Parameterization and Tiling of Polyhedral Surfaces

Dissertationsschrift
zur Erlangung des Grades
eines Doktors der Naturwissenschaften

am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

vorgelegt von
Dipl.-Math. Techn. Matthias Nieser

Submitted 13th October 2011
Defended 23rd February 2012

Referees:

Prof. Dr. Konrad Polthier, Freie Universität Berlin, Germany

Prof. Dr.-Ing. Kai Hormann, University of Lugano, Switzerland

Prof. Eugene Zhang, Oregon State University, United States of America

Contents

Acknowledgements

I foremost thank my advisor, Prof. Dr. Konrad Polthier, for his continuous support of my research as well as for providing and being part of a uniquely conducive environment that was an important precondition for all my recent years' efforts spent on thoroughly exploring the thrilling field of 3D geometry processing.

I am grateful to my friend and colleague Felix Kälberer for our successful collaboration on the development of the QuadCover algorithm and for many fruitful discussions. I also thank my colleagues Ulrich Reitebuch and Klaus Hildebrandt for their help and for many stimulating discussions, as well as Jonathan Palacios and Prof. Eugene Zhang for our very productive collaboration on hexagonal parameterizations.

I thank my parents, the rest of my family and my girlfriend who repeatedly encouraged me to continue and finish my work. A special thank goes to my newborn child who waited patiently until the thesis was concluded.

My research was supported by the DFG Research Center Matheon "Mathematics for key technologies" in Berlin.

Most of the geometry models used in this thesis are courtesy of the Aim@Shape database. Some texture patterns from Fig. 1.2 and the images in Fig. 1.3 (middle and right) and 4.2 are taken from the Wikipedia.

Verfassererklärung

Gemäß §7 (4) der Promotionsordnung versichere ich hiermit, diese Arbeit selbständig verfasst zu haben. Ich habe alle bei der Erstellung dieser Arbeit benutzten Hilfsmittel und Hilfen angegeben.

Matthias Nieser

Chapter 1

Introduction

Polyhedral surfaces are probably the most common representation for surfaces in computer graphics. Because of their simplicity, they can be processed very efficiently in modern computers and graphics cards.

The field of discrete differential geometry investigates polyhedral surfaces from a mathematical point of view. The aim is to provide proper discrete analogs of notions and methods of the smooth differential setting, and to establish a corresponding theory. The development of a consistent theory is challenging, since arbitrary polyhedral surface meshes often lack regularity and a high-level semantic structure. In particular for triangle meshes, important surface features are often not emphasized and triangles are not adapted to the overall shape.

Parameterization. Surface parameterization is the process of mapping a surface into the two-dimensional plane. Conceptually, the surface is considered as a distorted Euclidean plane where the parameterization describes the relation, the deformation and the differences to the plane. The parameterization equips the surface with additional structure, such as providing a metric and consistent local coordinates in all tangent spaces.

The task of generating a parameterization of high quality is challenging. Depending on the actual application, the desired properties are quite different. However, for almost all purposes, it is important to achieve low metric distortion, since then the real surface is approximated well by its image in the flat plane. Applications range from texture mapping in computer graphics and texture synthesis to remeshing and automatic construction of hierarchical subdivision surfaces. All of these applications benefit from the additional structure of a parameterization.

This thesis concerns the mathematical background and theory of *global parameterizations*. The theory is based on the concepts and formalism of discrete differential geometry. The presented approach is driven by conceptual clarity and establishes a consistent analogue to the smooth setting in differential geometry. Moreover, an algorithm called QuadCover is formulated which provides an automatism for generating a parameterization of any given triangular surface.

Tiling. Parameterization is closely related to *surface tiling* which is an analogue to *wallpaper tiling* for surfaces. A wallpaper tiling covers the two-dimensional plane with a repetitive pattern, based on the symmetries in this pattern (see Fig. 1.1, left). These symmetries are isometries, such as translations, rotations and reflec-

tions, and are classified by the 17 distinct *wallpaper groups* of the Euclidean plane. Special tilings are *regular tessellations* which are made of regular k-gons, $k \in \mathbb{N}$ and seamlessly cover the two-plane. There are only three possible regular tessellations, shown in Fig. 1.1, right: triangular ($k = 3$), quadrilateral ($k = 4$) and hexagonal tessellations ($k = 6$).

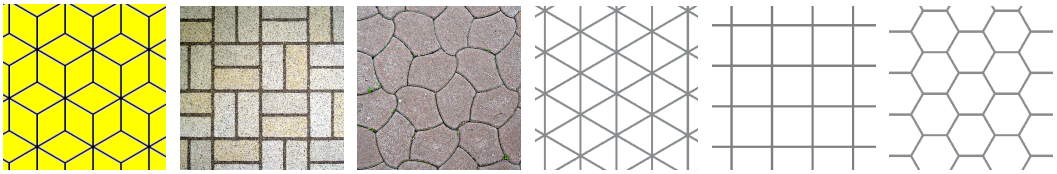


FIGURE 1.1: Tilings and regular tessellations of the two-plane.

The tiling of curved surfaces using regular patterns is an important yet challenging problem. Methods based on a local parameterization often lead to a visible breakup of the patterns along seams, i.e., where the surface is cut open during parameterization. Using a global parameterization can alleviate this problem when the translational and rotational symmetry in the parameterization is compatible with the tiling pattern.

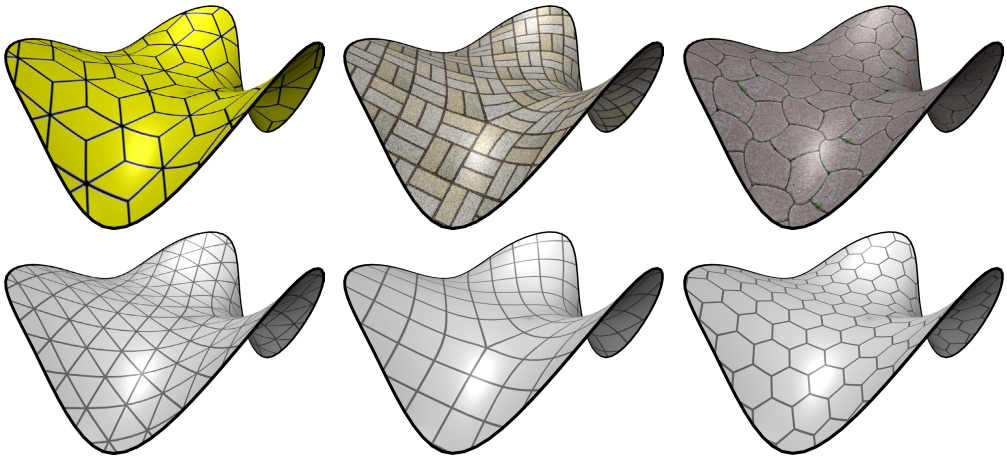


FIGURE 1.2: Tilings (top) and regular tessellations (bottom) on surfaces have applications in remeshing.

Most common surface parameterization approaches are made for quadrilateral tilings and generate a regular quadrilateral grid. In this thesis, different tilings and symmetry groups are also considered.

The proposed algorithm produces parameterizations with different symmetries which are suited for quadrilateral, triangular or hexagonal remeshing. But also various patterns of similar symmetry are possible, e.g. rhombic tilings or meshes with mixed tiles such as octagons and quadrilaterals.

Regular and symmetric patterns occur frequently in architecture and decorative art. The proposed method can therefore support the design process of architectural

constructions. Modern architectural styles often use freeform surfaces which are then covered by a regular structure. Glass roofs or facades are often modeled as unstructured triangle mesh and then segmented into regular triangles or quadrangles which define the shape of the glass panels (see Fig. 1.3).

FIGURE 1.3: Regular patterns on curved surfaces are used in architectural design. Left: Philological Library of the Freie Universität, Berlin. Middle: Exterior detail of the Selfridges Building, Birmingham. Right: 30 St Mary Axe Building, London.

1.1 Background on Parameterization

In classical differential geometry, a parameterized surface is often described as the image of an injective map $f : \Omega \to \mathbb{R}^3$, $(u, v) \mapsto (x, y, z)$ from a simply connected domain $\Omega \subset \mathbb{R}^2$.

This definition provides more than just a set of points in three-space. It also gives a one-to-one relation of the surface with the domain and therefore induces properties from the $\mathbb{R}^2$, e. g. the metric. The metric of a surface specifies how angles, lengths and areas are measured. Choosing a different parameterization – and therefore a different metric – will change the measures of these quantities.

When using a geographic map to navigate through a foreign territory we rely on having only little metric distortion to respect the actual shapes and sizes. Unfortunately, it follows from the famous "Theorema egregium" [Gauß 1828] that a perfect shape preserving parameterization does not exist for most surfaces. Thus, in each atlas of the earth, metric distortion cannot be avoided.

In the past, over 400 different maps are known implying different metric properties. Important ones are:

Preserving angles: Angle-preserving or *conformal* parameterizations may introduce enormous length and area distortion. Most famous examples of conformal maps of the earth are the stereographic projection and the Mercator projection.

Preserving area: An area-preserving parameterization maps all countries to a planar region of the same size. Thereby, its shape can be distorted very

much. An example is the azimuthally equal-area projection of J. H. Lambert [Lambert 1772] which is commonly used in atlases. The length distortion is low in the center of the map but increases towards the boundary.

Preserving distance: Preserving all measured lengths in a map of the earth would result in an isometry and is therefore impossible. There are maps which preserve a specific subset of distances. The azimuthal equidistant projection for example reproduces the exact distance of all points to the center of the map. It is used for radio services to determine distances from the own antenna.

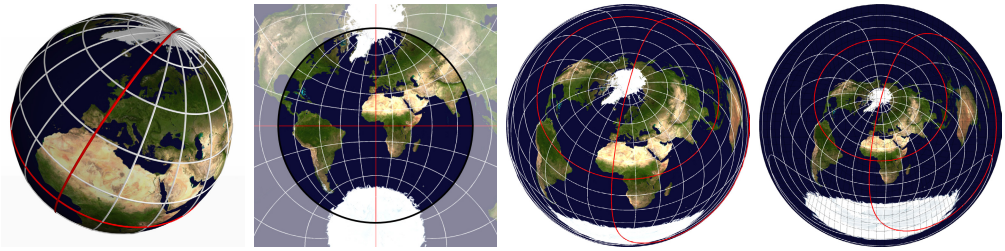


FIGURE 1.4: Parameterizations of the earth. From left to right: Parallel projection, Stereographic projection, Lambert azimuthal equal-area projection, Azimuthal equidistant projection.

Since it is in general impossible to achieve all of these properties, a practical parameterization must make compromises and minimizes all kinds of distortion in some sense.

1.2 The QuadCover Algorithm

In this thesis, an algorithm called QUADCOVER is proposed, which takes an unstructured triangle mesh as input and fully automatically provides a parameterization. The approach is based on a guiding *frame field* which is used to steer the shape of parameter lines. The parameterization pipeline consists of two major steps:

Construct a guiding frame field which encodes local geometric information about the surface, such as principal curvature directions or special surface features.

Compute a parameterization whose parameter lines follow the directions of the frame field in a best possible way.

A guiding-field-based parameterization method like QUADCOVER allows very specific control over the alignment of the parameterization and its metric. The automatically computed field in the first step is exchangeable by any other field provided as input. In this sense, QUADCOVER is a more general approach which can easily be adapted to match different user-specified criteria.

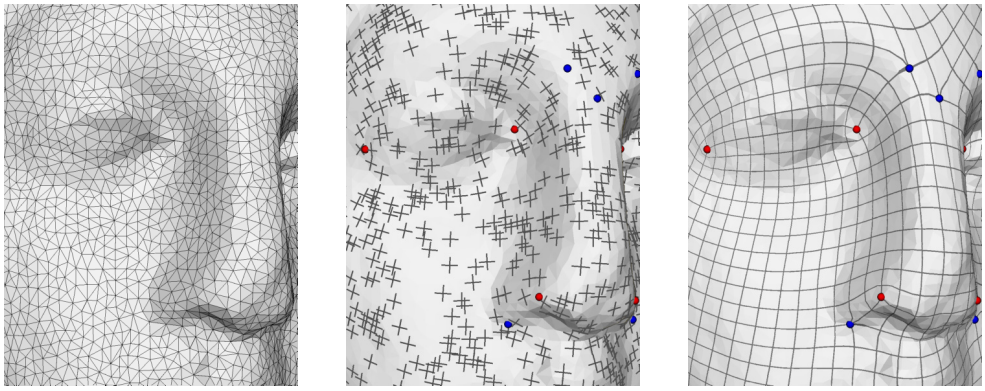


FIGURE 1.5: QUADCOVER pipeline. Left: Input triangle mesh. Middle: Guiding frame field. Right: Parameterization. Singularities are shown in red and blue.

Singularities. It turns out that the quality of the parameterization heavily depends on the proper choice of the guiding field. An unreasonable guiding field can cause huge distortion.

