then given as

$$\hat{E}_{occ}[\mathbf{x}] = \frac{1}{2} \sum_{T \in \mathcal{T}} F_T \hat{s}_{occ}^T + |T_S| \hat{s}_{ocd}^T$$

for a piecewise affine  $\mathbf{x}$  and its derivative in direction  $\mathbf{y}^v$  is given by

$$\partial_{\mathbf{x}} \hat{E}_{occ}[\mathbf{x}](\mathbf{y}^v \Psi_v) = \frac{1}{6} \sum_{T \in \mathcal{T}} F_T \nabla \hat{s}_{occ}^T$$

$$+ \frac{1}{6} \sum_{T \in \mathcal{T}} |T_S| \nabla \hat{s}_{ocd}^T + \frac{1}{2} \sum_{T \in \mathcal{T}} \hat{s}_{ocd}^T \sum_{w \in T} \frac{(\mathbf{e}^w)^t \mathbf{e}^v}{4|T_S|} \mathbf{x}^w \quad (\mathbf{v} \text{ not summed})$$

where  $\mathbf{e}_v$  denotes the edge lying opposite of vertex  $v$  in  $T$ .

## 16.5 Results and Discussion

### 16.5.1 Experimental Results

To evaluate our method we applied it to three different datasets with varying resolution, geometric complexity and feature richness. The feature sets and geographical regions were chosen according to different applications of our method. They are summarized in Figure 16.1.

For all the examples shown we used a regular triangular mesh with resolutions between 1024 to  $16k$  vertices on the finest level of detail. The resolution of the voxel grids  $\hat{s}_{\{occ,ocd\}}$  was chosen four times higher than the mesh resolution on



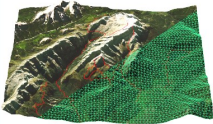

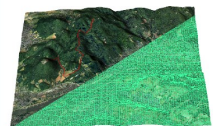

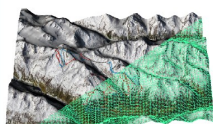

Dataset		Dimensions	Features
Area	Zugspitze, Alps, Bavaria	 21 x 21 km 64 x 64 vertices	 hiking trails
Type	tourist map		
Processing Time	5.3 min		
Area	Siebengebirge, Bonn	 8 x 8 km 128 x 128 vertices	 single GPS trail
Type	personal map		
Processing Time	33.8 min		
Area	Sölden, Alps, Austria	 16 x 16 km 64 x 64 vertices	 ski slopes and ski lifts
Type	ski map		
Processing Time	4.2 min		

Figure 16.1: A summary of the three data sets and application scenarios.

each level of detail. Although our current Matlab-based implementation is not optimized for speed, the optimization took only about 5 minutes for the lower resolution and about 33 minutes on the high resolution data set. Compared to the weeks and months needed for manual design, these times are acceptable.

The first scenario is a tourist map, one of the classical applications of panorama maps. The data set consists of elevation data and aerial images of the Zugspitze area, a famous mountain in the south of Germany. As typical for this application, hiking trails in this area were chosen as features which were imported from an a database of GPS tracks.

As described in the introduction, panoramas are usually drawn using an airplane perspective (see Figure 15.1), i.e. the view direction is nearly orthogonal to the ground in the foreground and parallel to it in the back. Such an effect can easily be incorporated in our method by mapping the valley in the foreground to a cylindrical surface prior to the optimization. Besides resolving some of the occlusions in the foreground, the airplane perspective also has to advantage of closing the bottom of the panorama image. A comparison of original and airplane perspective as well as the final panorama for the Zugspitze area are shown in 16.2.

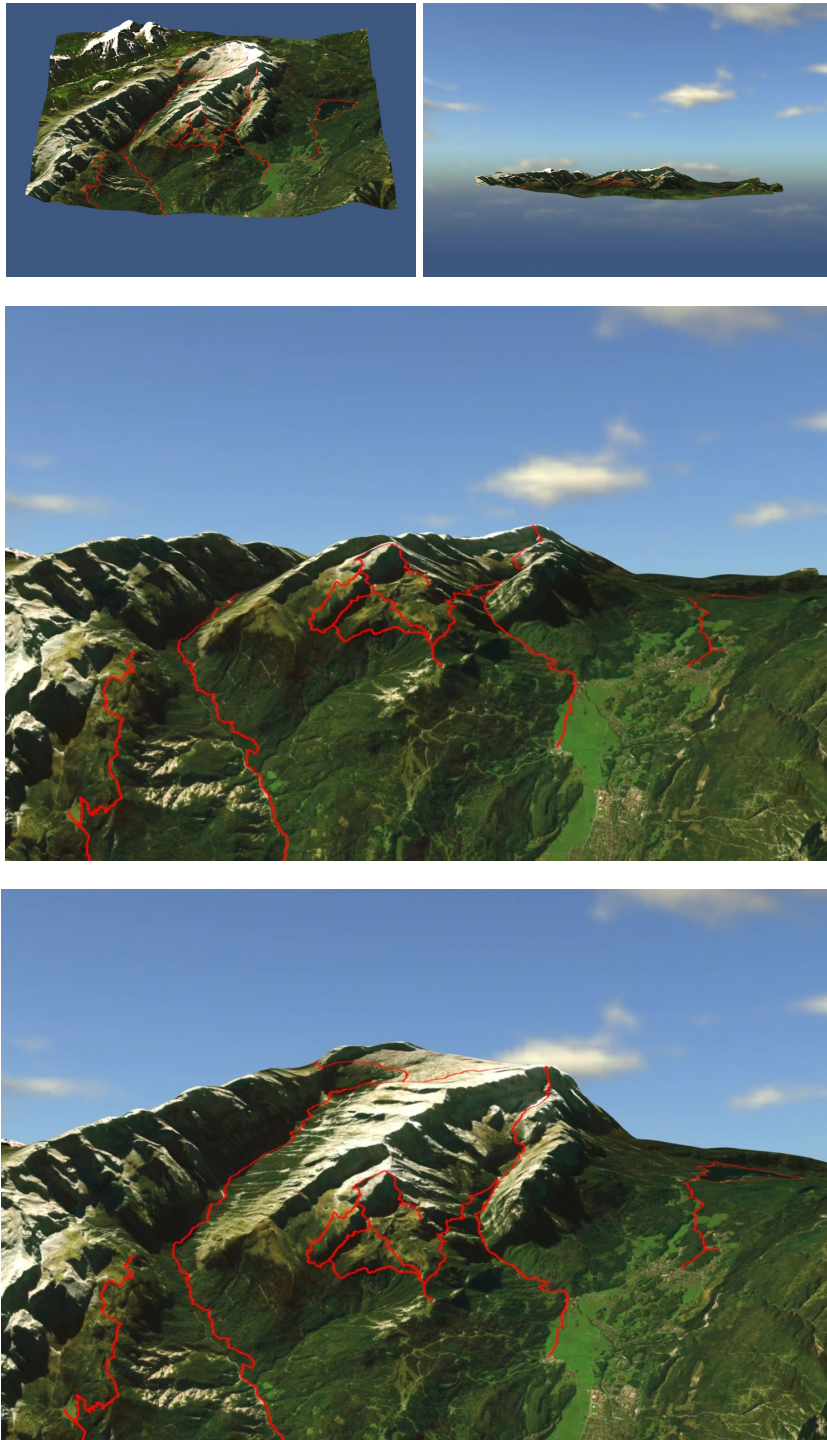


Figure 16.2: A tourist map of the Zugspitze showing hiking tracks. Top: View from on top and user specified original view. Middle: The result after applying a cylinder mapping to the foreground. Bottom: Final result after visibility optimization.

As shown in the figure, the airplane perspective already enhances visibility of features in the foreground. However, trails running in the background to the peak and a trail around a lake in the right become visible only after the optimization.

To visualize membrane strain, i.e. deformation of texture shapes we also mapped a checker board texture onto both the original and the deformed surface (see Figure 16.3). As visible in these renderings membrane strain is distributed nearly evenly over the whole surface and only slightly noticeable in the vicinity of features.

The second application scenario is a new kind of map which we have coined “personal panorama”. It is a visualization of a single path, e.g. an intended hiking trip or a personal GPS trace, which was recorded during a biking tour. While traditional design is too laborious to allow for the visualization of such data in a panorama, it becomes feasible with our method. For this example we used the GPS trace of a hiking trip in the Siebengebirge, a mountainous area in western Germany. The resulting panorama is shown in Figure 16.4. As the mountains in this area are only a hundred meters high, we vertically scaled the input data by a factor of 2 to increase impressiveness.

As a third application we applied our method to data from the Austrian ski resort Sölden to create a ski map. In the original data set that consists of a high resolution aerial color image and a  $2.5D$  height field, an additional layer describes ski slopes, lifts and the outline of buildings as vector data for all areas in the resort including two areas located on the glacier — Tiefenbach and Rettenbach. From this layer we created the feature map  $f$  by rasterization.

Figure 14.3 shows the Tiefenbach and Rettenbach areas as seen from the Venter valley. As demonstrated in the Figure the optimized surface shown on the right allocates far more screen space to features (slopes, lifts and buildings) and avoids occlusion completely. In particular, slopes on the Rettenbach glacier (marked in yellow) are disclosed in the optimized rendering.

Figure 16.5 demonstrates that our method is by no means limited by the number of features and can even handle complex feature arrangements. The figure shows the full ski resort with all slopes and lifts as seen from the Ötz valley. This viewpoint is similar to the one of the hand drawn panorama shown in Figure 16.6 with the Gaislachkogel in the front.

The importance of the screen size measure was slightly reduced in this example in favor of a better shape preservation. To ensure that all slopes are nevertheless visible the influence of the occlusion terms was increased accordingly. Despite we made these simple manual interventions, the required user interaction is negligible considering the weeks and months of work needed to draw a panorama manually. At the same time, the results of our method is very convincing and