A key issue for generating good parameterizations is the meaningful placement of *irregular points* (also called *singularities*). They are already present in the guiding field and determine topological properties of the parameterization. Therefore, the design of a suitable field with meaningful singularities is essential for the algorithm.

The method proposed in this thesis provides an automatism for frame field generation. Additionally, the user may have explicit control over the number of singularities and their actual position on the surface.

Covering Spaces. The guiding frame field of a parameterization consists of a set of vectors in each point of the surface. It can therefore be understood as a multi-valued vector field.

A more sophisticated viewpoint uses an idea by Riemann, who replaced multi-valued complex functions with functions whose domain is a *Riemann surface*. In our context, a frame field naturally simplifies to a vector field defined on a domain which is a *branched covering* of the original surface. Singularities in the field correspond to *branch points* of the covering space. This framework provides a clear and conceptual notion which links the analysis of frame fields to the more explored vector-field calculus and, hence, provides an analogue theory for frame fields.

Note, that covering spaces are not required for the implementation of QUADCOVER, but are used for the theory to establish a coherent setting. When heard for the first time, the idea of parameterizing a covering surface might seem to be a complex construct. But once accepted, this setting indeed clarifies the theory, simplifies notion and furthermore provides a nice connection to Riemannian geometry.

1.3 Contributions and Outline

This thesis provides a theory and notion of global parameterization based on guiding frame fields. The concept of covering spaces is used for describing frame fields in terms of vector fields. Thereby we provide a different interpretation of parameterizations with rotational discontinuities. This concept is not only valid for QuadCover, but also for other approaches [Tong 2006, Bommes 2009] which implicitly define a covering surface.

The original QuadCover algorithm [Kälberer 2007] was developed in cooperation with Felix Kälberer and Konrad Polthier. Since then, the underlying notion of QuadCover was improved to make it more self-contained, and additional features are implemented such as an advanced pipeline for frame field generation.

A notable improvement is the introduction of constraints to a parameterization. In this thesis, two different types of constraints are defined: geometrical and combinatorial constraints. Geometrical constraints are defined by curves which are then exactly respected by a parameter line, e.g. the boundary or sharp creases. Combinatorial constraints allow prescribing special properties on the topology of the grid of parameter lines. Both techniques are very useful for generating nearly perfect parameterizations of complex models. As far as known to the author, this is the first attempt to formulate these kinds of constraints for a parameterization.

The second part of the thesis describes how parameterizations are used for surface tiling with different symmetries, especially for hexagonal- and stripe patterns. In summary, the main contributions are:

1. Formulate a variational approach QuadCover [Kälberer 2007] for generating a parameterization from a given guiding frame field together with F. Kälberer, K. Polthier.
2. Provide an advanced pipeline for field-generation as result of many discussions with F. Kälberer.
3. Reformulate the notion of frame fields using covering spaces, and show the equivalence to vector field calculus.
4. Introduce Hexagonal Parameterizations [Nieser 2011a] for hexagonal and triangular surface tiling together with J. Palacios, E. Zhang, K. Polthier.
5. Introduce Stripe Parameterizations [Kälberer 2010] for surface tiling using stripe patterns together with F. Kälberer, K. Polthier.
6. Provide a concept for handling singularities of positive integer index, e.g. poles or vortices [Nieser 2009] together with K. Polthier.
7. Introduce geometrical constraints to force parameter lines to run along given curves.
8. Introduce combinatorial constraints to enforce given conditions on the connectivity of the grid of parameter lines.

1.4 Previous Work on Surface Parameterization

The research area of surface parameterization has a long and fruitful tradition and there is a wealth of different approaches. Excellent overviews [Hormann 2008, Sheffer 2006, Floater 2005] are summarizing the methods up to 2008.

Early Work. The notion of barycentric coordinates was already used [Tutte 1963, Floater 1997] to unfold a simply connected domain into the plane. The boundary is thereby mapped to a predetermined closed curve. While the method is robust and the bijectivity of the parameterization is guaranteed, however, it does not adapt to extrinsic geometrical properties, such as curvature or ridges of the surface.

In subsequent years, further methods are proposed, optimizing different energies [Lévy 1998, Hormann 2000, Sheffer 2001] in order to reflect the shape of the surface in a best possible way.

Conformal. A new class of approaches has arisen with the study of conformal parameterizations. The notion of conformal maps is discretized in various ways: using circle packings on the surface [Stephenson 1999], minimizing a non-linear energy [Haker 2000] or defining a least-squares measure for conformality [Lévy 2002, Desbrun 2002]. An advantage of discrete conformal maps is that they can be computed very efficiently and angles are nearly perfectly preserved, however, at the cost of possibly large length distortion.

Global. All mentioned methods require that the surface is simply connected. More complicated surfaces are cut open into a simply connected disk introducing visible seams in the parameterization. A first impressive work for constructing global parameterizations of surfaces with arbitrary genus is given by Gu and Yau [Gu 2003], and then further improved [Jin 2004]. They take the homology class of the surface into account and propose an algorithm for computing conformal parameterizations satisfying periodic conditions along each homology cycle.

A different way of parameterizing surfaces of arbitrary genus is to map it to a different *domain surface* instead of the Euclidean plane. The domain surface has the same genus and is a simpler representation of the shape, e. g. a polycube [Tarini 2004] or a coarsened version of the input surface [Pietroni 2010].

Singularities. Another important development is the introduction of singularities (also called irregular points or cone points) of fractional index. They have proven to be essential for high quality parameterizations and increase the flexibility of purely conformal mappings. An approach [Kharevych 2006] uses user given singularities for conformal parameterizations via circle packings. Tong et al. use a hand-picked quadrilateral meta-layout for positioning singularities [Tong 2006]. They appear at the nodes of the layout, and the patches are then parameterized solving for a global harmonic one-form. Still, the approach is constrained by the global nature of harmonic one-forms, in some sense, similar to the algorithm of Gu and Yau [Gu 2003]. Another method [Dong 2006] uses a similar idea for parameterization but creates

a quadrilateral meta-layout automatically from the Morse-Smale complex of eigenfunctions of the mesh Laplacian.

An important issue, is the question how to find reasonable locations for singularities. The placement has huge impact on the quality and distortion of the parameter map. Two different approaches [Ben-Chen 2008, Springborn 2008] automatically place singularities optimized for conformal parameterizations. In general, an optimal placement is very hard to find and is an ongoing research problem today.

Curvature Alignment. The PGP algorithm [Ray 2006] introduces a new perspective by generating parameterizations guided by two orthogonal input vector fields. These directions are normally generated as being eigenvectors of the curvature tensor. However, the global continuous structure of the parameterization causes problems in the vicinity of singularities. These regions are detected and reparameterized afterwards.

The approach of QuadCover [Kälberer 2007], strongly inspired by their work, formulates a consistent framework for frame field calculus. Furthermore, an algorithm is proposed for generating a parameterization which aligns to a user-given guiding field including the correct handing of singularities. This guiding field can either be taken from the principal curvature tensor, or be any arbitrary field, e. g. generated by using one of the recent tools for designing rotational symmetry fields (see Section 1.5). QuadCover uses a simple heuristic to approximate the solution of the arising NP hard mixed integer problem.

Special attention must be paid to singularities of positive non-fractional index [Nieser 2009], since they cannot be represented by a piecewise linear texture map on a triangle mesh. This issue concerns all parameterization algorithms using linear finite elements.

Mixed Integer Quadrangulation (MIQ) [Bommes 2009] is a technique based on QuadCover. While optimizing the same energy, the numerical solver was exchanged to enhance the approximation of the minimum. More details and a comparison to this method are given in Section 5.2.

A further guiding-field based parameterization method [Zhang 2010] computes a wave function and generates its Morse-Smale complex which provides a decomposition for a subsequent parameterization. Zadravec et al. compute parameterizations which are optimized for use in architectural design [Zadravec 2010].

Finally, parameterizations can also be used to generate layouts which do not consist of quadrilaterals. Different symmetries, such as stripes [Kälberer 2010] or hexagons [Nieser 2011a], open a new perspective on generating regular tilings with surface parameterization.

1.5 Previous Work on Field Design

Vector Fields. This thesis studies parameterization methods which are based on a guiding field. The design of such vector- and tensor fields is a separate task than the parameterization itself.

Early approaches on vector field design specify a sparse set of vectors and interpolate them over the whole surface [Praun 2000, Turk 2001]. Further techniques give direct control over the field topology by directly specifying the location and indices of singularities [Theisel 2002, Li 2006, Zhang 2006, Fisher 2007].

Line Fields and Cross Fields. Vector fields can be generalized to admit different rotational symmetries. Zhang et al. concern the design of line fields that do not distinguish between forward and backward directions [Zhang 2007].

Fields with 90-degrees rotational symmetry are often called frame fields, cross fields or 4-RoSy fields. They are used to represent principal curvature directions and play an important role in surface parameterization. Cross fields are computed from smoothing pricipal curvature directions in [Hertzmann 2000]. A similar method is used in QuadCover [Kälberer 2007] for the input guiding field of a parameterization. Zhang et al. propose to resize principal curvature vectors in order to minimize the curl [Zhang 2010]. A more general framework [Zadravec 2010] uses pairs of conjugate directions which serve as a guiding field for parameterization.

Smooth cross fields are very stably computed by variational approaches which minimize the squared angles between vectors in adjacent triangles. Such an energy function is first proposed by Wei et al. [Wei 2001]. The MIQ algorithm [Bommes 2009] computes principal curvature directions in some trusted areas and uses a similar energy to interpolate the cross field to the whole surface. Singularities are found automatically using a mixed integer solver. The work of Crane et al. provides a more theoretical relation which shows that a cross field which minimizes a similar energy can be seen as a discrete Levi-Civita connection [Crane 2010].

N-Rotational Symmetry. Fields with rotational symmetry of $2\pi/N$ for arbitrary $N \in \mathbb{N}$ are first considered by Palacios and Zhang [Palacios 2007] and called N-RoSy fields. Based on this notion, further design tools are developed [Ray 2008, Ray 2009, Lai 2010]. HexCover [Nieser 2011a] computes a 6-RoSy field which aligns to principal curvature directions.

Recently, Palacios and Zhang [Palacios 2011] have presented a technique for visualization of N-RoSy fields using line integral convolution (LIC).

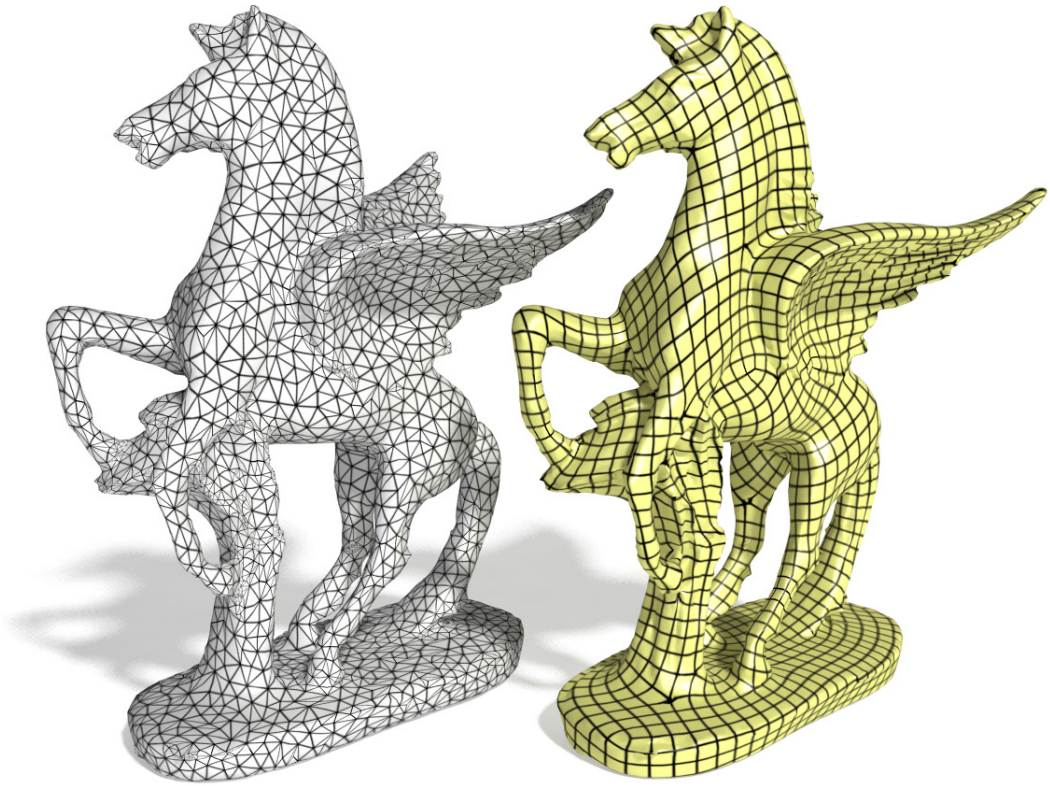


FIGURE 1.6: Parameterization with QUADCOVER. Left: Input triangle mesh. Right: Output quadrilateral structure.