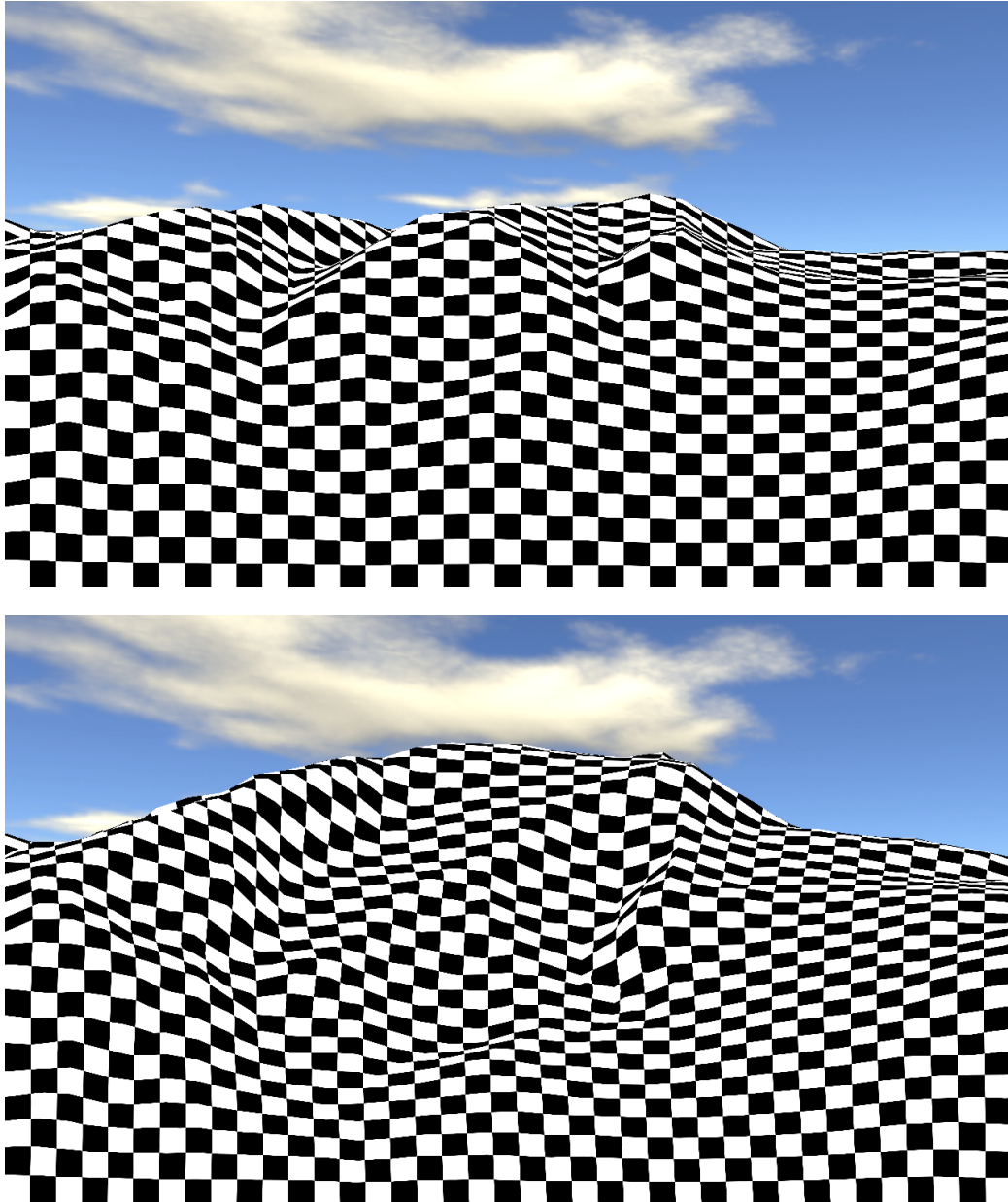


Figure 16.3: The Zugspitze panorama shown with a checker board texture to visualize membrane strain. The cylinder mapped surface is shown on top. Membrane strain on the deformed surface  $S$  (bottom) is nearly evenly distributed over the whole surface and locally hardly noticeable (The pattern is very well preserved and the checkers vary only little in size and shape.)



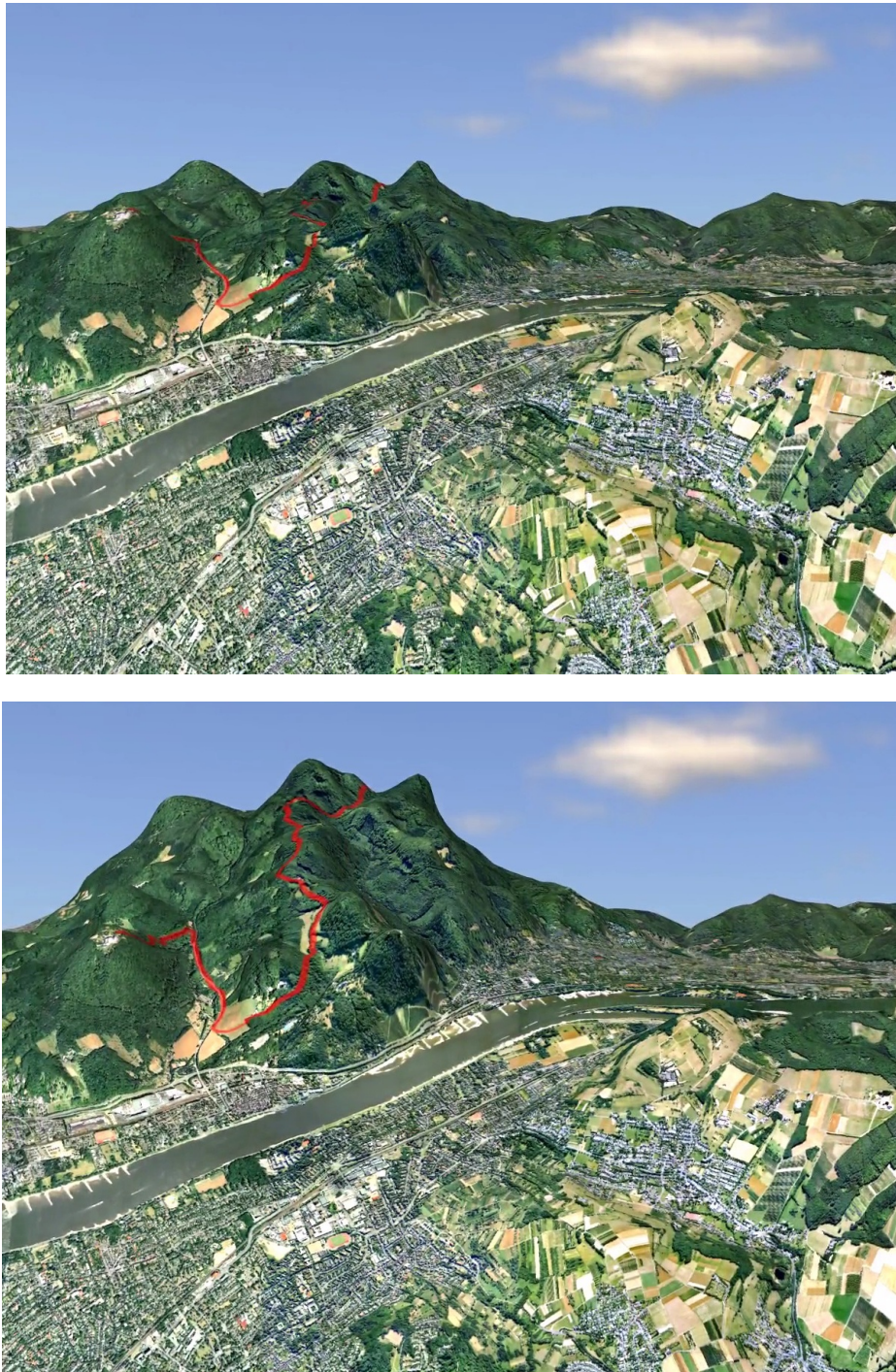


Figure 16.4: A personal panorama generated for a GPS trace of a hiking trail in the Siebengebirge, a mountainous area in western Germany. The top shows the original data after cylinder mapping and vertical scaling. The result of the optimization is shown in the bottom.

the geographic arrangement resembles that of the hand-drawn map in Figure 16.6 even though the latter naturally shows far more artful labeling and illustration of details. Please note, that in our current visualizations features are simply rendered into a texture and mapped onto the surface together with the aerial image. Adding and rendering 3D labels instead would clearly improve readability further.

### 16.5.2 Discussion

Similar to manual panorama design, the choice of a suitable view point has a large impact on the quality of the resulting panorama. It should be chosen so that only few features lie on back facing parts of the surface, since to make such features visible large shape deformations are necessary. This is demonstrated in Figure 16.7 where a bad viewpoint was chosen intentionally. Whereas back facing features close to silhouettes are handled gracefully by our method, the high shape deformation is unavoidable in this example as approximately half of the features are back facing and in particular silhouettes become less recognizable. One simple solution is therefore to detect and remove back facing features that are far from the silhouette. The resulting panorama is shown in the bottom of Figure 16.7. However, in our experiments we found that in such cases a better, but still earth bound view point can often be found that shows significantly less back facing features. As problematic situations are easily detected, an ideal extension to our method would be a semi interactive view point selection, that automatically suggest feasible view points to the user.

To illustrate the effect of the individual potentials and the influence of the  $\beta$  parameters on the optimizations, Figure 16.8 shows the result of our method for some parameter variations. The first row shows the undeformed original and a reference panorama of the surface. The results in the following rows were obtained by either setting the respective  $\beta$  parameter to zero (first column) or by scaling the parameter value of the reference deformation by five (second column). From the first column it becomes obvious that indeed all four potentials are necessary to obtain a reasonable result. While  $E_{bending}$  and  $E_{tangential}$  are vital for the shape preservation, a lack of the screen size component results in unoccluded but hardly visible features and without the occlusion component approximately half of the features in the Rettenbach area remain occluded. On the other hand, an increase in the  $\beta$  parameters that correspond to occlusion and screen size yields highly visible features but also induces significantly higher shape deformation.

Please note also the occlusion in the absence of the bending potential (in both Figure 16.8 and 15.4): In this particular case the surface folds into multiple flat



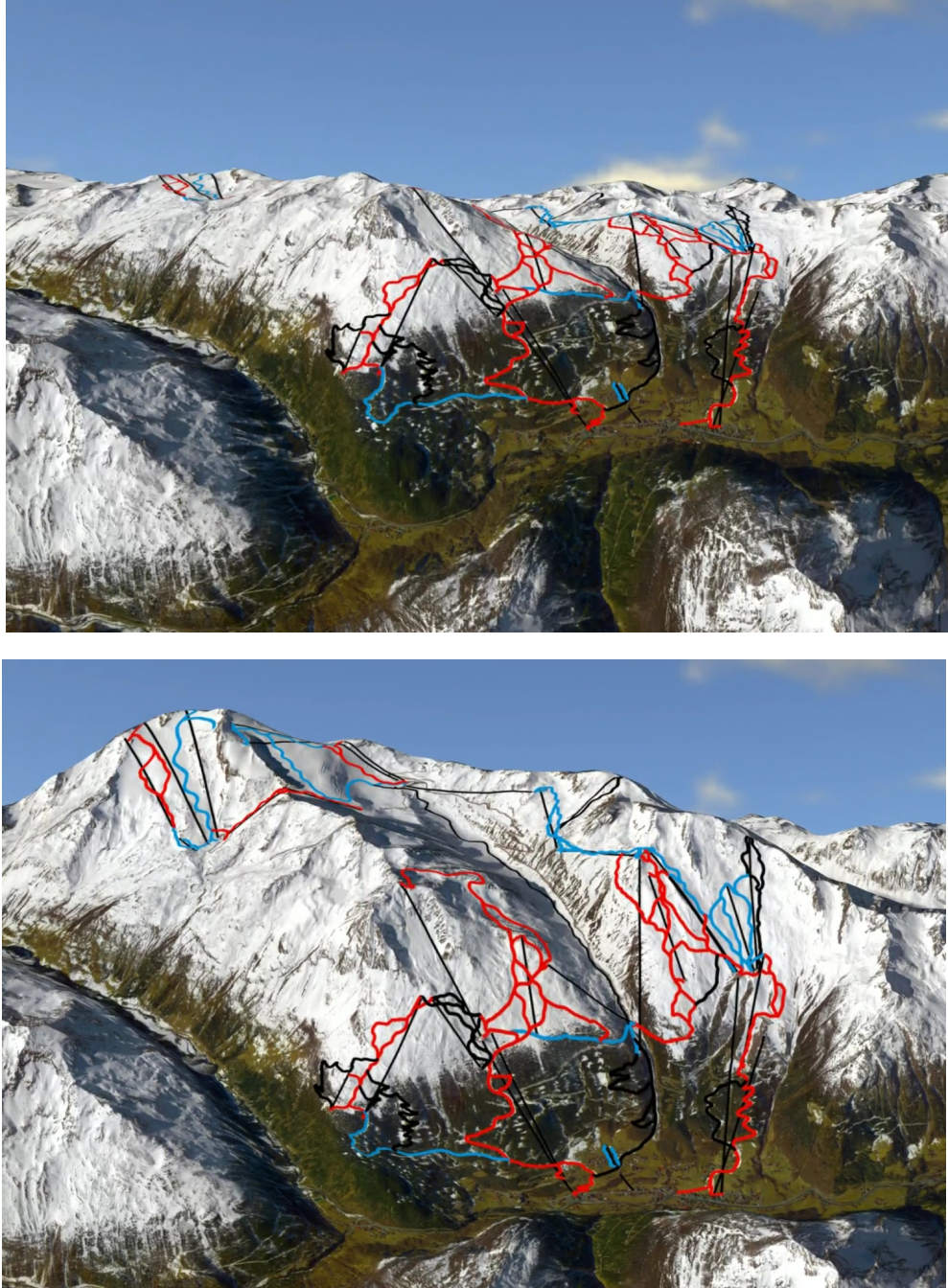


Figure 16.5: The complete Sölden ski resort seen from the valley of Ötz. The top image shows the original view after cylinder mapping, the deformed view is shown at the bottom. Note, how slopes in the glacier area in the left background and along the valley running in the middle become uncovered by the optimization.