Chapter 2

Field-Based Surface Parameterization

This chapter introduces the QuadCover algorithm which computes a parameterization from a given guiding frame field. Section 2.1 gives an introduction to the main idea while Section 2.2 settles the underlying notion used in the theory.

For illustrating the main ideas of QuadCover, the algorithm is divided into two parts: A simple version is considered in Section 2.3, excluding rotational symmetries. It is not intended for practical use, but introduces the main concept. The full algorithm is then explained in Section 2.4.

2.1 QuadCover Approach

Parameterization. The goal of QuadCover is to construct a parameterization for a given surface. The concepts described in this section are based on a setting from differential geometry. Hereby, the underlying surface is a smooth *two-dimensional manifold*, which is a second-countable Hausdorff space M covered by a collection of charts M_i. The charts are homeomorphic to a two-dimensional Euclidean disk and may overlap, i.e. a single point of M may be represented in several charts.

Definition 2.1.1 *A **parameterization** φ of a manifold M with charts M_i is a collection of injective maps $\varphi_i : M_i \to \mathbb{R}^2$, $p \mapsto (u, v)^T$. For all overlapping charts M_i, M_j, the map $\varphi_j \cdot \varphi_i^{-1} : \varphi_i(M_i \cap M_j) \to \varphi_j(M_i \cap M_j)$ is called the **transition function** between charts.*

Note that in this definition, φ maps from the manifold into $\mathbb{R}^2$ which is inverse to the common definition of parameterized surfaces in differential geometry (see Section 1.1). However, this notion is more practicable when different possible parameterizations of the same surface are considered.

For a non-degenerate parameterization, the set of points with one constant coordinate forms a collection of one-dimensional curves, called ***parameter lines*** of a parameterization. Define u-lines by restricting v, and v-lines by restricting u to a constant.

Consider all parameter lines whose isovalue is an integer number. They form a quadrilateral pattern on each chart and are given as preimage $\varphi_i^{-1}(C^2)$ of the unit grid in $\mathbb{R}^2$:

$$C^2 := \{(u, v) \in \mathbb{R}^2 \,|\, u \in \mathbb{Z} \text{ or } v \in \mathbb{Z}\}. \tag{2.1}$$

In order to obtain a globally consistent quadrangular grid the pattern has to match up in all overlapping charts (see Fig. 2.1). We therefore restrict all transition functions to those (linear) functions which leave C^2 invariant:

Definition 2.1.2 *A* quadrangular global parameterization *is a parameterization φ whose values in adjacent charts M_i, M_j are related by:*

$$\varphi_j(\boldsymbol{p}) = \boldsymbol{J}^{r_{ij}}\varphi_i(\boldsymbol{p}) + w_{ij}, \quad r_{ij} \in \mathbb{Z}, w_{ij} \in \mathbb{Z}^2, \quad \forall \boldsymbol{p} \in M_i \cap M_j, \tag{2.2}$$

*where $\boldsymbol{J}$ is a rotation by $\pi/2$ in the Euclidean plane. w_{ij} is called **gap** between M_i and M_j and denotes a constant integer translation. r_{ij} (called **matching**) is a number representing the rotational discontinuity between both charts.*

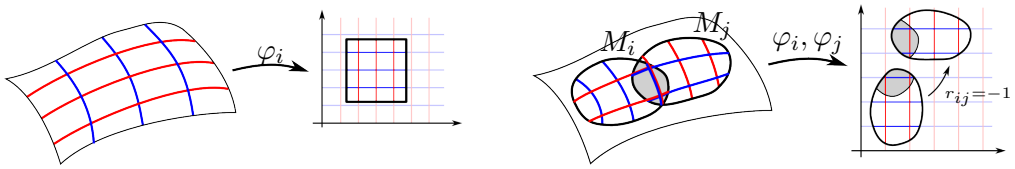


FIGURE 2.1: Left: Parameterization φ_i with u-lines (blue) and v-lines (red). Right: Parameterization of adjacent charts with matching $r_{ij} = -1$.

Frame Fields. The Jacobian of a parameterization $\varphi_i = (u_i, v_i)^T$ is given by $\mathrm{Jac}(\varphi_i) = (\nabla u_i, \nabla v_i)^T \in \mathbb{R}^{2x3}$, with ∇u_i and ∇v_i being the gradient fields of the components of φ_i. These gradients define two vectors in each tangent space which are perpendicular to the u- resp. v-lines (see Fig. 2.2, left). The length of these vectors corresponds to the "speed" of the parameterization (or the density of parameter lines) in u- and v-direction.

If the parameterization is not degenerated, then $(\nabla u_i(p), \nabla v_i(p))$ forms a basis of the tangent space at p and is therefore called a coordinate ***frame***. In overlapping charts M_i, M_j, gradient frames of a parameterization are related by the formal matrix-vector multiplication:

$$\begin{pmatrix} \nabla u_j \\ \nabla v_j \end{pmatrix} = \mathbf{J}^{r_{ij}} \begin{pmatrix} \nabla u_i \\ \nabla v_i \end{pmatrix}. \tag{2.3}$$

Thus, the matching r_{ij} determines how the vectors in both charts are combinatorially identified (or matched), see Fig. 2.2, right. Since the matchings determine the global topology of the field, they must be specified in the definition of any frame field.

Definition 2.1.3 *A **frame field** X on a manifold is defined by two vector fields $X_i := (U_i, V_i)$ in each chart M_i together with matchings $r_{ij} \in \{0,1,2,3\}$ given for each pair of overlapping charts M_i, M_j. Matchings are anti-symmetric, i.e. $r_{ij} = -r_{ji}$ mod 4, and transitive, i.e. if three charts M_i, M_j, M_k (or more) overlap in the same point, then: $(r_{ij} + r_{jk})$ mod $4 = r_{ik}$. Frames in adjacent charts M_i, M_j are related by: $\binom{U_j}{V_j} = \boldsymbol{J}^{r_{ij}} \binom{U_i}{V_i}$.*

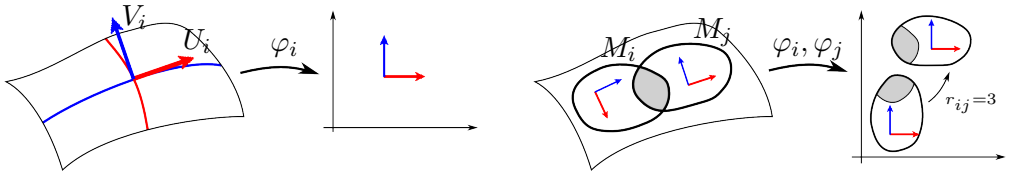


FIGURE 2.2: Left: Local coordinate frame on a surface. The frame vectors are perpendicular to the corresponding parameter line. Right: Frames in adjacent charts with matching $r_{ij} = 3$.

A gradient frame field provides information on metric properties of the parameterization, e.g. let $\nabla u_{|p}$, $\nabla v_{|p}$ be the gradients of a parameterization (u_i, v_i) in $p \in M$, then:

1. φ is conformal $\Leftrightarrow$ $\langle \nabla u_{|p}, \nabla v_{|p} \rangle = 0$, $\|\nabla u_{|p}\| = \|\nabla v_{|p}\|$, $\quad \forall p \in M$
2. φ is area-preserving $\Leftrightarrow$ $\|\nabla u_{|p} \times \nabla v_{|p}\| = 1$, $\quad \forall p \in M$
3. φ is length-preserving $\Leftrightarrow$ $\langle \nabla u_{|p}, \nabla v_{|p} \rangle = 0$, $\|\nabla u_{|p}\| = \|\nabla v_{|p}\| = 1, \forall p \in M$

In general, not all of these properties can be fulfilled at the same time. Even point 3 alone is impossible to achieve if the surface has non-vanishing Gauß curvature.

Energy Formulation. The concept of QUADCOVER is to separate the problem into designing a guiding frame field X with $X_i = (U_i, V_i)$ which represents the desired metric properties and then finding a corresponding parameterization. More specifically, a parameterization with $\nabla\varphi \approx X$ is computed, i.e. approximating the frame field best possible in L_2-norm by minimizing the QUADCOVER energy:

$$E(\varphi) := \int_M \|\nabla\varphi - X\|^2 \mathbf{d}A. \tag{2.4}$$

Here, $\mathbf{d}A$ denotes the two-dimensional surface element.

If U_i and V_i e.g. are perpendicular and of the same length, then the parameterization will tend to have low angle distortion. Even though, the exact conformal case will not be obtained in general, the parameterization will find an intermediate state between being conformal and aligning to the frame directions. In the same manner, an input field for optimizing area or length distortion can be defined by using frames with $|U_i \times V_i| = 1$ resp. perpendicular vectors with $|U_i| = |V_i| = 1$.

It turns out that the choice of a good frame field is a challenging problem. An arbitrary field may lead to unintended artifacts and possibly large metric distortion

in the parameterization. On the other hand, the setup enables the user to have specific control over the parameterization. Special requirements can easily be incorporated, e.g. alignment to surface features or adapting the density of parameter lines locally by scaling the frame field.

QuadCover Pipeline. A key observation is that the optimization of the energy in Eqn. (2.4) can be divided into two subproblems and solved independently. They define the two main steps of QuadCover:

1. **(local step)** Remove the curl from X to get a new frame field which is locally integrable.
2. **(global step)** Compute a *global* parameterization which is consistent in all overlapping charts.

Both steps do agree with each other, i.e. they optimize the same energy. It does not matter if they are combined into one, looking for the best aligning global parameterization, or if they are solved successively.

The next section describes the discrete setting while the discrete QuadCover algorithm is then described in Sections 2.3 and 2.4.

2.2 Setting

This section settles the underlying notion and reviews standard definitions of discrete differential geometry. Section 2.2.1 reviews basic notion of discrete surfaces. Discrete function spaces and differential operators are introduced in Section 2.2.2 and 2.2.3 and are similar to those introduced by Polthier [Polthier 2002].

2.2.1 Discrete Manifolds

As being a common data structure for representing surfaces in discrete geometry, we consider *polyhedral surfaces*. A polyhedral surface M_h consists of a set of triangles T which are glued along their edges so that the result is homeomorphic to a 2-dimensional manifold. Henceforth, we only consider surfaces which are connected and orientable. The set of vertices and edges is referenced by V and E, the total number of vertices, edges and triangles is denoted by $|V|$, $|E|$, resp. $|T|$.

Charts. In this thesis, polyhedral surfaces are considered as being two-dimensional *discrete manifolds*, where each triangle is a single chart. The charts of two adjacent triangles overlap at their common edge. This chart-based view resolves issues arising when surfaces with more complicated topology are parameterized.

Definition 2.2.1 *Let M_h be a discrete manifold with triangles (charts) T. A* ***discrete parameterization*** *φ of M_h consists of two piecewise linear scalar functions*

$\varphi_i = (u_i, v_i)$ on each chart $t_i \in T$. On the common edge between adjacent charts t_i, t_j, the value may differ in both charts and is related by the ***transition map*** *$\varphi_j \cdot \varphi_i^{-1}$.*

Analoguous to the smooth case, the transition maps are restricted by Eqn. (2.2) in order to define a *discrete quadrangular global parameterization*. In other words, the values of φ in t_i and t_j are related by the matchings r_{ij} and gaps g_{ij} which are constant per edge. Fig. 2.3, left shows a parameterization with rotational discontinuity.

According to Definition 2.1.3, a ***discrete frame field*** X is given by two constant vectors $X_i = (U_i, V_i)$ per triangle t_i together with matchings $r_{ij} \in \{0, 1, 2, 3\}$ for all edges in-between adjacent triangles t_i, t_j. The QuadCover energy (see Eqn. (2.4)) becomes:

$$E(\varphi) := \sum_{t_i \in T} A_i \|\nabla \varphi_i - X_i\|^2, \tag{2.5}$$

where A_i denotes the area of triangle t_i.

Euler Characteristic. An important invariant of a discrete manifold is the *Euler characteristic* $\chi = |T| - |E| + |V|$ which is related to the *genus* of the surface, counting the number of handles. If the boundary of M splits into $b \geq 0$ connected components, the genus is related to the Euler characteristic by $\chi + b = 2 - 2g$.

Fundamental group. When analyzing parameterizations, we investigate curves on surfaces. On a discrete manifold, a curve may pass arbitrarily along edges or through the inner of triangles. Two curves are called *homotopic* if they can be continuously deformed into each other.

On a polyhedral surface M_h, the **first fundamental group** $\pi_1(M_h, p)$ is defined similar to that of a differentiable surface. It consists of equivalence classes of all loops, passing through the root point $p \in M_h$. $\pi_1(M_h, p)$ has a group structure: curves are added by concatenation and inverted by reversing their direction. The neutral element is the class of curves which bound a simply connected disk, called **trivial loop**. For further material on the fundamental group, see related topology books, e.g. [Fulton 1995, Munkres 1999].

If M_h is closed (i.e. it has no boundary), then $\pi_1(M_h, p)$ is of dimension $2g$. There is a set of $2g$ curves, which span the whole group (as in Fig. 2.3, right). If there are $b > 0$ connected boundary components, then the dimension is $2g + b - 1$.