Figure 16.6: A hand drawn ski map for the ski resort Sölden taken from [wwwb] (Copyright by Bergbahnen Sölden). For comparison a cutout was chosen that corresponds to the computed panorama shown in Figure 16.5. While the hand drawn map shows far more artful labeling and illustration of details, the geographic arrangement resembles that in the computed panorama.



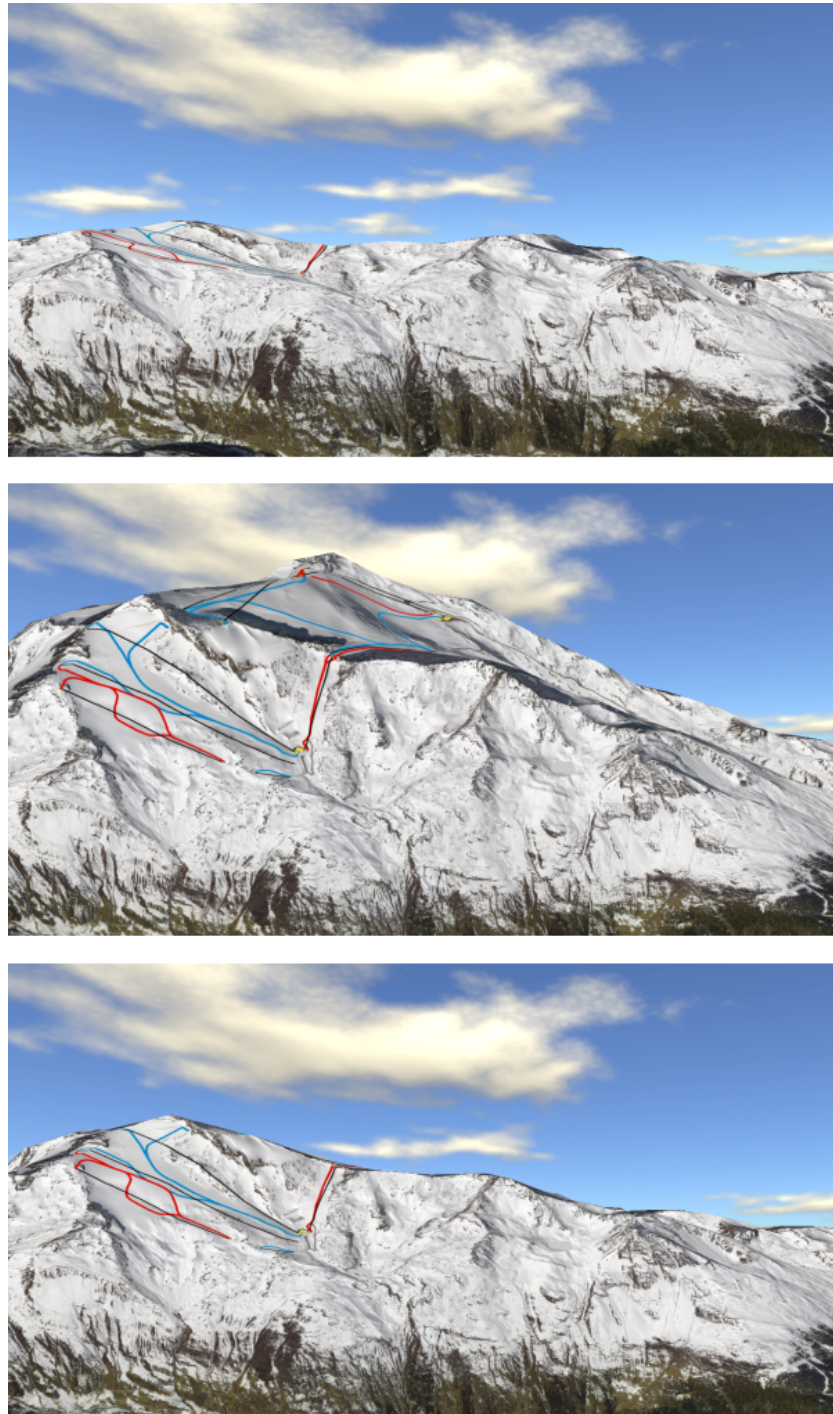


Figure 16.7: The glacier ski areas shown from the east. Top: original view (Rettenbach is back facing). Middle: deformation with back facing features enabled. Bottom: the same panorama if back facing features lying far from silhouettes are disabled.

layers. While the shape of individual triangles is preserved due to the tangential term, the original curvatures at the edges are lost. Without any curvature preservation, there are many local energy minima which correspond to different fold configurations. In the shown example the optimization is stuck in such a local minimum where the occlusion energy is too weak to push the occluding layer further down.

A limitation of our method is demonstrated in Figure 16.9 where a virtual road running all around the base of Mount Rainier was painted manually into the feature map. The shown viewpoint and feature combination is very challenging for our method as large parts of the road lie on back facing parts of the surface. Clearly, the chosen viewpoint is very far from being optimal. The panorama therefore exhibits a strong deformation of the original silhouette. A more suitable choice for this feature would rather be to look down on it from a higher viewpoint. Interestingly, as demonstrated in the figure, our method assumes exactly this viewpoint by tilting the mountain towards the viewer and scaling its peak. Although this behaviour is in principle not wrong, the visibility of the road can also be achieved by anisotropically scaling the height of the mountain peak (see Figure 16.9 bottom). An artist or cartographer would possibly opt for this latter approach in this particular example to reduce the overall shape deformation induced by tilting. As our initial intention was to devise a mostly automatic design method, the here proposed formulation currently provides no means to enforce such an alternative behaviour. However, we believe that our approach could be customized to implement a similar effect by adding further constraints or additional energy terms. The choice and derivation of appropriate constraints is an interesting avenue for future research.

### 16.5.3 Conclusion

In this part of the thesis we took a variational approach to panoramic map design. We selected appropriate potentials to quantify shape deformation and derived potentials for screen size and occlusion. Combining these measures we obtain a well-defined energy which is minimized to find a deformation that preserves the original local shape while enhancing the visibility of features.

In contrast to previous approaches, our method arranges and modifies geographic features automatically to optimize the visibility of features. It is thus suitable e.g. for fully automatic location dependent map generation on mobile devices. Although the proposed method requires the specification of the relative importance of shape preservation and visibility, these parameters directly correspond to four well-motivated potentials and have therefore an intuitive meaning.

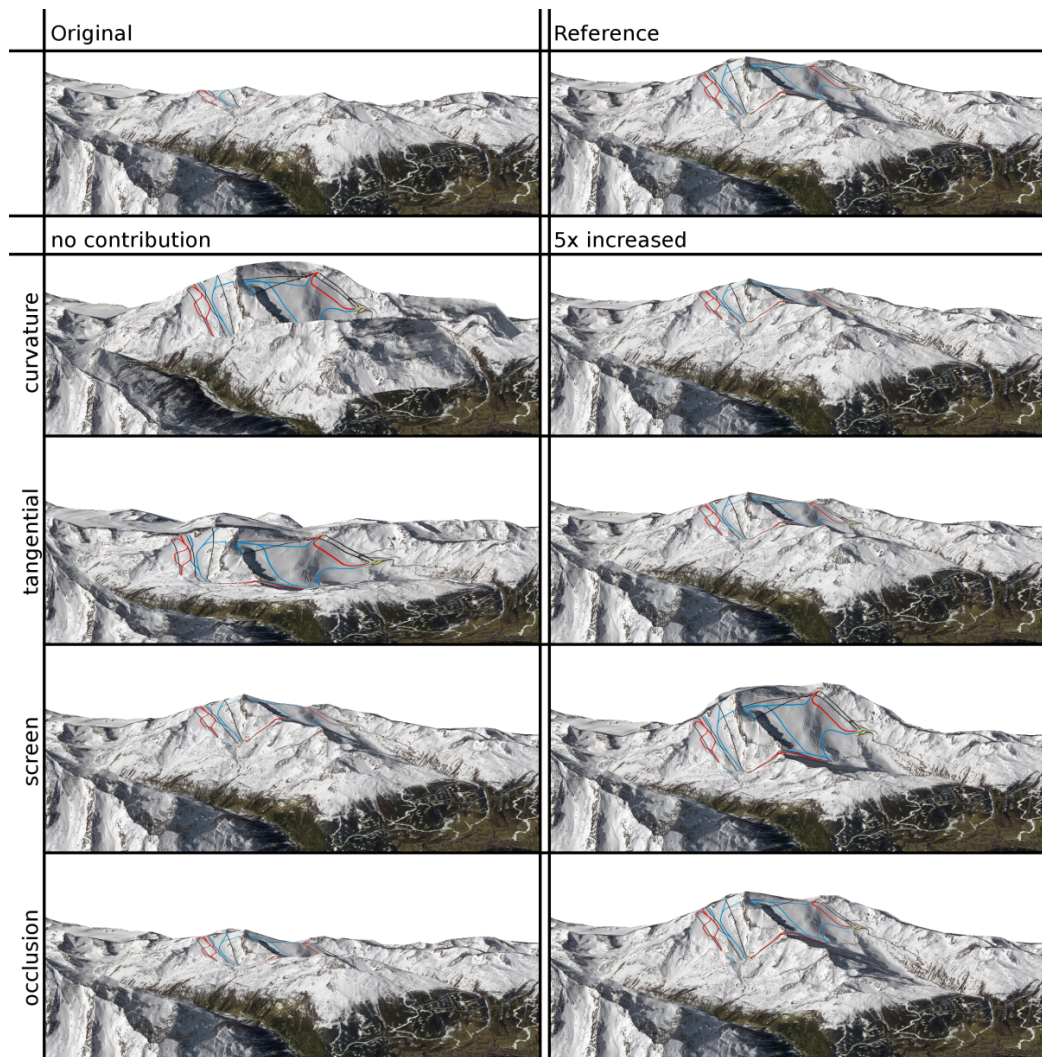


Figure 16.8: Influence of  $\beta$  parameters: the top row shows the original undeformed surface and a reference deformation. The following rows show the result when the respective energy is not used at all or five times increased w.r.t. the reference



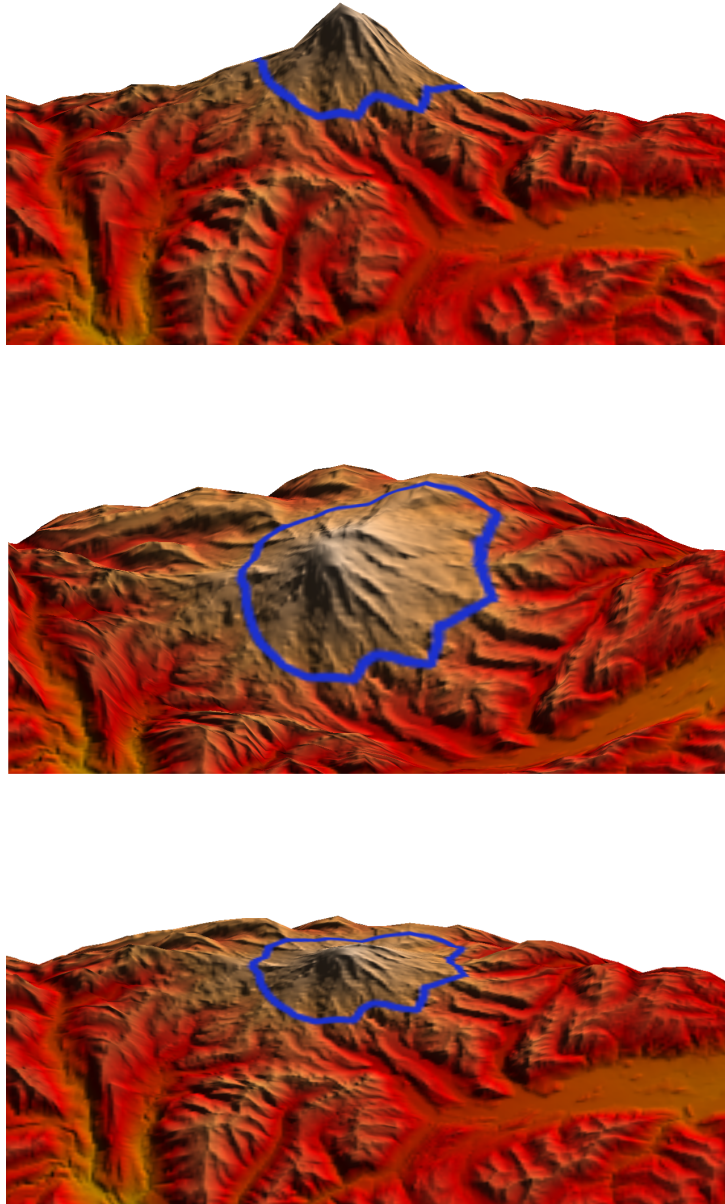


Figure 16.9: A fictive road running around the base of Mount Rainier. The view-point and feature combination is very challenging as large parts of the road are back facing. The original is shown on the top. Our method ensures the visibility of the road by tilting the mountain towards the observer (middle). Alternatively, in this particular example it is also possible to scale the height of the mountain peak in order to disclose the road (bottom).

As lined out in the last section, for some applications it might be desirable to gain more control over the kind of deformation. Even if our primary goal was to devise a method that automates the design of panoramic maps, we think that the proposed formulation can easily be extended to incorporate constraints specified by an artist in an interactive design session. In contrast to standard surface editing methods, such a panorama editor would ideally allow an artist to manipulate details of the projection, while relieving her from the burden of an occlusion-free feature arrangement. We would like to explore this possibility in future work. Another direction of future work is the choice of a suitable viewpoint: Although this choice is clearly subjective, selecting a good viewpoint can significantly reduce the necessary shape deformation and thus improve recognizability. An assisted selection that suggests possible viewpoints not only based on the resulting deformation but also on well known principles of image organisation is thinkable and an highly desirable extension to our method.



**Part V**

**Conclusion and Closure**





---

The concept of deformation is powerful in itself and has been used by humans for ages to shape their environment according to their needs. In this thesis we followed a generalized idea of deformation to shape surfaces for applications in computer graphics. Concentrating on the final, static equilibrium state of deformations, we focused on the potential energy that characterizes these states and proposed generalized potentials that account for application specific requirements and notions of shape preservation.

In summary, this approach was applied to three major application fields: For surface parameterization, the parameterization was understood as a deformation, that unfolds the given, curved surface into the plane. This allowed to compare a large number of existing parameterization methods in a common framework by looking at their corresponding energy potential. Potentials were derived that directly correspond to distortions of geometric shape properties like angles, length or area. In addition, we proposed a projection approach to texture map discrete surface representations with inconsistencies. For a special class of objects — which we termed “fabric covered surfaces” — we introduced material specific texture maps that instead of minimizing geometric shape deformation, reproduce fabric deformation for a given material. Consequently, material specific texture maps are found as extremal points of the anisotropic membrane potential. As an alternative application, the proposed parameterization methods were also applied in the design of sewing patterns for industrial upholstery production.

For interactive shape editing, we introduced a novel differential shape representation for discrete surfaces and gave an algorithm to minimize residuals in this representation at interactive frame rates. Using a handle metaphor this optimization can be used for interactive shape editing with intuitive surface deformations even under large handle transformations. We also showed, that by using a novel kind of constraints, it can also be used in the rapid visualization of upholstered furniture in industrial design.

Panorama map design is not a classical computer graphics application but is up to now still a tedious and artistic manual process. Generalized surface deformation naturally lend themselves to automate this task. By selecting appropriate potentials for shape preservation and by deriving novel visibility potentials, we developed the first automatic panorama generation algorithm. Beyond a substantial reduction of manual effort in the traditional panorama design, we envisage new applications of automatically generated panoramas for personal use and on mobile navigation devices.

While implications of the proposed algorithms and potential for future work has already been discussed in context of the individual applications, we would like to draw some conclusions on the general surface deformation approach here. First, generalized surface deformation can obviously only be applied in situations where a certain “urshape” is present, that has to be preserved. E.g. in case of surface

---

parameterization the urshape is given by the input surface or more precisely by its intrinsic shape while for shape editing it is the local shape of the original, unedited surface. Certainly, the general variational principle of finding shapes as extrema of certain energies can be applied without the notion of an urshape, but it is the existence of an urshape that distinguishes our surface deformation approach from other variational approaches like shape optimization.

As a consequence, potential applications of generalized surface deformations are problems that require a balance between shape preservation and further application dependent constraints. As demonstrated in this thesis, applications can vary vastly in the specific notion of shape that is to be preserved. A popular way to characterize different notions of shape is to consider their invariance with respect to transformations: In general, most people will agree that shape is something that is invariant with respect to translation and rotation. This notion of shape underlies the potentials used for panorama design. For interactive shape editing, we considered *conformal shape*, that is additionally invariant with respect to uniform scaling. And in context of parameterization we considered the *intrinsic shape* of a surface, that is characterized by its invariance with respect to isometries. Notions of shape can easily be more diverse by considering different invariance properties for preimage  $\bar{S}$  and image  $S$ . For example, the elastic membrane potential of the deformation  $\tilde{\mathbf{x}}$  depends only on the metric tensor and is thus invariant with respect to rotations  $\mathbf{R} \circ \tilde{\mathbf{x}}$  of the surface  $S$ . However, as long as the material is not isotropic, the elastic potential is not invariant with respect to a rotation  $\tilde{\mathbf{x}} \circ \mathbf{R}$  of the urshape  $\bar{S}$ . The choice of the appropriate notion of shape is thus highly application specific.

To find an appropriate shape potential, it is tempting to resort to physical models like the thin shell model. As people are accustomed to deformations of elastic shells like leaves or cloth, deformations resulting from physical models are highly intuitive and seem natural. In fact, the elastic potential can be a good starting point and has been suggested both in context of surface parameterization and for interactive shape editing. However, as argued in this thesis, the applicability of physical models is limited: For surface parameterizations the elastic potential leads to gap artifacts as demonstrated in Section 4.2.2 and in context of shape editing it cannot handle transformations like scaling that are physically impossible. Moreover, physical material parameters as the elastic tensor have a physical interpretation. They are therefore easily specified if the deformation corresponds to an actual physical process but their choice can be far less obvious in the case of generalized deformations as considered in this thesis. Summing up, we conclude that physical models can be beneficial if an intuitive behavior is desired, closely corresponding to an actual physical deformation. In case of generalized deformation, the limitations inherited from physical deformations and the choice of parameters often renders them less suitable.

---

We would also like to remark, that the correct choice of a shape potential becomes more important if larger deformations are expected. Certainly a shape potential should attain its minimum for  $\tilde{\mathbf{x}} = \mathbf{id}$  and many potentials are nearly identical if the deformation is small. It can thus make sense to choose the shape potential depending on the expected amount of deformation. For example, for surfaces that are nearly developable, minimizing quadratic conformal potential yields approximately the same result than an optimization of the non-linear combined potential proposed in Section 5.4, but at the same time can be optimized much faster. The difference between the two potentials becomes noticeable only on surfaces that are more “difficult” to flatten.

Apart from shape preservation, further constraints are obviously also application specific and can be specified as hard constraints to the optimization, e.g. the handle positions in interactive shape editing or in surface parameterization where the preimage of the parameterization must be planar by construction. Alternatively, they can be specified as soft constraints by adding further potentials as e.g. the visibility potentials for panorama design. In this latter case, the generalized deformation approach becomes very similar to a shape optimization where the shape preservation acts as a regularization.

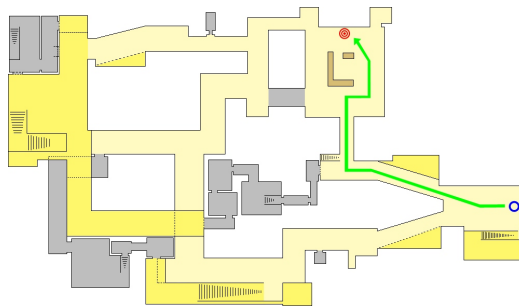
We concentrated in this thesis on surfaces represented by triangle meshes as this is the predominant surface representation in computer graphics. While some of the developed methods as e.g. the quaternion based representation are explicitly tailored to this representation other concepts can be easily generalized to smooth regular surfaces and other surface representations. In this way it is e.g. easily possible to generalize the surface parameterization potentials and algorithms to spline or subdivision surfaces. The same is true for the potentials discussed in context of panorama design with the notable exception of the bending energy whose discretization is based on a discrete differential geometry approach.

Beyond the three discussed applications, generalized surface deformation have already been applied to a number of other problems in computer graphics as morphing, shape matching, surface reconstruction or symmetry detection (see Section 1.1). We realize that with a growing understanding of shape potentials and their properties, their use becomes more and more popular. We think that, still, the full potential of the generalized deformation approach has not been harvested.

Furthermore, we see much potential in the concept of visibility driven deformations introduced in Part IV. To demonstrate the applicability of visibility driven deformations we would like to point to some follow up work: In [DSSK08] we proposed a novel path visualization based on this very concept. Given a geometric scene description and a path, this visualization conveys the whole path in a single static image as shown in Figure 16.10. Occlusions are suppressed by a space deformation so that the path remains visible even if it takes several turns. At the



(a)



(b)



(c)

Figure 16.10: A result of the path visualization presented in [DSSK08]: (a) the static image showing the hole path from its start to the destination marked by a green arrow. Signs indicate turns. (b) a traditional street map of the same path. In contrast to the visualization (a), features are abstract and orientation and self location requires considerable mental effort. (c) a screen shot of the interactive route visualization system presented by Möser et al. [MDWK08].

---

same time the visualization shows realistic depictions of features that can be easily mapped to the observer's environment. Self location and orientation thus become much easier compared to a traditional street map. In [MDWK08], we present an interactive route visualization that targets at mobile navigation devices. Surface deformations are used to increase the visibility of the upcoming route, facades of buildings and to enlarge areas of interest on the screen. An example screen shot of the system is shown in Figure 16.10(c). We envisage a range of novel visualizations in the same spirit that actively enhance visibility and suppress occlusion while hiding unavoidable shape deformation. With the increasing availability of geographic data, we see many potential applications for such visualizations in consumer geographic information system like Google Earth. But even beyond geography one easily finds application for such visualizations in augmented reality systems or for mechanical or medical illustrations.