The generators of the fundamental group are used in QuadCover to describe harmonic vector fields (see Section 2.3.2).

2.2.2 Discrete Function Spaces

Given a discrete manifold M_h, we consider piecewise linear scalar or vector valued functions.

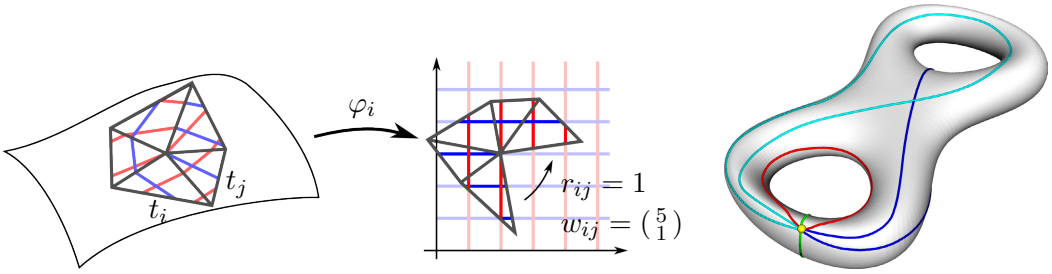


FIGURE 2.3: Left: Each triangle is considered as a chart and mapped to $\mathbb{R}^2$. The transition between triangles t_i, t_j is a 90-degrees rotation. Right: Generators of the fundamental group of a genus 2 manifold.

Definition 2.2.2 *Let*

$$S_h = \{u : M_h \to \mathbb{R} \,|\, u \text{ is linear on each triangle and continuous on } M_h\} \tag{2.6}$$

be the space of ***conforming finite elements****. A basis of S_h is given by the* ***Lagrange basis functions*** *$\Phi_i \in S_h$ (also called nodal basis), which are defined by their values on all vertices $p_j \in V$ by $\Phi_i(p_j) := \delta_{ij}$ (Fig. 2.4, top).*

Each conforming function $u \in S_h$ is uniquely represented as a linear combination $u = \sum_{i=0}^{|V|-1} u_i \Phi_i$. In the subsequent sections, we will use the letter u to indicate the function itself as well as its vector representation $\vec{u} = (u_0, \ldots, u_{|V|-1})$.

Furthermore, we investigate another function space with data located on edges of M_h:

Definition 2.2.3 *Let*

$$\begin{aligned} S_h^* = \{u : M_h \to \mathbb{R} \,|\, & u \text{ is linear on each triangle and} \\ & \text{continuous at all edge midpoints} \,\} \end{aligned} \tag{2.7}$$

be the space of ***non-conforming finite elements****. A basis of this space is given by the nodal basis functions Ψ_i with values $\Psi_i(m_j) = \delta_{ij}$ on the edge midpoints m_j (Fig. 2.4, bottom).*

Again, each function $u^* \in S_h^*$ is uniquely represented by its coefficient vector $\vec{u}^* = (u_0^*, \ldots, u_{|E|-1}^*)$ with $u^* = \sum_{i=0}^{|E|-1} u_i^* \Psi_i$.

In this context, vector fields are always restricted to be piecewise constant since we mainly consider *gradient fields* ∇u of a piecewise linear function u (in either S_h or S_h^*).

Definition 2.2.4 *Let $\mathcal{X}_\mathrm{h}$ be the space of* ***piecewise constant vector fields*** *with one vector in the tangent space of each triangle.*

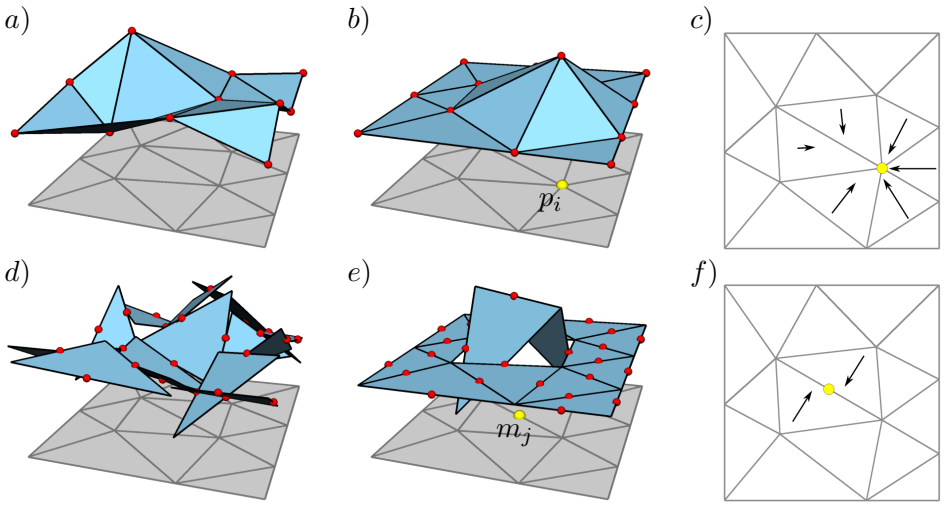


FIGURE 2.4: Scalar functions visualized as graph over a triangle domain. a) Conforming function $u \in S_h$, b) Basis Φ_i at vertex p_i, c) Gradient field $\nabla\Phi_i$, d) Non-conforming function $u^* \in S_h^*$, e) Basis Ψ_j at edge midpoint m_j, f) Gradient field $\nabla\Psi_j$.

2.2.3 Discrete Differential Operators

We have now defined the underlying function spaces used in the theory. For doing differential geometry, we also need to specify how differential operators act on a function, like the divergence, the curl and the Laplacian.

Since discrete functions are not differentiable in the seooth sense, it makes no sense to speak about pointwise evaluation of differential operators. Instead, discrete differential operators are defined as functionals, which only gives a real number when paired with a *test function*. The test function has to be in S_h (for conforming elements) resp. in ${S_h}^*$ (for non-conforming elements). Applying the nodal basis functions Φ_i resp. Ψ_i as test function can be understood as the value of the operator at a specific vertex resp. an edge of the mesh.

Divergence. To motivate the definition of discrete divergence, consider a closed differential manifold M with a vector field U and a scalar function u. Then the following relation holds:

$$\int_M \langle \operatorname{div} U, u \rangle \, \mathbf{d}A = -\int_M \langle U, \nabla u \rangle \, \mathbf{d}A. \tag{2.8}$$

The left term is a weak formulation of $\operatorname{div} X$, using u as test function. In the discrete case, the divergence is defined as follows:

Definition 2.2.5 *The **discrete divergence** of a vector field $U \in \mathcal{X}_\mathrm{h}$ at a vertex*

p_i, resp. at an edge e_i is defined as

$$\operatorname{div} U(p_i) := -\int_{M_h} \langle U, \nabla\Phi_i \rangle \, \mathbf{d}A \tag{2.9}$$

$$\operatorname{div}^* U(e_i) := -\int_{M_h} \langle U, \nabla\Psi_i \rangle \, \mathbf{d}A \tag{2.10}$$

The divergence operator measures the "flux" into the vertex star resp. edge star.

Curl. The *curl*-operator measures the "rotation" around a given point. Let $\mathbf{J}$ be the operator which takes a vector field and rotates each vector by 90 degrees in its tangent plane in mathematical positive direction. On a differentiable two-dimensional manifold, the curl is then given as the divergence of the rotated field: $\operatorname{curl}(U) = \operatorname{div}(\mathbf{J}U)$.

Definition 2.2.6 *The discrete curl of a vector field $U \in \mathcal{X}_\mathrm{h}$ at a vertex p_i, resp. at an edge e_i is defined as*

$$\operatorname{curl} U(p_i) := -\int_{M_h} \langle \boldsymbol{J}U, \nabla\Phi_i \rangle \, \mathbf{d}A \tag{2.11}$$

$$\operatorname{curl}^* U(e_i) := -\int_{M_h} \langle \boldsymbol{J}U, \nabla\Psi_i \rangle \, \mathbf{d}A = \frac{1}{2} \left\langle U_{|t_r} - U_{|t_l}, e_i \right\rangle, \tag{2.12}$$

where $U_{|t_r}$, $U_{|t_l}$ are the vectors given in adjacent triangles on the right resp. left side of the (oriented) edge e_i.

Integrability. There is a fundamental relation between gradient fields and curl-free vector fields which carries over from the differential setting. Consider a function $u \in S_h$ and an edge e between adjacent triangles t_r, t_l. Then, the gradients $\nabla u_{|t_r}$ and $\nabla u_{|t_l}$ match in the sense, that their path integral along the common edge e coincide: $\left\langle \nabla u_{|t_r}, e \right\rangle = \left\langle \nabla u_{|t_l}, e \right\rangle$. According to Eqn. (2.12), this is equivalent to $\operatorname{curl}^* \nabla u(e) = 0$. More generally:

Theorem 2.2.1 [Polthier 2002] *Let $U \in \mathcal{X}_\mathrm{h}$ be a vector field on a simply connected surface M_h. Then:*

$$\textit{There exists a } u \in S_h \textit{ with } \nabla u = U \quad \Leftrightarrow \quad \textit{curl}^*(U) = 0 \tag{2.13}$$

respectively

$$\textit{There exists a } u^* \in S_h^* \textit{ with } \nabla u^* = U \quad \Leftrightarrow \quad \textit{curl}(U) = 0. \tag{2.14}$$

Note. In order to compute the potential $u \in S_h$ of a curl*-free field U, set $u(v_0) := 0$ at an arbitrary root vertex p_0. For any other vertex p, take an arbitrary curve γ from p_0 to p and set $u(p) := \int_\gamma \langle U, \gamma' \rangle \, \mathbf{d}s$. This integral is independent of the choice

of γ. For example, choose a curve $\gamma = (p_0, p_1, \ldots, p_{n-1}, p_n = p)$, $n \in \mathbb{N}$ which runs only on edges. The value at v is then computed as:

$$u(p) = \sum_{i=0}^{n-1} \langle p_{i+1} - p_i, U_{|t_i} \rangle \tag{2.15}$$

Here, t_i denotes a triangle adjacent to the edge (p_i, p_{i+1}). Since $\operatorname{curl}^* U = 0$, it does not matter which adjacent triangle is chosen.

Theorem 2.2.1 does not hold if the surface is not simply connected. In general, the path integral around a non-trivial loop, such as a handle or a hole, does not vanish. However, the value of the path integral only depends on the homotopy class of the loop γ and is called a *period* of U.

Definition 2.2.7 *A vector field $U \in \mathcal{X}_\mathrm{h}$ is called **locally integrable** in S_h resp. S_h^*, if*

$$\operatorname{curl}^* U = 0, \text{ resp. } \operatorname{curl} U = 0. \tag{2.16}$$

*It is called **globally integrable**, if it is locally integrable and all periods vanish.*

A vector field U is therefore a gradient field if and only if it is globally integrable.

Harmonic Fields. Harmonic vector fields play a crucial role in surface parameterization. In the differential case, a vector field is called harmonic if it is free of curl and of divergence.

Definition 2.2.8 *A vector field $H \in \mathcal{X}_\mathrm{h}$ is called **discrete harmonic** if it satisfies*

$$\operatorname{curl}^* H = 0 \quad \text{and} \quad \operatorname{div} H = 0 \tag{2.17}$$

A harmonic field is by definition locally integrable. It is furthermore uniquely defined, if all periods are given. The set of harmonic fields therefore forms a vector space which is isomorphic to the fundamental group $\pi_1(M_h, p)$, $p \in M_h$ and has dimension $2g$ (resp. $2g + b - 1$, if $b > 0$ boundary components are present).

Definition 2.2.9 *The **discrete Laplace operator** of a function $u \in S_h$ at a vertex p_i is defined as*

$$\triangle u(p_i) := -\int_{M_h} \langle \nabla u, \nabla \Phi_i \rangle \, \mathbf{d}A \tag{2.18}$$

*A function $u \in S_h$ with $\triangle u = 0$ is called **discrete harmonic**.*

According to Definition 2.2.5, the gradient field of a harmonic function always satisfies $\operatorname{div} \nabla u = 0$ and is therefore a discrete harmonic vector field.

Using the representation of u as a coordinate vector $\vec{u}$, the Laplace operator can be expressed as a quadratic functional:

$$\triangle u = \vec{u}^t L \vec{u}, \tag{2.19}$$

where L is a $|V| \times |V|$ matrix with entries

$$\begin{aligned} L_{ij} &= \int_{M_h} \langle \nabla \Phi_i, \nabla \Phi_j \rangle \, \mathbf{d}A \\ &= \begin{cases} -\frac{1}{2}(\cot \alpha_{ij} + \cot \beta_{ij}), & \text{if } p_i \text{ adjacent to } p_j \\ -\sum_{k \neq i} L_{ik}, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{2.20}$$

α_{ij} and β_{ij} are the two angles which are opposite to the edge e_{ij} in both adjacent triangles (Fig. 2.5, left).