---

## LIST OF PUBLICATION

---

- Christopher Schwartz, Ruwen Schnabel, Patrick Degener, and Reinhard Klein. Photopath: Single image path depictions from multiple photographs. To appear in Proceedings of the 18th International Conference on Computer Graphics, Visualization and Computer Vision 2010 (WSCG 2010), February 2010.
- Ruwen Schnabel, Patrick Degener, and Reinhard Klein. Completion and reconstruction with primitive shapes. *Computer Graphics Forum (Proc. of Eurographics)*, 28(2):503–512, March 2009.
- Christopher Schwartz, Patrick Degener, and Reinhard Klein. Interactive editing of upholstered furniture. In *International Conference on Computer Graphics, Visualization and Computer Vision (WSCG '09)*, February 2009.
- Patrick Degener and Reinhard Klein. A variational approach for automatic generation of panoramic maps. *ACM Transactions on Graphics*, 28(1):1–14, January 2009.
- Sebastian Möser, Patrick Degener, Roland Wahl, and Reinhard Klein. Context aware terrain visualization for wayfinding and navigation. *Computer Graphics Forum*, 27(7):1853–1860, October 2008.
- Patrick Degener, Ruwen Schnabel, Christopher Schwartz, and Reinhard Klein. Effective visualization of short routes. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1452–1458, October 2008.
- Markus Schlattmann, Patrick Degener, and Reinhard Klein. Scale space based feature point detection on surfaces. *Journal of WSCG*, 16(1-3), February 2008.
- Nikolas Paries, Patrick Degener, and Reinhard Klein. Simple and efficient mesh editing with consistent local frames. In *PG '07: Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, pages 461–464, Washington, DC, USA, 2007. IEEE Computer Society.

- Patrick Degener and Reinhard Klein. Texture atlas generation for inconsistent meshes and point sets. In *IEEE International Conference on Shape Modeling and Applications 2007 (SMI'07)*, pages 156–168. IEEE Computer Society, June 2007.
- Gerhard H. Bendels, Patrick Degener, Roland Wahl, Marcel Körtgen, and Reinhard Klein. Image-based registration of 3d-range data using feature surface elements. In Y. Chrysanthou, K. Cain, N. Silberman, and Franco Niccolucci, editors, *The 5th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST 2004)*, pages 115–124. Eurographics, December 2004.
- Roland Wahl, Manuel Massing, Patrick Degener, Michael Guthe, and Reinhard Klein. Scalable compression and rendering of textured terrain data. *Journal of WSCG*, 12(3):521–528, February 2004.
- Patrick Degener, Jan Meseth, and Reinhard Klein. An adaptable surface parametrization method. In *The 12th International Meshing Roundtable 2003*, September 2003.



---

## INDEX

---

- (covariant) metric tensor, 12
- ABF++, 49
- adjacency matrix, 19
- adjacent, 19
- alternating optimization, 142
- Angle Based Flattening, 49
  
- bending potential, 31
- bending strain, 30
- bending stress tensor, 31
- bidirectional texture functions, 102
  
- Cauchy-Riemann equations, 47
- conformal, 15
- conformal factor, 15
- conformal shape, 236
- contravariant, 94
- contravariant base vectors, 24
- contravariant metric tensor, 15
- contravariant tensor, 86
- covariant, 94
- covariant base vectors, 24
- cross parameterization, 42
  
- developable surfaces, 39
- differential, 17
- Dirichlet Energy, 46
- discrete authalic energy, 47
- discrete conformal, 46
  
- edge-spring model, 41
- elastic potential energy, 25
  
- first fundamental form, 12
- Fixed seams, 92
- flattening, 83
- floating seams, 92
- force field constraint, 171
  
- Gaussian curvature, 17
- geometric stretch, 52
- global parameterization, 42
- Green-Lagrange strain tensor, 25
  
- harmonic maps, 46
  
- incident, 19
- interactive shape editing, 5
- intrinsic, 15
- intrinsic shape, 236
- inverse parameterization, 40
- isometric, 15, 37
- isotropic elastic materials, 26
  
- Laplacian coordinates, 131
- Least Squares Conformal Maps (LSCM),  
48
- linear methods, 44
  
- manifold, 19
- material specific texture maps, 83
- mean curvature, 17
- membrane potential, 31
- membrane strain, 30
- membrane stress tensor, 31
- Membranes, 27

---

MIPS energy, 48  
Multi Dimensional Scaling (MDS), 51  
non-planar parameterization, 42  
normal stress, 95  
orientable, 12, 20  
orientation constraints, 142  
oriented, 20  
Panorama maps, 6  
parameterization, 11  
pattern, 81  
per face texture coordinates, 110  
Poisson mesh editing, 131  
Poisson's ratio, 27  
polycube, 42  
position constraints, 142  
principal curvature directions, 16  
principal curvatures, 17  
regular, 24  
regular surface, 11  
seam strain constraint, 91  
seam stress constraint, 92  
second fundamental form, 16  
shape operator, 16  
shape optimization, 8  
shape spaces, 8  
shell director, 29  
shells, 27  
smooth, 17  
strain, 24  
strain constraint potential, 93  
stress, 26  
stress constraint potential, 95  
surface gradient, 18  
Surface matching, 7  
surface normal, 12  
surface parameterization, 4  
surface reconstruction, 7  
surface sampling potentials, 51  
tangent plane, 12  
texture coordinates, 40  
thin plates, 28  
thin shell models, 4  
thin spline energy, 129  
translation-insensitivity, 132  
triangle mesh, 18  
vertex relaxation, 70  
visibility driven, 6  
visibility driven deformation, 185  
Young's modulus, 27

---

## BIBLIOGRAPHY

---

- [ACP03] Brett Allen, Brian Curless, and Zoran Popovic. The space of human body shapes: reconstruction and parameterization from range scans. *ACM Trans. Graph.*, 22(3):587–594, 2003.
- [Ale03] Marc Alexa. Differential coordinates for local mesh morphing and deformation. *The Visual Computer*, 19(2-3):105–114, 2003.
- [ANA02] Phillip N. Azariadis, Andreas C. Nearchou, and Nikos A. Aspragathos. An evolutionary algorithm for generating planar developments of arbitrarily curved surfaces. *Computers in Industry*, 47(3):357–368, March 2002.
- [ARV07] Brian Amberg, Sami Romdhani, and Thomas Vetter. Optimal step nonrigid ICP algorithms for surface registration. In *CVPR*. IEEE Computer Society, 2007.
- [AS01] Maneesh Agrawala and Chris Stolte. Rendering effective route maps: improving usability through generalization. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 241–249, New York, NY, USA, 2001. ACM.
- [AS04] P. N. Azariadis and N. S. Sapidis. Planar development of free-form surfaces: quality evaluation and visual inspection. *Computing*, 72(1):13–27, 2004.
- [ASK<sup>+</sup>05] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. *ACM Trans. Graph.*, 24(3):408–416, 2005.
- [ATLF06] Oscar Kin-Chung Au, Chiew-Lan Tai, Ligang Liu, and Hongbo Fu. Dual laplacian editing for meshes. *IEEE Transactions on Visualization and Computer Graphics*, 12(3):386–395, 2006.

- 
- [Aza04] Phillip N. Azariadis. Parameterization of clouds of unorganized points using dynamic base surfaces. *Computer-Aided Design*, 36(7):607–623, 2004.
- [AZM00] Maneesh Agrawala, Denis Zorin, and Tamara Munzner. Artistic multiprojection rendering. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*, pages 125–136, London, UK, 2000. Springer-Verlag.
- [Bar84a] Alan H. Barr. Global and local deformations of solid primitives. *SIGGRAPH Comput. Graph.*, 18(3):21–30, 1984.
- [Bar84b] Alan H. Barr. Global and local deformations of solid primitives. In *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 21–30, New York, NY, USA, 1984. ACM.
- [Bär01] Christian Bär. *Elementare Differentialgeometrie (Monografie)*. de Gruyter Lehrbuch. Walter de Gruyter & Co., Berlin, 2001.
- [BB06] Charles Bloom and Jonathan Blow. Errors and omissions in marc alexa's. linear combination of transformations, [www.cbloom.com/3d/techdocs/lcot\\_errors.pdf](http://www.cbloom.com/3d/techdocs/lcot_errors.pdf), 2006.
- [BBK05] Mario Botsch, David Bommes, and Leif Kobbelt. Efficient linear system solvers for mesh processing. In *Mathematics of Surfaces XI*, volume 3604 of *Lecture Notes in Computer Science*, pages 62–83. Springer Berlin, 2005.
- [BBW<sup>+</sup>09] A. Berner, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. Generalized intrinsic symmetry detection. Technical report, MPI Informatik, 2009.
- [BCGB08] Mirela Ben-Chen, Craig Gotsman, and Guy Bunin. Conformal flattening by curvature prescription and metric scaling. *Comput. Graph. Forum*, 27(2):449–458, 2008.
- [BFGS03] Jeff Bolz, Ian Farmer, Eitan Grinspun, and Peter Schröder. Sparse matrix solvers on the gpu: conjugate gradients and multigrid. *ACM Trans. Graph.*, 22(3):917–924, 2003.
- [BH03] Philip L. Bowers and Monica K. Hurdal. Planar conformal mappings of piecewise flat surfaces. In Hans-Christian Hege and Konrad Polthier, editors, *Visualization and Mathematics III*, pages 3–34. Springer-Verlag, Heidelberg, 2003.

- 
- [BK04] Mario Botsch and Leif Kobbelt. An intuitive framework for real-time freeform modeling. In *ACM SIGGRAPH 2004 Papers*, pages 630–634, Los Angeles, California, 2004. ACM.
- [BK05] Stephan Bischoff and Leif Kobbelt. Structure preserving CAD model repair. *Computer Graphics Forum, Proceedings of Eurographics 2005*, 24(3):527–536, 2005.
- [BKS03] Gerhard H. Bendels, Reinhard Klein, and A. Schilling. Image and 3d-object editing with precisely specified editing regions. In T. Ertl, B. Girod, G. Greiner, H. Niemann, H.-P. Seidel, E. Steinbach, and R. Westermann, editors, *Vision, Modeling and Visualization 2003*, pages 451–460. Akademische Verlagsgesellschaft Aka GmbH, Berlin, November 2003.
- [BM92] Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, 1992.
- [BMBZ02] Henning Biermann, Ioana Martin, Fausto Bernardini, and Denis Zorin. Cut-and-paste editing of multiresolution surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 312–321. ACM Press, 2002.
- [BNG06] Tom Brunet, K. Evan Nowak, and Michael Gleicher. Integrating dynamic deformations into interactive volume visualization. In *EuroVis*, pages 219–226. Eurographics Association, 2006.
- [BNK02] P. Borodin, M. Novotni, and R. Klein. Progressive gap closing for mesh repairing. In *Advances in Modelling, Animation and Rendering*, pages 201–213. July 2002.
- [BPGK06a] Mario Botsch, Mark Pauly, Markus Gross, and Leif Kobbelt. PriMo: coupled prisms for intuitive surface modeling. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 11–20, Cagliari, Sardinia, Italy, 2006. Eurographics Association.
- [BPGK06b] Mario Botsch, Mark Pauly, Markus Gross, and Leif Kobbelt. Primo: Coupled prisms for intuitive surface modeling. In *Eurographics Symp. on Geom. Processing*, pages 11–20, June 2006.
- [BPK05] Stephan Bischoff, Darko Pavic, and Leif Kobbelt. Automatic restoration of polygon models. *ACM Trans. Graph.*, 24(4):1332–1352, 2005.