There is also a non-conformal version of the Laplace matrix L^*, which is given by:

$$\begin{aligned} L^*_{ij} &= \int_{M_h} \langle \nabla \Psi_i, \nabla \Psi_j \rangle \, \mathbf{d}A \\ &= \begin{cases} -\frac{1}{2} \cot \delta_{ij}, & \text{if } m_i \neq m_j, \text{ but in same triangle} \\ -\sum_{k \neq i} L^*_{ik}, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{2.21}$$

α_{ij} denote the angle between the corresponding edges of m_i and m_j in the common triangle (Fig. 2.5, right).

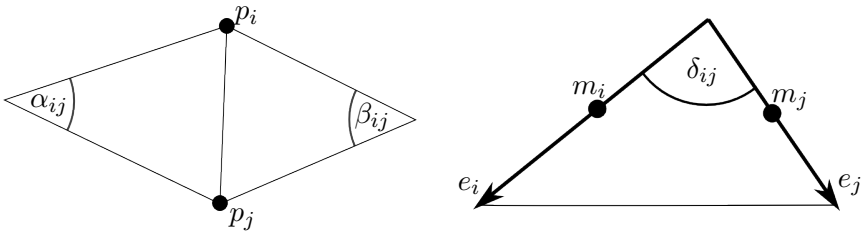


FIGURE 2.5: Left: Angles α_{ij}, β_{ij} from Eqn. (2.20) are opposite to edge (p_i, p_j). Right: Angle δ_{ij} from Eqn. (2.21) is between edges e_i, e_j.

2.3 Simplified QuadCover

For explanatory reasons, the formulation of QuadCover is divided into two parts: a simplified version and a full version. This section focuses on the simplified version which restricts the symmetries of the texture to pure translations only. The full algorithm allowing also rotational symmetries is explained in Section 2.4.

2.3.1 Variational Formulation

Starting from a given input frame field $X = (U_i, V_i)$, a parameterization $\varphi = (u_i, v_i)$ is computed which maps each triangle t_i into the plane adhering valid transitions

from Eqn. (2.2). For simplified QuadCover, all matchings r_{ij} are restricted to 0, thus all transition functions between adjacent triangles are pure translations.

As a consequence, according to Definition 2.1.3, a frame field decouples into two vector fields $U_i, V_i \in \mathcal{X}_\mathrm{h}$; all transition functions match U_i with U_j and V_i with V_j vectors. The QuadCover-Energy (see Eqn. (2.5)) decouples into two energy terms $E(u_i, v_i) := E_U(u_i) + E_V(v_i)$ with

$$E_U(u_i) = \sum_{t_i \in T} A_i \|\nabla u_i - U_i\|^2, \quad E_V(v_i) = \sum_{t_i \in T} A_i \|\nabla v_i - V_i\|^2, \tag{2.22}$$

which can be minimized separately. The remainder of this section focuses on minimizing E_U only. Finding a minimizer of E_V is done similarly.

QuadCover computes a solution u_i from a function space $\hat{S}_h$ which is a generalization of the conforming finite element space S_h to discontinuous functions.

Definition 2.3.1 *Let M_h be a discrete manifold with triangles T. Define the linear function space*

$$\begin{aligned}\hat{S}_h := \{(u_i : t_i \to \mathbb{R})_{t_i \in T} \,|\, & u_i \text{ is linear on triangle } t_i, \\ & \text{for adjacent } t_i, t_j \text{ and } p \in t_i \cap t_j : \; u_i(p) = u_j(p) + w_{u,ij},\, w_{u,ij} \in \mathbb{R}\}.\end{aligned} \tag{2.23}$$

A function $u_i \in \hat{S}_h$ is uniquely given by its values at vertices in all its incident triangles (i.e. 6——F— many scalar values). The difference between values in adjacent triangles are restricted to a constant gap $w_{u,ij}$ along the common edge. Since one of these constraints turns out to be redundant (see Section 2.3.2), and due to the relation $2|E| = 3|F|$ for closed triangulations, the dimension of $\hat{S}_h$ on a closed surface is $\dim(\hat{S}_h) = 3\,|F| - |E| - 1 = |E| - 1$.

Two given functions $u, v \in \hat{S}_h$ (with gaps $w_{u,ij}$ resp. $w_{v,ij}$) define a global parameterization (u, v) according to Definition 2.1.2 if and only if all $w_{u,ij}$ resp. $w_{v,ij}$ are in $\mathbb{Z}$. This integer constraint turns the energy minimization into an NP-hard problem which is equivalent to the known so-called *closest vector problem*. In practice, if the number of integer variables is too high we cannot get the optimal solution in reasonable time.

However, in order to generate a parameterization, QuadCover uses a heuristic to achieve a nearly-optimal result very efficiently: In the first stage (the local step), E_U is minimized for $u \in \hat{S}_h$ ignoring the integer constraint. In a second stage (the global step), the gaps are enforced to integer values yielding a globally consistent parameterization. The heuristic is fast and robust and in practice provides nearly optimal results.

2.3.2 Vector Field Analysis

The local step of QuadCover minimizes the energy E_U from Eqn. (2.22) for $u_i \in \hat{S}_h$ (resp. E_V for v_i). A direct solution would be to set all partial derivatives to 0 and solve this linear system of equations using Lagrange multiplier for the constraints. However, the number of variables is quite large ($3\,|F|$) with $|E|$ many constraints. From a numerical point of view, it is better (and of course more elegant) to eliminate redundant variables and solve the system without any constraints.

This section focuses on analysing the input vector field U and provides further observations about the minimizer of E_U. The analysis involves the *Hodge-Helmholtz decomposition* of vector fields. While coming from differential geometry, a discrete version of this theorem was given in [Polthier 2003].

The set of piecewise linear vector fields $\mathcal{X}_\mathrm{h}$ is a linear vector space, equipped with the L^2 scalar product $\langle U, V\rangle_2 := \int_{M_h} \langle U, V\rangle \,\mathbf{d}A$, $U, V \in \mathcal{X}_\mathrm{h}$. Consider the following linear subspaces of $\mathcal{X}_\mathrm{h}$:

$\mathcal{X}_\mathrm{h}^\mathrm{g}$ is the space of *gradient vector fields* in $\mathcal{X}_\mathrm{h}$. Any $U \in \mathcal{X}_\mathrm{h}^\mathrm{g}$ can be written as $U = \nabla f$ for some $f \in S_h$ (vanishing on the boundary). A gradient vector field satisfies $\mathrm{curl}^*\, U = 0$, thus it is free of vorticity; critical points of U are sinks and sources (Fig. 2.6, bottom left).

$\mathcal{X}_\mathrm{h}^\mathrm{cg}$ is the space of *co-gradient fields* with $V = \mathbf{J}\nabla g$, $g \in S_h^*$ (also vanishing on the boundary). A co-gradient field yields $\mathrm{div}\, V = 0$ and is therefore free of sinks and sources, but may have vortices (Fig. 2.6, bottom middle).

$\mathcal{H}_\mathrm{h}$ is the space of *harmonic vector fields*, i.e. those fields which apply $\mathrm{curl}^*\, H = 0$ and $\mathrm{div}\, H = 0$. On a genus g surface without boundary, the dimension of $\mathcal{H}_\mathrm{h}$ is $2g$. It consists of all more or less "parallel" flows around the topological handles and the holes of M_h (Fig. 2.6, bottom right).

Notice that $\mathcal{X}_\mathrm{h}^\mathrm{g}$, $\mathcal{X}_\mathrm{h}^\mathrm{cg}$ and $\mathcal{H}_\mathrm{h}$ are perpendicular, i.e. for any two fields in different spaces, their L^2 scalar product vanishes. Furthermore, according to the following theorem, these spaces span the whole $\mathcal{X}_\mathrm{h}$ (see Fig. 2.6).

Theorem 2.3.1 [Polthier 2003, Wardetzky 2006] ***(Discrete Hodge-Helmholtz decomposition)*** *Let M_h be a polyhedral surface. The space of vector fields $\mathcal{X}_\mathrm{h}$ decomposes into the direct sum*

$$\mathcal{X}_\mathrm{h} = \mathcal{X}_\mathrm{h}^\mathrm{g} \oplus \mathcal{X}_\mathrm{h}^\mathrm{cg} \oplus \mathcal{H}_\mathrm{h}, \tag{2.24}$$

i.e. each vector field $U \in \mathcal{X}_\mathrm{h}$ can be uniquely written as:

$$U = \nabla f + \boldsymbol{J}\nabla g^* + H, \tag{2.25}$$

with $f \in S_h, g^* \in S_h^*, H \in \mathcal{H}_\mathrm{h}$.

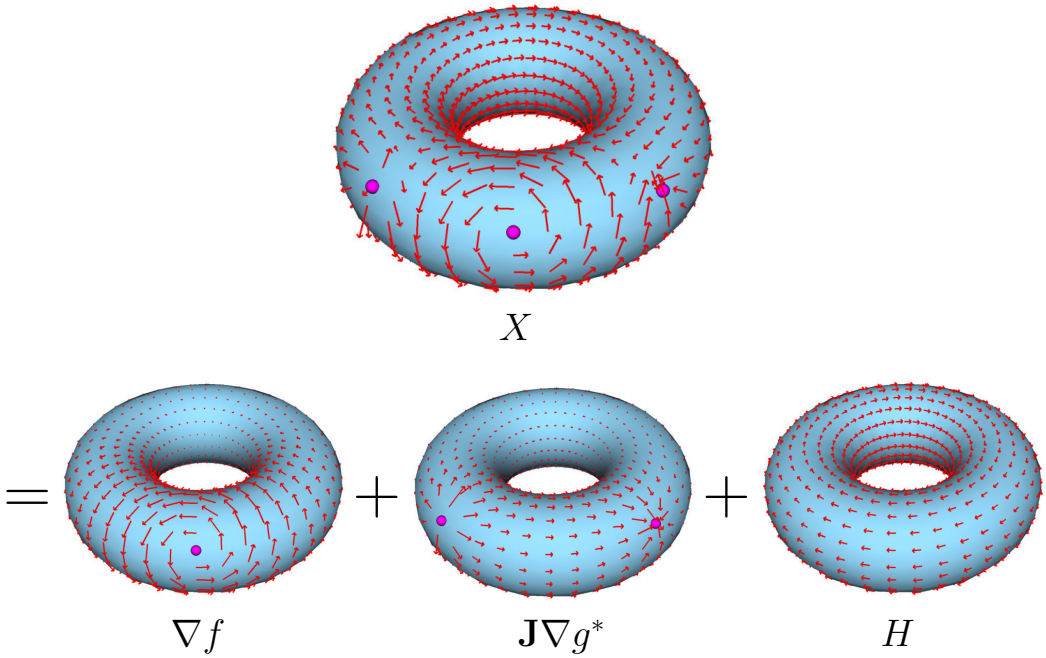


FIGURE 2.6: Hodge-Helmholtz decomposition of a vector field X.

Notice that by definition, $\text{curl}^*\,\nabla f = \text{curl}^*\,H = 0$, and hence the fields ∇f and H are locally integrable. The whole curl of U is concentrated in the co-potential term $\mathbf{J}\nabla g^*$.

Lemma 2.3.1 *The space* $\nabla\hat{S}_h := \{\nabla f \mid f \in \hat{S}_h\}$ *satisfies:*

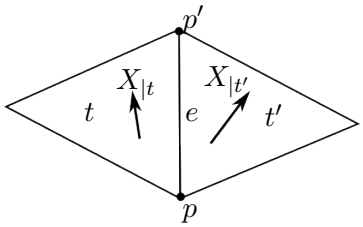
$$\nabla\hat{S}_h = \mathcal{X}_{\text{h}}^{\text{g}} + \mathcal{H}_{\text{h}}\,. \tag{2.26}$$

Thus $\hat{S}_h$ *contains exactly the potentials of all locally integrable vector fields.*

Proof. For any piecewise linear function u (possibly discontinuous at edges), the derivative $X := \nabla u$ is a piecewise constant vector field $X \in \mathcal{X}_{\text{h}}$. Also, any given vector field $X \in \mathcal{X}_{\text{h}}$ can be integrated on all triangles locally resulting in a piecewise linear function u with $\nabla u = X$.

By definition, $u \in \hat{S}_h$ if and only if for each edge e connecting the vertices p, p', the gap between the adjacent triangles t, t' is constant, i.e.:

$$\begin{aligned}
& u(p)_{|t} - u(p)_{|t'} = u(p')_{|t} - u(p')_{|t'} \\
\Leftrightarrow\quad & u(p)_{|t} - u(p')_{|t} = u(p)_{|t'} - u(p')_{|t'} \\
\Leftrightarrow\quad & \langle X_{|t}, p' - p\rangle = \langle X_{|r'}, p' - p\rangle \\
\Leftrightarrow\quad & \text{curl}^*\, X(e) = 0
\end{aligned}$$



which is equivalent to $X \in \mathcal{X}_{\text{h}}^{\text{g}} + \mathcal{H}_{\text{h}}$.

□

Theorem 2.3.2 *Let $U = \nabla f + \mathbf{J}\nabla g^* + H$ be the Hodge-Helmholtz decomposition of a vector field $U \in \mathcal{X}_{\mathrm{h}}$ with $f \in S_h$, $g^* \in S_h^*$, $H \in \mathcal{H}_{\mathrm{h}}$. Then, $u \in \hat{S}_h$ is a minimizer of E_U if and only if*

$$\nabla u = \nabla f + H. \tag{2.27}$$

Proof. According to Lemma 2.3.1 and the definition of E_U in Eqn. (2.22), ∇u is the projection of U onto the space $\mathcal{X}_{\mathrm{h}}^{\mathrm{g}} + \mathcal{H}_{\mathrm{h}}$ in L^2-norm. The theorem follows from the fact that $\mathcal{X}_{\mathrm{h}}^{\mathrm{g}}$, $\mathcal{X}_{\mathrm{h}}^{\mathrm{cg}}$ and $\mathcal{H}_{\mathrm{h}}$ are L^2-perpendicular.

□

This observation provides the key for computing the parameterization u. Instead of minimizing energy E_U the same result is obtained by computing the co-potential part $\mathbf{J}\nabla g^*$ and subtracting it from U. This curl*-free field ∇u is then directly integrated.

2.3.3 Parameterization Space

In this section, we define the proper function space for global parameterizations which is a subspace of $\hat{S}_h$. Since a given $u \in \hat{S}_h$ can be translated by a constant in any triangle without altering $E_U(u)$, there is still too much redundancy. However, according to Theorem 2.3.2, the gradient ∇u of the solution is uniquely determined. Given this gradient field, the solution can be computed up to arbitrary integration constants in each triangle. The solution space has therefore dimension $|T|$ and will now be restricted to a smaller space in order to obtain a unique solution.

There is a natural choice for the integration constants which removes redundancies: Let S be an arbitrary triangle spanning tree, i.e. a maximum spanning tree on the dual of M_h. Starting with the root triangle, integrate ∇u iteratively on all children and choose the integration constants such that u_i fits continuously to the parent triangle. In other words, the gaps w_{ij} are constrained to 0 for all pairs of triangles t_i, t_j in S.

If the surface is simply connected, then the solution u does not depend on the chosen spanning tree (since ∇u is locally integrable). Therefore, all other gaps automatically vanish as well. u would be restricted to the space of conforming finite elements S_h.

This is different on surfaces with arbitrary genus g or with boundary. Before integrating ∇u we have to cut the surface into a topological disk.

Definition 2.3.2 *Let M_h be a triangular surface. A* cut graph *G on M_h is a minimal set of edges of M_h such that $M_h \setminus G$ is simply connected, i.e. if any edge is removed from G, $M_h \setminus G$ would not longer be simply connected.*

Given a cut graph G, we can w.l.o.g. restrict the parameterization space to:

$$\hat{S}_{G,h} := \{u_i \in \hat{S}_h \,|\, \text{for adjacent triangles } t_i,\, t_j \text{ in } M_h \setminus G : w_{u,ij} = 0\}. \tag{2.28}$$

Notice that the minimizer $u_i \in \hat{S}_{G,h}$ of E_U will not depend on the choice of a cut graph up to adding a constant in some triangles which does not change the overall energy value.

Next, we will construct a basis of $\hat{S}_{G,h}$. A cut graph can always be represented by a collection of curves $\{\gamma_k, k \in \{0, \dots, N\}\}$ with $\bigcup_k \gamma_k = G$. Here, the curves run along edges of M_h and are either closed loops or start and end at the surface boundary. A surface of genus g with $b > 0$ boundary components has $2g + b - 1$ may cut paths.

Lemma 2.3.2 *For any $u \in \hat{S}_{G,h}$, the gaps are constant for all edges along any path γ_k. Note that there is one exception if two or more paths partially overlap. In this case the gaps add up on the common part.*

Proof. Let p be a vertex on γ_k (assuming that no other cut path contains p). The vertex star is split into two parts: triangles T_r, T_l on the right and left side of γ_k. In all $t_i \in T_r$, the values $u_i(p)$ are identical since gaps are vanishing on $M_h \setminus G$. The same is true for all $t_i \in T_l$. Thus the gap for the two adjacent edges in γ_k is also similar and is just the difference between both values.

□

It therefore makes sense to define the **gap $w_{u,k}$ of a path** γ_k as the constant translational discontinuity across γ_k. We can represent all $u \in \hat{S}_{G,h}$ uniquely by its coefficients u_i and $w_{u,k}$:

$$u = \sum_{v_i \in V} u_i \Phi_i + \sum_{\gamma_k \in G} w_{u,k} \hat{\Phi}_k, \tag{2.29}$$

where Φ_i are the nodal basis functions from Definition 2.2.2 and $\hat{\Phi}_k$ is defined in a triangle t_j at vertex p_i as (see Fig. 2.7, right):

$$\hat{\Phi}_k(p_i)\,|_{t_j} := \begin{cases} 1, & \text{if } p_i \in \gamma_k \text{ and } t_j \text{ is on the right side of } \gamma_k \\ 0, & else \end{cases} \tag{2.30}$$

Notice that the dimension of $\hat{S}_{G,h}$ is $|V| + 2g$ if M_h is closed, resp. $|V| + 2g + b - 1$ if M_h has $b > 0$ boundary components. Since this is exactly the dimension of $\mathcal{X}_{\mathrm{h}}^{\mathrm{g}} + \mathcal{H}_{\mathrm{h}}$, and since $\nabla \hat{S}_{G,h} = \nabla \hat{S}_h = \mathcal{X}_{\mathrm{h}}^{\mathrm{g}} + \mathcal{H}_{\mathrm{h}}$, the spaces $\nabla \Phi_i$, $\nabla \hat{\Phi}_k$ form a basis of all locally integrable vector fields.

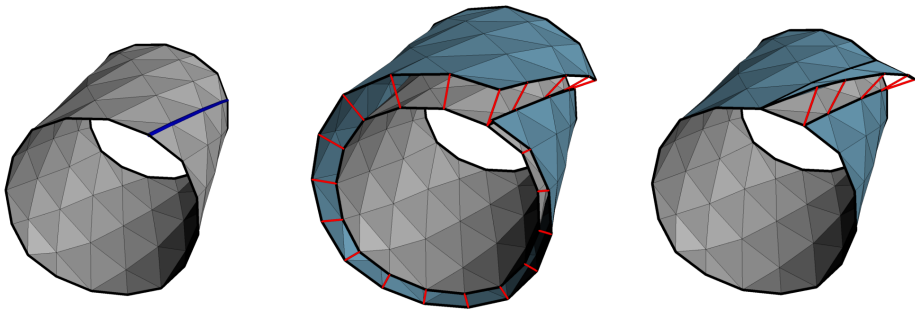


FIGURE 2.7: Left: Geometry with cut graph G consisting of one path γ_0 (blue). Middle: Harmonic function $u \in \hat{S}_{G,h}$ drawn as graph over the surface. Right. Basis function $\hat{\Phi}_0$.

2.3.4 Algorithm

Local Step. Using the knowledge from the previous section, we are now able to give an algorithm for minimizing E_U from Eqn. (2.22). Given a guiding vector field U defined by vectors U_i on triangles t_i, we construct a function $u \in \hat{S}_h$ minimizing E_U. In a first step, the co-potential field $\mathbf{J}\nabla g^* = U - \nabla u$ is computed by projecting U onto the space $\mathcal{X}_{\mathrm{h}}^{\mathrm{cg}}$, i.e. by minimizing the following energy:

$$E_U^*(g^*) := \sum_{t_i \in T} A_i \|\mathbf{J}\nabla g^*_{|t_i} - U_i\|^2. \tag{2.31}$$

Here, $g^* \in S_h^*$ is a scalar non-conforming finite element function, which is linear in each triangle and defined by values on edge midpoints. At midpoints of boundary edges, g^* is fixed to 0. A_i denotes the area of triangle t_i.

The energy can be written as $E_U^*(g^*) = \vec{g}^{*T} L^* \vec{g}^* - 2 \langle\, \vec{g}^*, c_U^* \,\rangle + const.$ Hereby, L^* is the non-conforming cotan-Matrix (see Eqn. (2.21)) of dimension $|E|$, c_U^* is a vector with one entry for each edge e_i: $(c_U^*)_i = \mathrm{curl}^* U(e_i)$, and $\vec{g}^* \in \mathbb{R}^{|E|}$ contains the coordinates of g^* at the edge midpoints. Setting all partial derivatives of E_U^* to 0 leads to the following linear system of equations:

$$L^* \vec{g}^* = c_U^*. \tag{2.32}$$

From the solution $\vec{g}^*$, the function g^*, its gradient ∇g^* and therefore ∇u can directly be computed.

Next, a cut-graph G is computed. The choice of a cut graph is not unique, but in theory the parameterization will not depend on the chosen cut-graph. However, in practice we observe better results when using *shorter* cut graphs, i.e. minimizing the total length of cuts. The reason for this is that we will later use a heuristic to approximate the solution of the NP-hard optimization problem. It will turn out that this heuristic gives a better approximation if the cuts are shorter, see below in the *global step*. Therefore, we use a *shortest cut-graph* as described in the paper from Erickson and Whittlesey [Erickson 2005].

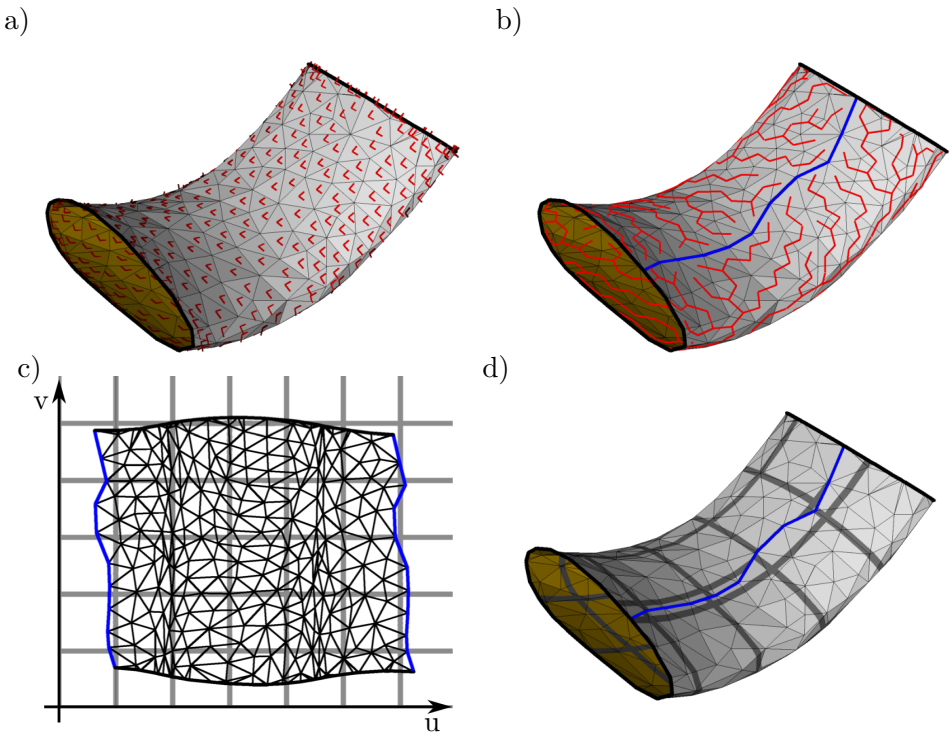


FIGURE 2.8: a) Discrete surface with gradient frame field $(\nabla u, \nabla v)$. b) Cut graph (blue) and spanning tree (red) c) Triangles mapped to texture space via integrated functions (u, v). Notice, that both sides of the cut (blue) match perfectly with a gap of $(5.2, -0.05)$. d) Parameterization of the surface after the local step.

The desired solution u will be computed from its gradient field ∇u as explained in Section 2.3.3: The surface is cut open along a cut graph G. Then a spanning tree on $M_h \setminus G$ is computed and starting with the root triangle, ∇u is iteratively integrated on all children triangles, choosing the integration constant such that the function continuously fits to that of the parents (see Fig. 2.8).

Algorithm 1: Local Step

Input: triangle mesh M_h, guiding fields (U, V)
Output: cut graph G, parameterization $(u, v) \in \hat{S}^2_{G,h}$

1 Solve $L^* \vec{g}^* = d_U$ for $\vec{g}^*$
2 Compute $\nabla u = U - \nabla g^*$
3 Compute arbitrary cut graph $G = \bigcup \gamma_k$
4 Integrate ∇u on $M_h \setminus G$
5 Do the same for function v with input vector field V

Global Step. Up to now, the integer constraint for the gaps, i.e. $w_{u,k} \in \mathbb{Z}$, was ignored, leading to visible seams at the cuts γ_k. The global step enforces the integer constraints by rounding the gaps to the nearest integer. Energy E_U from Eqn. (2.22) is then minimized for the new solution $\tilde{u}$ holding all gaps fix.