- 
- [BPR<sup>+</sup>06] Mario Botsch, Mark Pauly, Christian Rossl, Stephan Bischoff, and Leif Kobbelt. Geometric modeling based on triangle meshes. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses*, page 1, New York, NY, USA, 2006. ACM Press.
- [BPWG07] Mario Botsch, Mark Pauly, Martin Wicke, and Markus Gross. Adaptive space deformations based on rigid cells. *Computer Graphics Forum*, 26(3):339–347, 2007.
- [BS86] E. A. Bier and K. R. Sloan, Jr. Two part texture mappings. *IEEE Comp. Graph. and Appl.*, 6(9):40–53, September 1986.
- [BS08] Mario Botsch and Olga Sorkine. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):213–230, 2008.
- [BSPG06] Mario Botsch, Robert Sumner, Mark Pauly, and Markus Gross. Deformation transfer for detail-preserving surface editing. In *Vision, Modeling and Visualization*, pages 357–364, 2006.
- [BSSS07] John Brosz, Faramarz F. Samavati, M. T. Carpendale Sheelagh, and Mario Costa Sousa. Single camera flexible projection. In *NPAR '07: Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*, pages 33–42, New York, NY, USA, 2007. ACM.
- [BTB02] Laurent Balmelli, Gabriel Taubin, and Fausto Bernardini. Space-optimized texture maps. *Computer Graphics Forum*, 21(3), 2002.
- [BVI91] Chakib Bennis, Jean-Marc Vézien, and Gérard Iglésias. Piecewise surface flattening for non-distorted texture mapping. *ACM SIGGRAPH Computer Graphics*, 25(4):237–246, 1991.
- [BW98] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 43–54. ACM, 1998.
- [BZK09] David Bommes, Henrik Zimmer, and Leif Kobbelt. Mixed-integer quadrangulation. *ACM Trans. Graph.*, 28(3), 2009.
- [CB05] Luca Chittaro and Stefano Burigat. Augmenting audio messages with visual directions in mobile guides: an evaluation of three approaches. In *MobileHCI '05: Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, pages 107–114, New York, NY, USA, 2005. ACM.



- 
- [CBGS05] Patrick Coleman, Leon Barrett, Cindy Grimm, and Karan Singh. Sketching 3d scene projections. Technical Report 2005-13, Washington University, 2005.
- [CG91] George Celniker and Dave Gossard. Deformable curve and surface finite-elements for free-form shape design. *SIGGRAPH Comput. Graph.*, 25(4):257–266, 1991.
- [Cia88] Philippe G. Ciarlet. *Mathematical Elasticity Volume I: Three-Dimensional Elasticity*. North-Holland, 1988.
- [Cia05] Philippe G. Ciarlet. *An introduction to differential geometry with applications to elasticity*. 2005.
- [CLR04a] U. Clarenz, N. Litke, and M. Rumpf. Axioms and variational problems in surface parameterization. *Computer Aided Geometric Design*, 21(8):727–749, 2004.
- [CLR04b] Ulrich Clarenz, Nathan Litke, and Martin Rumpf. Axioms and variational problems in surface parameterization. *Comput. Aided Geom. Des.*, 21(8):727–749, 2004.
- [COS00] Fehmi Cirak, Michael Ortiz, and Peter Schröder. Subdivision surfaces: A new paradigm for thin-shell finite-element analysis. *International Journal for Numerical Methods in Engineering*, 47:2039–2072, 2000.
- [CSC07] Carlos Correa, Debora Silver, and Mi Chen. Illustrative deformation for data exploration. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1320–1327, 2007.
- [dAST<sup>+</sup>08] Edilson de Aguiar, Carsten Stoll, Christian Theobalt, Naveed Ahmed, Hans-Peter Seidel, and Sebastian Thrun. Performance capture from sparse multi-view video. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, pages 1–10, New York, NY, USA, 2008. ACM.
- [dC76] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976. 503 pages.
- [dC92] Manfredo Perdigao do Carmo. *Riemannian Geometry*. Birkh user Boston, 1 edition, 1992.

- 
- [Deg03] Patrick Degener. Computing parameterizations of triangulated surfaces with minimal metric deformations. Diplomarbeit, Institut für Informatik II, Universität Bonn, 2003.
- [DG07] Shen Dong and Michael Garland. Iterative methods for improving mesh parameterizations. In *Proceedings of the IEEE International Conference on Shape Modeling and Applications 2007*, pages 199–210. IEEE Computer Society, 2007.
- [DGL97] James W. Demmel, John Gilbert, and Xiaoye S. Li. SuperLU users’ guide. Technical Report CSD-97-944, 8, 1997.
- [DK07] Patrick Degener and Reinhard Klein. Texture atlas generation for inconsistent meshes and point sets. In *IEEE International Conference on Shape Modeling and Applications 2007 (SMI’07)*, pages 156–168. IEEE Computer Society, June 2007.
- [DK09] Patrick Degener and Reinhard Klein. A variational approach for automatic generation of panoramic maps. *ACM Transactions on Graphics*, 28(1):1–14, January 2009.
- [DM97] I. L. Dryden and K. V. Mardia. *Statistical Shape Analysis*. Wiley, Chichester, 1997.
- [DMA02] Mathieu Desbrun, Mark Meyer, and Pierre Alliez. Intrinsic parameterizations of surface meshes. In *Eurographics conference proceedings*, pages 209–218, 2002.
- [DMK03] P. Degener, J. Meseth, and R. Klein. An adaptable surface parameterization method. In *Proceedings of the 12th International Meshing Roundtable*, pages 227–237, 2003.
- [DMSB99] Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 317–324. ACM Press/Addison-Wesley Publishing Co., 1999.
- [DSSK08] Patrick Degener, Ruwen Schnabel, Christopher Schwartz, and Reinhard Klein. Effective visualization of short routes. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1452–1458, October 2008.

- 
- [DTK] Patrick Degener, Woyten Tiesch, and Reinhard Klein. Material specific texture maps. In preparation.
- [Dua01] *Intelligent balloon: a subdivision-based deformable model for surface reconstruction of arbitrary topology*, New York, NY, USA, 2001. ACM.
- [Duc77] Jean Duchon. *Splines minimizing rotation-invariant semi-norms in Sobolev spaces*, pages 85–100. 1977.
- [EDD<sup>+</sup>95] Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Michael Lounsbery, and Werner Stuetzle. Multiresolution analysis of arbitrary meshes. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 173–182. ACM, 1995.
- [EL78] J. Eells and L. Lemaire. *A report on harmonic maps*. Bulletin of the London Mathematical Society, 1978.
- [EL88] J. Eells and L. Lemaire. *Another report on harmonic maps*, volume 20. Citeseer, 1988.
- [EP09] Michael Eigensatz and Mark Pauly. Positional, metric, and curvature control for constraint-based surface deformation. *Comput. Graph. Forum*, 28(2):551–558, 2009.
- [ESP08] Michael Eigensatz, Robert W. Sumner, and Mark Pauly. Curvature-domain shape processing. *Comput. Graph. Forum*, 27(2):241–250, 2008.
- [FAT07] Hongbo Fu, Kin-Chung Oscar Au, and Chiew-Lan Tai. Effective derivation of similarity transformations for implicit laplacian mesh editing. *Computer Graphics Forum*, 26(1):34–45, March 2007.
- [FH05] M. S Floater and K. Hormann. Surface parameterization: a tutorial and survey. *Advances in multiresolution for geometric modelling*, 1, 2005.
- [Flo97] Michael S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(4):231–250, 1997.
- [Flo01] Michael Floater. Convex combination maps. *ALGORITHMS FOR APPROXIMATION IV*, pages 18–23, 2001.

- 
- [Flo03] M. S Floater. Mean value coordinates. *Computer Aided Geometric Design*, 20(1):19–27, 2003.
- [FR01] M. S Floater and M. Reimers. Meshless parameterization and surface reconstruction. *Computer Aided Geometric Design*, 18(2):77–92, 2001.
- [FSWE07] Martin Falk, Tobias Schafhitzel, Daniel Weiskopf, and Thomas Ertl. Panorama maps with non-linear ray tracing. In *GRAPHITE '07: Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, pages 9–16, New York, NY, USA, 2007. ACM.
- [GASP08] Floraine Grabler, Maneesh Agrawala, Robert W. Sumner, and Mark Pauly. Automatic generation of tourist maps. *ACM Trans. Graph.*, 27(3):1–11, 2008.
- [GGH02] Xianfeng Gu, Steven J. Gortler, and Hugues Hoppe. Geometry images. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 355–361. ACM Press, 2002.
- [GG03] C. Gotsman, X. Gu, and A. Sheffer. Fundamentals of spherical parameterization for 3 d meshes. *ACM Transactions on Graphics*, 22(3):358–363, 2003.
- [GHDS03] Eitan Grinspun, Anil N. Hirani, Mathieu Desbrun, and Peter Schröder. Discrete shells. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 62–67, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.
- [GN04] Michael T. Gastner and M. E. J. Newman. Diffusion-based method for producing density equalizing maps. *PROC.NATL.ACAD.SCI.USA*, 101:7499, 2004.
- [GPRJ00] Sarah F. Frisken Gibson, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones. Adaptively sampled distance fields: a general representation of shape for computer graphics. In *Proc. of SIGGRAPH*, pages 249–254, 2000.
- [Grö95] Eduard Gröller. Nonlinear ray tracing: visualizing strange worlds. *The Visual Computer*, 11(5):263–274, 1995.

- 
- [GTLH98] A. Gueziec, G. Taubin, F. Lazarus, and W. Horn. Converting sets of polygons to manifold surfaces by cutting and stitching. In *IEEE Visualization*, pages 383–390, October 1998.
- [GY03] X. Gu and S. T. Yau. Global conformal surface parameterization. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 127–137, 2003.
- [GZ02] A. E. Green and W. Zerna. *Theoretical Elasticity*. Dover Phoenix Editions, 2002.
- [HB00] Donald H. House and David E. Breen, editors. *Cloth modeling and animation*. A. K. Peters, Ltd., 2000.
- [HBS<sup>+</sup>99] Monica K. Hurdal, Philip L. Bowers, Ken Stephenson, De Witt L. Summers, Kelly Rehm, Kirt Schaper, and David A. Rottenberg. Quasi-conformally flat mapping the human cerebellum. In *MICCAI*, pages 279–286, 1999.
- [HC00] Adam Hunter and Jonathan D. Cohen. Uniform frequency images: adding geometry to images to produce space-efficient textures. In *Proceedings of the conference on Visualization '00*, pages 243–250. IEEE Computer Society Press, 2000.
- [HG00] K. Hormann and G. Greiner. MIPS: An efficient global parametrization method. In P.-J. Laurent, P. Sablonnière, and L. L. Schumaker, editors, *Curve and Surface Design: Saint-Malo 1999*, Innovations in Applied Mathematics, pages 153–162. Vanderbilt University Press, Nashville, 2000.
- [HGC99] K. Hormann, G. Greiner, and S. Campagna. Hierarchical parametrization of triangulated surfaces. In B. Girod, H. Niemann, and H.-P. Seidel, editors, *Proceedings of Vision, Modeling, and Visualization 1999*, pages 219–226, Erlangen, Germany, November 1999. infix.
- [HGM02] Günter Hake, Dietmar Grünreich, and Liqiu Meng. *Kartographie*. de Gruyter, Berlin, 8 edition, 2002.
- [HK06] Alexander Hornung and Leif Kobbelt. Robust reconstruction of watertight 3d models from non-uniformly sampled point clouds without normal information. In *SGP '06: Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 41–50, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

- 
- [HLS07] Kai Hormann, Bruno Lévy, and Alla Sheffer. Mesh parameterization: theory and practice. In *ACM SIGGRAPH 2007 courses*, page 1, San Diego, California, 2007. ACM.
- [HM03] J. Haslinger and R. A. E. Makinen. *Introduction to Shape Optimization: Theory, Approximation, and Computation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003.
- [Hop96] Hugues Hoppe. Progressive meshes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108, New York, NY, USA, 1996. ACM.
- [Hor87] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4:629–642, April 1987.
- [Hor01] K. Hormann. *Theory and Applications of Parameterizing Triangulations*. PhD thesis, Department of Computer Science, University of Erlangen, November 2001.
- [HSC02] Adrian Hilton, Jonathan Starck, and Gordon Collins. From 3d shape capture to animated models. *3dprt*, 00:246, 2002.
- [HSL<sup>+</sup>06] Jin Huang, Xiaohan Shi, Xinguo Liu, Kun Zhou, Li-Yi Wei, Shang-Hua Teng, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Subspace gradient domain mesh deformation. In *SIGGRAPH '06*, pages 1126–1134. ACM Press, 2006.
- [HSS<sup>+</sup>09] N. Hasler, C. Stoll, M. Sunkel, B. Rosenhahn, and H.-P. Seidel. A statistical model of human pose and body shape. In P. Durré and M. Stamminger, editors, *Computer Graphics Forum (Proc. Eurographics 2008)*, volume 2, Munich, Germany, March 2009.
- [Ina91] M Inakage. Non-linear perspective projections. In *Proceedings of the IFIP*, pages 203–215, 1991.
- [JKG08] Miao Jin, Junho Kim, Feng Luo 0002, and Xianfeng Gu. Discrete surface ricci flow. *IEEE Trans. Vis. Comput. Graph.*, 14(5):1030–1043, 2008.
- [JKs05] D. Julius, V. Kraevoy, and A. Sheffer s. D-charts: Quasi-developable mesh segmentation. In *Computer Graphics Forum*, volume 24, pages 581–590, 2005.