Notice, that this greedy integer rounding is a heuristic to find the exact minimum of the energy under integer constraints, which is NP-hard. This is the only part of the algorithm whose result depends on the choice of a cut graph G since each cut introduces two integer constraints. The solution of the heuristic is better the more decoupled the integer constraints are. In practice we observe that the heuristic works very well if the cut paths are short (and therefore the corresponding integer variable has more local influence).

The solution $u \in \hat{S}_{G,h}$ from the local step is represented by values in all triangles at the 3 corner vertices. In the global step, we need the representation from Eqn. (2.29) with coefficients u_i, $w_{u,k}$. For each cut path γ_k, set $w_{u,k} := u_i(p) - u_j(p)$ where t_i is a triangle on the right side of γ_k and t_j the adjacent triangle on the left side.

The gaps $w_{u,k}$ are then snapped to the nearest integer $[w_{u,k}]$, and energy $E_U(\tilde{u})$ is minimized for the remaining unknowns. This is done by solving the linear system of equations:

$$L\vec{\tilde{u}} = d_U, \tag{2.33}$$

where the solution vector $\vec{\tilde{u}}$ contains all coordinates of $\tilde{u}$ at the vertices. L denotes the conforming Laplace matrix from Eqn. (2.20) and the right hand side d_U contains one entry for each vertex p_i: $(d_U)_i = \operatorname{div} U(p_i) - \sum_{\gamma_k} [w_{u,k}] \int_{M_h} \langle\, \nabla\Phi_i, \nabla\hat{\Phi}_k \,\rangle \, \mathbf{d}A$.

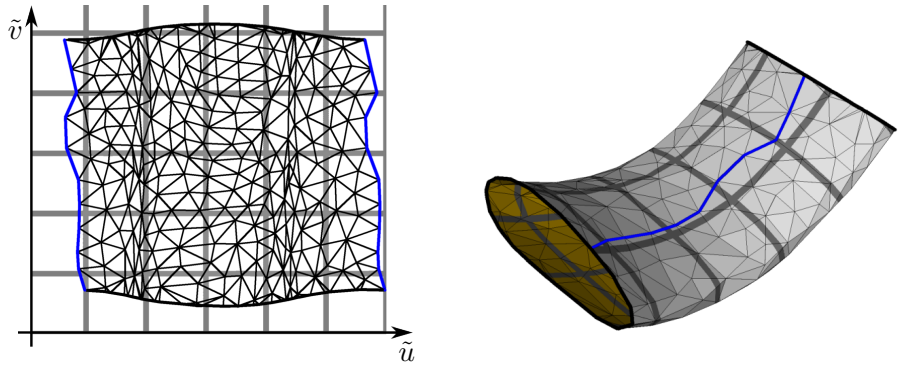


FIGURE 2.9: Surface from Fig. 2.8 with global parameterization. The gap at the blue cut line is snapped to $(5, 0)$.

Note, that the solution $\vec{\tilde{u}}$ is in general very similar to $\vec{u}$. Thus, in practice the computing time is reduced by solving $L\vec{x} = d_U - L\vec{u}$ for $\vec{x}$ and setting $\vec{\tilde{u}} = \vec{u} + \vec{x}$.

Algorithm 2: Global Step

Input: cut graph $G = \{\gamma_k\}$, parameterization $(u, v) \in \hat{S}^2_{G,h}$

Output: globally coherent parameterization $(\tilde{u}, \tilde{v}) \in \hat{S}^2_{G,h}$

1 Compute gaps $w_{u,k}$ of u for all paths γ_k
2 Round $w_{u,k}$ to nearest integer
3 Solve Eqn. (2.33) for $\vec{\tilde{u}}$
4 Do the same for v and $\tilde{v}$

2.4 Full QuadCover

The previous Section 2.3 introduced the simplified version of QuadCover. Any given planar texture image which tiles the two-dimensional plane is mapped continuously onto the surface. The full version of QuadCover extends this setting to texture images with additional 90 degrees rotational symmetry. The unit grid texture is for example invariant under rotations of 90 degrees around any corner or midpoint of a quad.

Using a texture image with rotational symmetry can be exploited to enlarge the space of parameterizations significantly. A parameter map needs only to be described "up to" rotation, i.e. u- and v-parameter lines cannot be distinguished anymore. Consequently, a frame (U_i, V_i) defined on a triangle t_i has no unique representation, since it is equivalent to $(V_i, -U_i)$, $(-U_i, -V_i)$ and $(-V_i, U_i)$ (see Fig. 2.10).

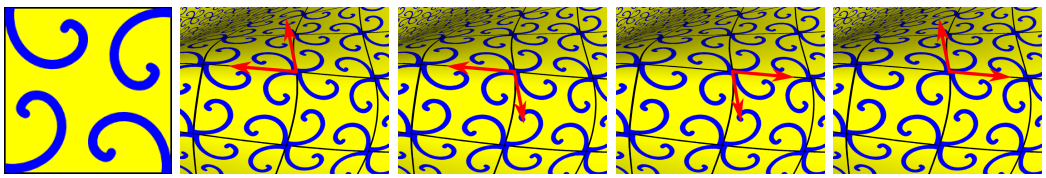


Figure 2.10: Rotational symmetric texture image (left) which is mapped by a parameterization onto a surface. The right four images show the different equivalent interpretations of the gradient frames (U_i, V_i).

Therefore, the input guiding field for the full QuadCover algorithm is more general, e.g. we are now able to use a curvature field whose vectors point in principal curvature directions. In general this is not possible using vector fields.

This section introduces the notion of frame fields as being vector frames divided by rotational symmetry. Equivalently, a frame field can be seen as a multi-valued vector field on the surface. Unfortunately, it is not clear how multi-valued functions should be handled by QuadCover and how to apply standard vector field calculus, e.g. the Hodge-Helmholtz decomposition.

One way of dealing with multi-valued functions goes back to the research of Bernhard Riemann. Such function can be described as a single-valued function on a Riemann Surface. We are using this idea to establish the correct calculus for frame fields. Section 2.4.1 gives an introduction into Riemann Surfaces. Section 2.4.2 and 2.4.3 explain how parameterizations with rotational discontinuities are related to Riemann surfaces and therefore provide a theoretical justification for the approach.

Parameterizations with rotational discontinuities imply singularities of fractional index which correspond to branch points of the Riemann surface. They are discussed in Section 2.4.4. Finally, Section 2.4.5 describes the full QuadCover algorithm. The reader merely interested in the algorithm can directly go to Section 2.4.4, skipping

the theory behind the algorithm.

2.4.1 Covering Spaces

Riemann Surfaces and Covering Spaces. Riemann surfaces are a fundamental concept in modern complex analysis, topology and algebraic geometry. They were first studied by B. Riemann in his dissertation at Göttingen [Riemann 1851]. Half a century later, F. Klein and H. Weyl emphasized that Riemann surfaces are an indispensable component, and even the foundation, of the theory of analytic functions. Since then Riemann surfaces serve as generalized domains for complex functions because multi-valued complex functions can be turned into single-valued functions when defined on such a surface instead of the complex plane.

We review the basic notions of Riemann surfaces and ramified covering maps from algebraic topology. This naturally leads us to the notion of path lifting and the deck transformation group which are used as a basic concept in the setting of QuadCover. A good overview about the general theory is given e.g. in [Farkas 1980, Lamotke 2005, Forster 1999, Needham 2000].

Definition 2.4.1 (Riemann Surface) *A* Riemann Surface *is a Hausdorff space together with a holomorphic structure, i.e. an atlas of charts* $\{(U_i, h_i) \,|\, h_i : U_i \to \mathbb{C}\}$ *whose transition maps* $h_j \circ h_i^{-1}$ *are* biholomorphic *complex functions.*

Here, we consider only one-dimensional Riemann Surfaces which are in fact two-dimensional real manifolds equipped with a complex structure. The concept of Riemann surfaces is closely related to branched covering spaces.

Definition 2.4.2 (Branched Covering) *Let* M *be an orientable two-dimensional manifold. A* branched covering manifold M' *of* M *provides a continuous surjective map* $\pi : M' \to M$ *such that each point* $p \in M$ *has a neighbourhood* $U \subset M$ *with the following properties:*

1. *The preimage* $\pi^{-1}(U)$ *is a union of countably many disks* V_k *(called* sheets *or* layers*).* $p_k = \pi^{-1}(p) \cap V_k$ *denotes the point in sheet* V_k *which gets mapped onto* p*.*

2. *For all the sets* U*,* V_k*, there are local coordinates* $z : U \to \mathbb{C}$*,* $z(p) = 0$ *and* $w_k : V_k \to \mathbb{C}$*,* $w_k(p_k) = 0$*, such that* π *is given by the local representation* $w_k = z^{n(p_k)}$ *with* $n(p_k) > 0$ *being a constant integer. If* $n > 1$*, then* p_k *is called a* branch point *or* ramification point *of index* $n(p_k)$*.*

Away from branch points, a covering surface locally consists of multiple copies which are mapped homeomorphically onto the domain surface M. Globally, the different layers are globally connected forming a smooth manifold (e.g. as in Fig. 2.11, left).

At a branch point, several layers are glued together forming a helical winding structure, like a spiral staircase (Fig. 2.11, middle). Against a first intuition, the vicinity of a branch point is indeed a topological disk. The covering surface in Fig. 2.11, right is realized in a way that the covering map π is projecting along the surface normal. In general, embeddings in $\mathbb{R}^3$ are not always possible without having self-intersection.

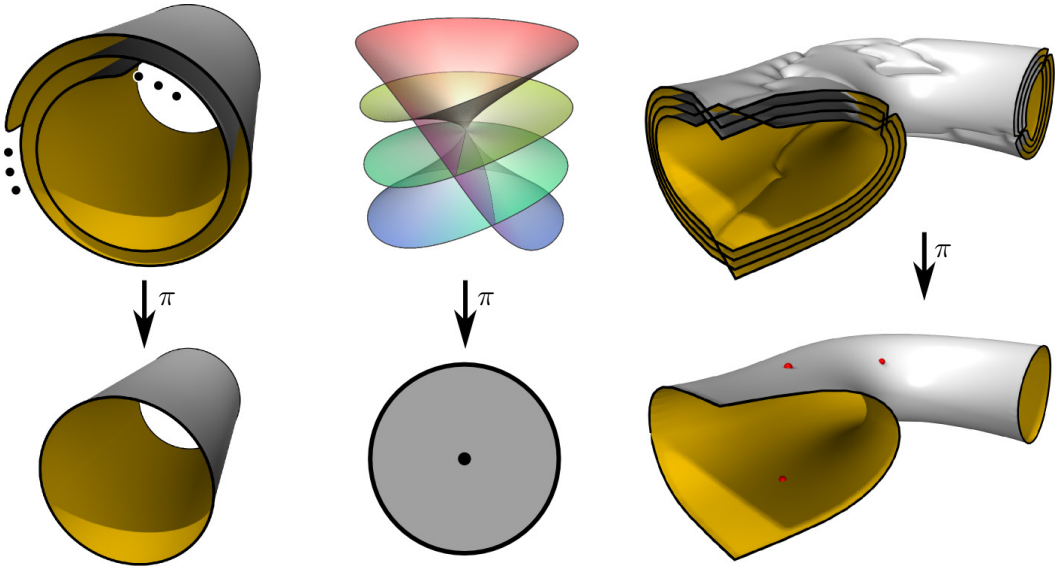


FIGURE 2.11: Examples of covering spaces. Left: Covering of a cylinder with infinitely many layers. Middle: 4-sheeted covering of a disk with branch point (π projects along the height axis). Right: 4-sheeted covering of a surface with branch points (red).

Definition 2.4.3 *The preimage $\pi^{-1}(p)$ of a point $p \in M$ is called the **fibre** of p. It is shown that if the fibre is finite then the sum $gr(\pi) := \sum_{q \in \pi^{-1}(p)} n(q)$ is independent of the choice of p and is called the **grade** of π. Away from branch points, the grade is simply the number of layers over each point.*

The following result is remarkable since it reduces every non-constant holomorphic map between Riemann surfaces to a very simple local representation.

Theorem 2.4.1 *Every non-constant holomorphic map between Riemann surfaces is a branched covering map.*

Using this theorem, the theory of multi-valued complex functions can be described by the concept of single-valued functions on a covering space. The domain of a multi-valued function is replaced by a Riemann surface which "covers" the image surface and the function lifts to a single-valued function on the covering (see Fig. 2.12).