- 
- [JSW05] T. Ju, S. Schaefer, and J. Warren. Mean value coordinates for closed triangular meshes. *Proceedings of ACM SIGGRAPH 2005*, 24(3):561–566, 2005.
- [Ju04] Tao Ju. Robust repair of polygonal models. *ACM Trans. Graph.*, 23(3):888–895, 2004.
- [KCVS98] Leif Kobbelt, Swen Campagna, Jens Vorsatz, and Hans-Peter Seidel. Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 105–114. ACM, 1998.
- [KHYS02] Kolja Kaehler, Joerg Haber, Hitoshi Yamauchi, and Hans-Peter Seidel. Head shop: generating animated head models with anatomical structure. In *Proc. of SIGGRAPH/Eurographics symp. on Comp. anim.*, pages 55–63, 2002.
- [KL96] Venkat Krishnamurthy and Marc Levoy. Fitting smooth surfaces to dense polygon meshes. *Computer Graphics*, 30(Annual Conference Series):313–324, 1996.
- [KLP03] A. Khodakovsky, N. Litke, and P. Schröder. Globally smooth parameterizations with low distortion. *ACM Transactions on Graphics*, 22(3):350–357, 2003.
- [KNP07] Felix Kälberer, Matthias Nieser, and Konrad Polthier. Quadcover - surface parameterization using branched coverings. *Computer Graphics Forum*, 26(3):375–384, September 2007.
- [KS04] Vladislav Kraevoy and Alla Sheffer. Cross-parameterization and compatible remeshing of 3D models. In *ACM SIGGRAPH 2004 Papers*, pages 861–869, Los Angeles, California, 2004. ACM.
- [KS05] Vladislav Kraevoy and Alla Sheffer. Template-based mesh completion. In *Symp. on Geom. Processing*, pages 13–22, 2005.
- [KSS06] L. Kharevych, B. Springborn, and P. Schröder. Discrete conformal mappings via circle patterns. *ACM Transactions on Graphics (TOG)*, 25(2):412–438, 2006.
- [LCOGL07] Yaron Lipman, Daniel Cohen-Or, Ran Gal, and David Levin. Volume and shape preservation via moving frame manipulation. *ACM Trans. Graph.*, 26(1):5, 2007.

- 
- [LDRS05] Nathan Litke, Marc Droske, Martin Rumpf, and Peter Schröder. An image processing approach to surface matching. In *SGP '05: Proceedings of the third Eurographics symposium on Geometry processing*, page 207, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.
- [LE06] G. Loy and J. O. Eklundh. Detecting symmetry and symmetric constellations of features. In *ECCV*, pages II: 508–521, 2006.
- [Lev98] Jonathan Levene. A framework for non-realistic projections. *Massachusetts Institute of Technology*, May 1998.
- [LG96] Helwig Löffelmann and Eduard Gröller. Ray tracing with extended cameras. *Journal of Visualization and Computer Animation*, 7(4):211–227, 1996.
- [LGM07] H. Lensch, M. Gösele, and Gero Müller. Capturing reflectance - from theory to practice. Tutorial Notes of Eurographics, September 2007.
- [Lie03] Peter Liepa. Filling holes in meshes. In *Symposium on Geometry Processing*, pages 200–206, 2003.
- [LPRM02a] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics*, 21(3):362–371, 2002.
- [LPRM02b] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérôme Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21(3):362–371, 2002.
- [LPRM02c] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérôme Maillot. Least Squares Conformal Maps for Automatic Texture Atlas Generation. In ACM, editor, *Special Interest Group on Computer Graphics - SIGGRAPH'02, San-Antonio, Texas, USA*, July 2002.
- [LRS<sup>+</sup>09] H. Lensch, B. Rosenhahn, H.-P. Seidel, C. Theobalt, T. Thormählen, and M. Wand. Dagm 2009 tutorial: Visual 3d scene analysis and synthesis. Tutorial Notes, September 2009.
- [LSCO<sup>+</sup>04] Yaron Lipman, Olga Sorkine, Daniel Cohen-Or, David Levin, Christian Rössl, and Hans-Peter Seidel. Differential coordinates for interactive mesh editing. In *Proceedings of Shape Modeling International*, pages 181–190. IEEE Computer Society Press, 2004.



- 
- [LSLCO05] Yaron Lipman, Olga Sorkine, David Levin, and Daniel Cohen-Or. Linear rotation-invariant coordinates for meshes. In *SIGGRAPH '05*, pages 479–487. ACM Press, 2005.
- [LSS<sup>+</sup>98] A. W.F Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin. MAPS: multiresolution adaptive parameterization of surfaces. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 95–104, 1998.
- [LZX<sup>+</sup>08] Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J. Gortler. A Local/Global approach to mesh parameterization. *Computer Graphics Forum*, 27(5):1495–1504, 2008.
- [Mac95] Alan M. MacEachren. *How Maps Work: Representation, Visualization and Design*. Guilford Press, New York, 1995.
- [MDWK08] Sebastian Möser, Patrick Degener, Roland Wahl, and Reinhard Klein. Context aware terrain visualization for wayfinding and navigation. *Computer Graphics Forum*, 27(7):1853–1860, October 2008.
- [MGP07] Niloy J. Mitra, Leonidas J. Guibas, and Mark Pauly. Symmetrization. *ACM Trans. Graph*, 26(3):63, 2007.
- [MGR00] Stephen R. Marschner, Brian K. Guenter, and Sashi Raghupathy. Modeling and rendering for realistic facial animation. In *Rendering Techniques*, pages 231–242. Springer, 2000.
- [MHC05] J. McCartney, B.K. Hinds, and K.W. Chong. Pattern flattening for orthotropic materials. *Computer-Aided Design*, 37(6):631–644, May 2005.
- [MHS99] J. McCartney, B. K. Hinds, and B. L. Seow. The flattening of triangulated surfaces incorporating darts and gussets. *Computer-Aided Design*, 31(4):249–260, April 1999.
- [MM98] R. Mencl and H. Möller. Interpolation and approximation of surfaces from three-dimensional scattered data points. In H.-P. Seidel and S. Coquillart, editors, *STAR Proceedings of Eurographics 1998*, Lisbon, Portugal, September 1998. Eurographics Association.
- [MPS05] Chunhui Mei, Voicu Popescu, and Elisha Sacks. The occlusion camera. *Comput. Graph. Forum*, 24(3):335–342, 2005.

- 
- [MT00] T. McInerney and D. Terzopoulos. T-snakes: Topology adaptive snakes. *Medical Image Analysis*, 4:73–91, 2000.
- [MYV93] J. Maillot, H. Yahia, and A. Verroust. Interactive texture mapping. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 27–34, 1993.
- [NMK<sup>+</sup>06] A. Nealen, M. Muller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. In *Computer Graphics Forum*, volume 25, pages 809–836, 2006.
- [NT03] Fakir S. Nooruddin and Greg Turk. Simplification and repair of polygonal models using volumetric techniques. *IEEE Trans. on Vis. and Comp. Graphics*, 9(2):191–205, 2003.
- [Par07] Nikolas Paries. Editing of surfaces based on differential shape representations. Master’s thesis, Institut für Informatik II, Universität Bonn, 2007.
- [Pat00] Tom Patterson. A view from on high: Heinrich berann’s panoramas and landscape visualization techniques for the u.s. national park service. In *High Mountain Cartography*, 2000.
- [PDK07] Nikolas Paries, Patrick Degener, and Reinhard Klein. Simple and efficient mesh editing with consistent local frames. In *PG ’07: Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, pages 461–464, Washington, DC, USA, 2007. IEEE Computer Society.
- [PJS06] Tiberiu Popa, Dan Julius, and Alla Sheffer. Material-aware mesh deformations. In *SMI ’06: Proceedings of the IEEE International Conference on Shape Modeling and Applications 2006 (SMI’06)*, page 22, Washington, DC, USA, 2006. IEEE Computer Society.
- [PKKG03] Mark Pauly, Richard Keiser, Leif P. Kobbelt, and Markus Gross. Shape modeling with point-sampled geometry. *ACM Trans. Graph.*, 22(3):641–650, 2003.
- [PMW<sup>+</sup>08] Mark Pauly, Niloy J. Mitra, Johannes Wallner, Helmut Pottmann, and Leonidas J. Guibas. Discovering structural regularity in 3D geometry. *ACM Trans. Graph*, 27(3), 2008.
- [PP93] Ulrich Pinkall and Konrad Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

- 
- [Pre02] Simon Premoze. Computer generated panorama maps. In *Proceedings of the ICA Mountain Cartography Workshop*, 2002.
- [PSG<sup>+</sup>06] Joshua Podolak, Philip Shilane, Aleksey Golovinskiy, Szymon Rusinkiewicz, and Thomas A. Funkhouser. A planar-reflective symmetry transform for 3D shapes. *ACM Trans. Graph*, 25(3):549–559, 2006.
- [PT01] L.A. Piegl and W. Tiller. Parameterization for surface fitting in reverse engineering. *Comp.-Aided Design*, 33(8):593–603, 2001.
- [RLL<sup>+</sup>06] N. Ray, W. C Li, B. Lévy, A. Sheffer, and P. Alliez. Periodic global parameterization. *ACM Transactions on Graphics (TOG)*, 25(4):1460–1485, 2006.
- [SA07] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *SGP '07: Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 109–116, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.
- [SAPH04] John Schreiner, Arul Asirvatham, Emil Praun, and Hugues Hoppe. Inter-surface mapping. In *ACM SIGGRAPH 2004 Papers*, pages 870–877, Los Angeles, California, 2004. ACM.
- [SB09] Olga Sorkine and Mario Botsch. Interactive Shape Modeling and Deformation. pages 11–37, Munich, Germany, 2009. Eurographics Association.
- [SCL<sup>+</sup>04] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 175–184, Nice, France, 2004. ACM.
- [SCOGL02] Olga Sorkine, Daniel Cohen-Or, Rony Goldenthal, and Dani Lischinski. Bounded-distortion piecewise mesh parameterization. In *Proceedings of the conference on Visualization '02*, pages 355–362. IEEE Press, 2002.
- [SDK09] Christopher Schwartz, Patrick Degener, and Reinhard Klein. Interactive editing of upholstered furniture. In *International Conference on Computer Graphics, Visualization and Computer Vision (WSCG '09)*, February 2009.

- 
- [SdS01] Alla Sheffer and Eric de Sturler. Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers*, 17(3):326–337, 2001.
- [SdS02] A. Sheffer and E. de Sturler. Smoothing an overlay grid to minimize linear distortion in texture mapping. *ACM Transactions on Graphics (TOG)*, 21(4):874–890, 2002.
- [Ses00] Monika Sester. Generalization based on least squares adjustment. In *In: International Archives of Photogrammetry and Remote Sensing*, pages 931–938, 2000.
- [SGS05] Nisha Sudarsanam, Cindy Grimm, and Karan Singh. Interactive manipulation of projections with a curved perspective. In *Eurographics short papers*, volume 24 of *Computer Graphics Forum*, pages 105–108. Eurographics, September 2005.
- [SGSH02] Pedro V. Sander, Steven J. Gortler, John Snyder, and Hugues Hoppe. Signal-specialized parametrization. In *Proceedings of the 13th workshop on Rendering*, pages 87–98. Eurographics Association, 2002.
- [SH02] A. Sheffer and J. C. Hart. Seamster: inconspicuous low-distortion texture seam layout. *IEEE Visualization, 2002. VIS 2002*, pages 291–298, 2002.
- [She96] Jonathan Richard Shewchuk. Triangle: Engineering a 2D quality mesh generator and delaunay triangulator. In *WACG*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. Springer, 1996.
- [Sho85] Ken Shoemake. Animating rotation with quaternion curves. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, New York, NY, USA, 1985. ACM.
- [Sin02] Karan Singh. A Fresh Perspective. In *GI2002*, pages 17–24, 2002.
- [SK04] Alla Sheffer and Vladislav Kraevoy. Pyramid coordinates for morphing and deformation. In *3DPVT '04*, pages 68–75. IEEE Computer Society, 2004.
- [SLCO<sup>+</sup>04] Olga Sorkine, Yaron Lipman, Daniel Cohen-Or, Marc Alexa, Christian Rössl, and Hans-Peter Seidel. Laplacian surface editing. In

---

*Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 179–188. ACM Press, 2004.

- [SLMB04] A. Sheffer, B. Lévy, M. Mogilnitsky, and A. Bogomyakov. ABF++: fast and robust angle based flattening. 2004.
- [SLS<sup>+</sup>06] Andrei Sharf, Thomas Lewiner, Ariel Shamir, Leif Kobbelt, and Daniel Cohen-Or. Competing fronts for coarse-to-fine surface reconstruction. *Comput. Graph. Forum*, 25(3):389–398, 2006.
- [SOS04] Chen Shen, James F. O’Brien, and Jonathan R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. *ACM Trans. Graph.*, 23(3):896–904, 2004.
- [SP86] Thomas W. Sederberg and Scott R. Parry. Free-form deformation of solid geometric models. In *SIGGRAPH ’86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 151–160, New York, NY, USA, 1986. ACM.
- [SP04] Robert W. Sumner and Jovan Popovic;. Deformation transfer for triangle meshes. *ACM Trans. Graph.*, 23(3):399–405, 2004.
- [SPR06] A. Sheffer, E. Praun, and K. Rose. Mesh parameterization methods and their applications. *Foundations and Trends in Computer Graphics and Vision*, 2(2):105–171, 2006.
- [SS01] A. Sheffer and E. De Sturler. Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers*, 17(3):326–337, 2001.
- [SSGH01a] P. V Sander, J. Snyder, S. J Gortler, and H. Hoppe. Texture mapping progressive meshes. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 409–416, 2001.
- [SSGH01b] Pedro V. Sander, John Snyder, Steven J. Gortler, and Hugues Hoppe. Texture mapping progressive meshes. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 409–416. ACM Press, 2001.
- [SSP08] Boris Springborn, Peter Schröder, and Ulrich Pinkall. Conformal equivalence of triangle meshes. *ACM Trans. Graph.*, 27(3):1–11, 2008.

- 
- [SWB98] Peter-Pike J. Sloan, David M. Weinstein, and J. Dean Breder-son. Importance driven texture coordinate optimization. *Computer Graphics Forum*, 17(3):??-??, 1998.
- [SZR<sup>+</sup>06] Carsten Stoll, Karni Zachi, Christian RÅssel, Hitoshi Yamauchi, and Hans-Peter Seidel. Template deformation for point cloud fitting. *Symposium on Point-Based Graphics*, pages 27–35, 2006.
- [SZT<sup>+</sup>07] Xiaohan Shi, Kun Zhou, Yiyong Tong, Mathieu Desbrun, Hujun Bao, and Baining Guo. Mesh puppetry: cascading optimization of mesh deformation with inverse kinematics. *ACM Trans. Graph.*, 26(3):81, 2007.
- [TACSD06] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun. Designing quadrangulations with discrete harmonic forms. In *SGP '06: Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 201–210, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.
- [TGG05] G. Tewari, C. Gotsman, and S. J. Gortler. Meshing genus-1 point clouds using discrete one-forms. Technical report, Harvard University, June 2005.
- [THCM04] M. Tarini, K. Hormann, P. Cignoni, and C. Montani. Polycube-maps. In *International Conference on Computer Graphics and Interactive Techniques*, pages 853–860, 2004.
- [Thu85] W. P. Thurston. The finite riemann mapping theorem. nvited talk at the symposium on the occasion of the proof of the Bieberbach conjecture held at Purdue University, March 1985.
- [Tol03] S. Toledo. Taucs: A library of sparse linear solvers, version 2.2, <http://www.tau.ac.il/stoledo/taucs>, 2003.
- [TON<sup>+</sup>02] Shigeo Takahashi, Naoya Ohta, Hiroko Nakamura, Yuriko Takeshima, and Issei Fujishiro. Modeling surperspective projection of landscapes for geographical guide-map generation. *Comput. Graph. Forum*, 21(3), 2002.
- [TPBF87] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 205–214, New York, NY, USA, 1987. ACM Press.

- 
- [Tut60] W. Tutte. Convex representation of graphs. In *Proc. London Math. Soc.*, volume 10, 1960.
- [Tut63] W. Tutte. How to draw a graph. In *Proc. London Math. Soc.*, volume 13, pages 743–768, 1963.
- [TYSN06] Shigeo Takahashi, Kenichi Yoshida, Kenji Shimada, and Tomoyuki Nishita. Occlusion-free animation of driving routes for car navigation systems. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1141–1148, 2006.
- [VCM05] Pascal Volino, Frederic Cordier, and Nadia Magnenat-Thalmann. From early virtual garment simulation to interactive fashion design. *Computer-Aided Design*, 37(6):593–608, May 2005.
- [Wan08] Charlie C.L. Wang. WireWarping: a fast surface flattening approach with length-preserved feature curves. *Computer-Aided Design*, 40(3):381–395, March 2008.
- [WAO<sup>+</sup>09] Michael Wand, Bart Adams, Maksim Ovsjanikov, Alexander Berner, Martin Bokeloh, Philipp Jenke, Leonidas Guibas, Hans-Peter Seidel, and Andreas Schilling. Efficient reconstruction of non-rigid shape and motion from real-time 3d scanner data. *ACM Trans. Graph.*, 28(2):1–15, 2009.
- [WAY03] Z. Wu, C.K. Au, and Matthew Yuen. Mechanical properties of fabric materials for draping simulation. *International Journal of Clothing Science and Technology*, 15:56 – 68, 2003.
- [Wei] Gilbert Weinstein. Conformal geometry seminar the poincaré uniformization theorem.
- [WK95] Ma Weiyin and J. P. Kruth. Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces. *Comp.-aided Design*, 27(9):663–675, 1995.
- [WSY02] Charlie C. L. Wang, Shana S-F. Smith, and Matthew M. F. Yuen. Surface flattening based on energy model. *Computer-Aided Design*, 34(11):823–833, September 2002.
- [WTY05] Charlie C.L. Wang, Kai Tang, and Benjamin M.L. Yeung. Freeform surface flattening based on fitting a woven mesh model. *Computer-Aided Design*, 37(8):799–814, July 2005.

- 
- [WW92] William Welch and Andrew Witkin. Variational surface modeling. *SIGGRAPH Comput. Graph.*, 26(2):157–166, 1992.
- [wwwa] www.berann.com. The world of h.c. berann: <http://www.berann.com>.
- [wwwb] www.soelden.com. Ski map of the soelden ski resort: [http://ext.soelden.com/skimap\\_popup/images/skimap.jpg](http://ext.soelden.com/skimap_popup/images/skimap.jpg).
- [wwwc] www.vielkind.at. Homepage of the panorama studio vielkind: <http://www.vielkind.at>.
- [XZY<sup>+</sup>07] Weiwei Xu, Kun Zhou, Yizhou Yu, Qifeng Tan, Qunsheng Peng, and Baining Guo. Gradient domain editing of deforming mesh sequences. *ACM Trans. Graph.*, 26(3):84, 2007.
- [YCB05] Yonggao Yang, Jim X. Chen, and Mohsen Beheshti. Nonlinear perspective projections and magic lenses: 3d view deformation. *IEEE Comput. Graph. Appl.*, 25(1):76–84, 2005.
- [YKL<sup>+</sup>08] Yong-Liang Yang, Junho Kim, Feng Luo, Shi-Min Hu, and Xianfeng Gu. Optimal surface parameterization using inverse curvature map. *IEEE Transactions on Visualization and Computer Graphics*, 14(5):1054–1066, 2008.
- [YM04] Jingyi Yu and Leonard McMillan. General linear cameras. In *ECCV (2)*, pages 14–27, 2004.
- [YZS07] Yunchu Yang, Weiyuan Zhang, and Cong Shan. Investigating the development of digital patterns for customized apparel. *International Journal of Clothing Science and Technology*, 19(3/4):167 – 177, 2007.
- [YZX<sup>+</sup>04] Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Mesh editing with poisson-based gradient field manipulation. In *SIGGRAPH 2004*, pages 644–651, New York, NY, USA, 2004. ACM Press.
- [Zan69] Z. Zangwill. *Nonlinear Programming: A Unified Approach*. Prentice Hall, 1969.
- [ZG04] M. Zwicker and C. Gotsman. Meshing point clouds using spherical parameterization. In *Proc. Eurographics Symp. Point-Based Graphics*, pages 173–180, 2004.



- 
- [ZGVdF97] Ruben Zonenschein, Jonas Gomes, Luiz Velho, and Luiz Henrique de Figueiredo. Texturing implicit surfaces with particle systems. In *Proc. of SIGGRAPH*, page 172, 1997.
- [ZKK01] G. Zigelman, R. Kimmel, and N. Kiryati. Texture mapping using surface flattening via multi-dimensional scaling. *IEEE Transactions on Visualization and Computer Graphics*, 2001.
- [ZPKG02] Matthias Zwicker, Mark Pauly, Oliver Knoll, and Markus H. Gross. Pointshop 3D: an interactive system for point-based surface editing. *ACM Trans. Graph*, 21(3):322–329, 2002.
- [ZRKS05] Rhaleb Zayer, Christian Rössl, Zachi Karni, and Hans-Peter Seidel. Harmonic guidance for surface deformation. In Marc Alexa and Joe Marks, editors, *EUROGRAPHICS 2005*, volume 24 of *Computer Graphics Forum*, pages 601–609, Dublin, Ireland, 2005. Eurographics, Blackwell.
- [ZSC<sup>+</sup>08] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. van Kaick, and A. Tagliasacchi. Deformation-Driven shape correspondence. *Computer Graphics Forum*, 27(5):1431–1439, 2008.
- [ZX06] Yueqi Zhong and Bugao Xu. A physically based method for triangulated surface flattening. *Computer-Aided Design*, 38(10):1062–1073, October 2006.

---

## CURRICULUM VITAE

---

### Personal Data

PATRICK DEGENER  
Rheinische-Friedrich-Wilhelms-Universität Bonn  
Institut für Informatik II - Computergrafik  
Römerstrasse 164, 53117 Bonn, Deutschland

- |                   |   |
|-------------------|---|
| 2003 - 2009       | Universität Bonn,<br>Wissenschaftlicher Mitarbeiter am<br>Institut für Informatik           |
| 1996 - 2003       | Universität Bonn<br>Studium der Informatik und Mathematik,<br>Diplom (Mathematik Vordiplom) |
| 30. Mai 1996      | Dietrich-Bonhoeffer-Gymnasium, Wiehl<br>Abitur  |
| 16. November 1975 | Geboren in Köln   |