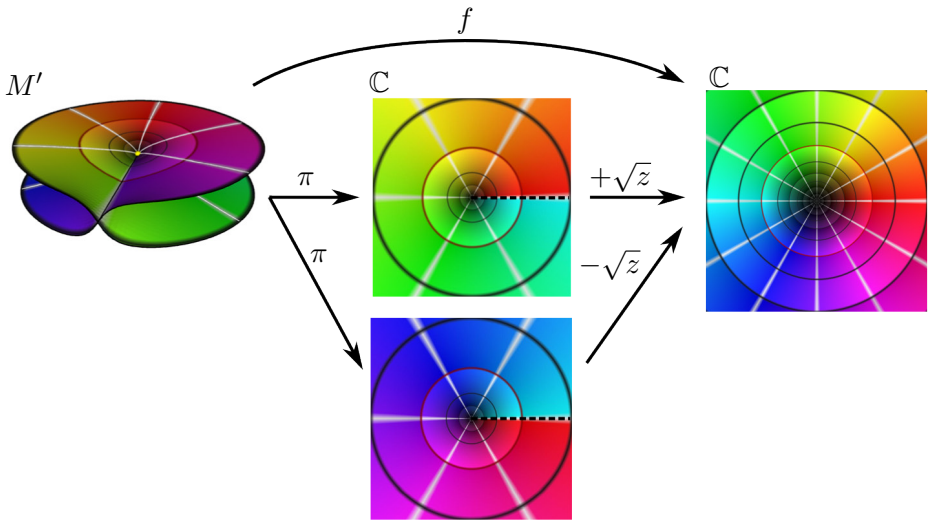


FIGURE 2.12: Representation of a multi-valued function $\pm\sqrt{z}$ using covering spaces. The image space (right) is colored by some radial texture image which is mapped back onto the domain (middle) of the function. Notice that the two branches $+\sqrt{z}$ and $-\sqrt{z}$ are discontinuous across the positive real axis. With using a two-sheeted covering space (M', π) (left) of the complex plane containing a branch point at 0, the function lifts to a single-valued map f on M'.

Genus of Covering. In the next section, we will construct an n-sheeted covering M' of the input surface M. For the mathematical theory, we are interested in the genus of M'. If one thinks of a triangulated surface and branch points are lying on vertices, then an n-sheeted covering comes with a natural triangulation consisting of n copies of all triangles, edges and vertices, except at the branch points where some vertices are identified. The genus of M' can now be computed using the Euler formula which leads to the following relation:

Theorem 2.4.2 (Riemann-Hurwitz) *Let M be a closed surface and M' a connected k-sheeted covering of M having branch points b_i with ramification indices $n(b_i)$. Then the genus g' of M' is related to the genus g of M by:*

$$g' = k(g-1) + 1 + \frac{1}{2}\sum_{b_i}(n(b_i) - 1) \tag{2.34}$$

Consider a k-sheeted covering of a genus 0 surface. It is impossible to have only one branch point, since then g' would be negative. If there are two branch points b_i of index $n(b_i) = k$, the covering is homotopic to another sphere. Each additional pair of branch points of index k increases the genus by $(k-1)$ and hence enlarges the fundamental group of the covering.

Monodromy Action and Deck Transformation Group. From now on, all coverings are assumed to be connected, locally path connected and semi-locally simply connected, since these properties are required for the following propositions.

Given a covering M' with the above properties, a root point $p' \in M'$, $p := \pi(p')$ and a closed curve γ on M starting in p. Then, there is a unique *lift* γ' of γ onto the covering, i.e. $\pi(\gamma') = \gamma$, which starts in p'. The end point of this lifted path is not necessary equal to p', but is some point in the same fibre. It turns out that the end point depends only on the homotopy class of γ. Each element in $\pi_1(M, p)$ therefore is an *action* on M' which maps p' to a point in its fibre. These actions form a group, called the *monodromy group*, which is isomorphic to $\pi_1(M, p)/\pi_1(M', p')$.

Definition 2.4.4 (Deck transformation group) *A **deck transformation** is a homeomorphism $h : M' \to M'$ which leaves all fibres invariant ($\pi \circ h = \pi$). The set of deck transformations form a group under composition, the* deck transformation group *$Aut(\pi)$.*

A deck transformation actually permutes the elements in the fibre of all points $p \in M$.

From now on, we restrict our theory to *transitive* (or *normal*) coverings provided that whenever $\pi(p'_1) = \pi(p'_2)$, there is a unique deck transformation $f \in Aut(M_h)$ with $f(p_1) = p_2$. For normal coverings, the action of a deck transformation $f \in Aut(\pi)$ coincides with a monodromy action, i.e. the deck group is isomorphic to the monodromy group and therefore also to $\pi_1(M, p)/\pi_1(M', p')$.

2.4.2 Parameterization on a Covering

We use the concept of covering surfaces to formulate the notion for global parameterizations with rotational discontinuities. Hereby, the domain surface M_h is replaced by a 4-sheeted covering $M'_h, \pi : M'_h \to M_h$. Let (X, r_{ij}) be the guiding frame field as in Definition 2.1.3. It is then lifted to two vector fields (U', V') on M'_h and the parameterization problem is formulated using standard vector field notion similar to as in Section 2.3. The final parameterization is then projected back onto M_h. Notice, that this is only a theoretical construct, M' is never computed in practice.

Let (U_i, V_i) be the vector frame of X in all triangles $t_i \in T$ and r_{ij} be the matchings between triangles. The construction of the covering consists of 3 steps (see Fig. 2.13).

1. **Generate trivial covering.** For each triangle t_i, construct a *trivial* 4-sheeted covering which consists of four copies t'_{ik}, $k \in \{0, 1, 2, 3\}$ of t_i.

2. **Lift input frames to covering.** For each triangle t_i, define frames on the covering as vector fields $(U'_{i,k}, V'_{i,k})$ which are constant on the sheets t'_{ik}. They are defined by the formal matrix vector multiplication:

$$\begin{pmatrix} U'_{i,k} \\ V'_{i,k} \end{pmatrix} := \mathbf{J}^k \begin{pmatrix} U_i \\ V_i \end{pmatrix}.$$

The results are two (constant) vector fields on all trivial coverings.

3. **Glue sheets together.** For each triangle t_i, connect the triangles in its fibre combinatorially to the fibre of adjacent triangles t_j, such that corresponding frames are linked; triangle t'_{ik} is glued to $t'_{j,(k-r_{ij}) \bmod 4}$. Notice that triangles are glued in a way that maintains the cyclical order of the triangles in adjacent fibres.

The result is a polyhedral surface M'_h which is a normal covering of M_h. Its realization has self-overfoldings (all triangles in the same fibre are geometrically exactly identical), however combinatorially, it is a polyhedral surface. Each triangle in M'_h is connected to three neighbouring triangles (except at the boundary) and the local neighbourhood of each vertex is a topological disk.

One might think of the 4-sheeted covering surface as a warehouse with four floors. Between the shops (the transition between triangles), there are staircases which connect each floor with the next one given by r_{ij} cyclically. The matchings prescribe if there is a staircase between two triangles ($r_{ij} \neq 0$) and describes how many floors the stairs go upwards.

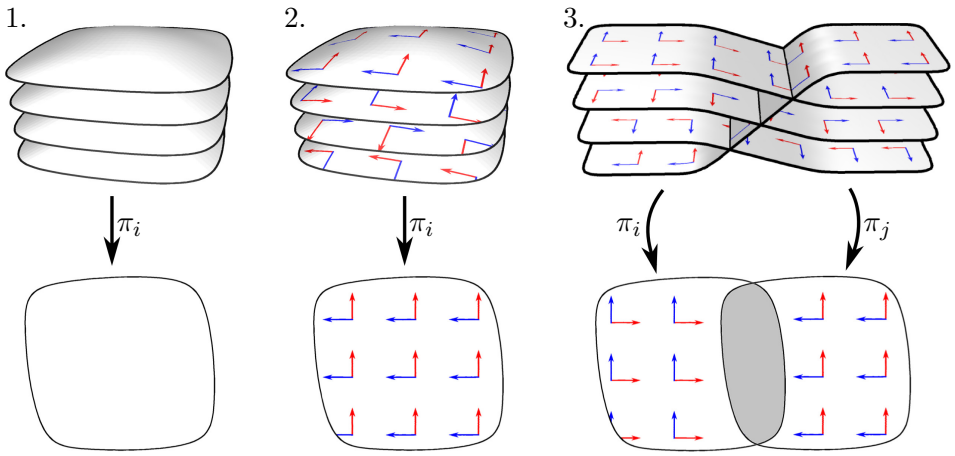


FIGURE 2.13: Theoretical concept of constructing a 4-sheeted covering. 1. Trivial covering over each chart (or triangle). 2. Lifted frame field on covering. 3. Layers of adjacent charts, glued together (here, the matching is $r_{ij} = 3$).

In the second step, the four possible interpretations of the alignment of the frames are lifted to the four layers over each triangle in a cyclically ordered way. Since adjacent sheets are glued in a similar cyclically ordered way in the third step, corresponding frames are adjacent on M'_h. Although the frames (U, V) on M_h are continuous only up to applying matchings, on M'_h they decouple into two separate vector fields (U', V').

The lifted frame field can now be used to parameterize the covering surface similarly as described in Section 2.3. It is thereby very important that the cyclic order of the layers is maintained by the parameterization:

Definition 2.4.5 *A **global parameterization on a covering** M'_h is a map $\varphi' \in \hat{S}_h(M'_h)^2$ with the following property:*

Let $p'_k \in M'_h$, $k = \{0,1,2,3\}$ be the four cyclically ordered points in one fibre and $p'_4 := p'_0$. Then,

$$\varphi'(p'_k) = \mathbf{J}\varphi'(p'_{k+1}).$$

The property of this definition enforces a symmetric behaviour of a parameterization on all layers, such that the projection from different layers back onto M_h is congruent (see Fig. 2.14).

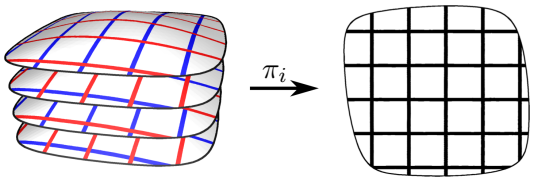


FIGURE 2.14: Global parameterization on a covering is projected down on the original surface. The resulting grid does not depend on which layer is used for projection.

2.4.3 Parameterization on the Domain Surface

In practice, constructing a covering as explained in Section 2.4.2 and using it as domain is inconvenient. In QUADCOVER, the parameterization is still computed on the original surface, but the parameterization on the covering is implicitly defined.

Theorem 2.4.3 *Let M_h be a polyhedral surface, (M'_h, π) be a covering arising from the construction in Section 2.4.2 and φ' be a global parameterization on M'. Then, the deck transformation group $Aut(\pi)$ contains exactly 4 transformations, and each $f \in Aut(\pi)$ corresponds to a global rotation of φ' by $r \cdot \frac{\pi}{2}, r \in \{0,1,2,3\}$, i.e.:*

$$\varphi'(f(p)) = \mathbf{J}^r \varphi'(p), \quad \forall p \in M'_h. \tag{2.35}$$

Proof. Since M'_h is a normal covering, the deck transformations $f \in Aut(\pi)$ are uniquely determined by the image of one point in M'_h (see Section 2.4.1). Therefore, there are exactly four deck transformations, mapping an arbitrary point to all four points in its fibre.

Given a deck transformation $f \in Aut(\pi)$ and a point $p \in M_h$, enumerate the points in fibre $\pi^{-1}(p)$ by p'_0, p'_1, p'_2, p'_3 according to the enumeration of layers in step 1 of the construction of M'_h. Since $Aut(\pi)$ is equivalent to the monodromy group (see Section 2.4.1), there is a closed loop $\gamma : [0,1] \to M_h$, $\gamma(0) = \gamma(1) = p$ which can be lifted to a path $\gamma'_0 : [0,1] \to M'_h$ with $\gamma'_0 = p'_0$ and $\gamma'_0(1) = f(p'_0)$. The lifted curve γ'_0 ends in the same fibre, i.e. $f(p'_0) = p'_{r(p)}$, $r(p) \in \{0,1,2,3\}$.

By applying the monodromy action, we can evaluate the image $f(p_k)$ of all p_k, $k \in \{0,1,2,3\}$ by lifting the same curve γ with using p'_k as start point. Since the cyclic ordering of layers is maintained in step 3 of the construction of M'_h, the lifted curve γ'_k is similar to γ'_0, but shifted by k layers. We obtain the following equation for all points in the fibre of p:

$$f(p'_k) = p'_{k+r(p)}, \quad k \in \{0,1,2,3\}. \tag{2.36}$$

Eqn. (2.36) is also true for all other points $p \in M$. resulting in a continuous function $r(p)$. Since $r(p)$ takes only discrete values, it must be constant everywhere. Therefore the deck transformations globally shifts M_h by $r(p)$ layers. The statement follows from Definition 2.4.5.

□

The theorem enables us to represent a global parameterization φ' on M'_h by a function φ on M_h. In each triangle t_i of M_h, choose one *reference triangle* t'_{ik} from the fibre over t_i and set $\varphi_{|t_i}$ to $\varphi'_{|t'_{ik}}$. For adjacent triangles t_i, t_j, the corresponding reference triangles must not necessarily be connected in M'_h. There is always a unique Deck transform $f \in Aut(M'_h)$ which maps the reference triangle of t_i to that from t_j (on the common edge) which corresponds to a rotation of φ' by some amount r_{ij}. This number r_{ij} is associated to the edge. The values φ together with all r_{ij} are enough to reconstruct the parameterization on the covering.

The necessary data (φ and r_{ij}) is similar to the earlier mentioned setting with a parameterization φ on M_h which has rotational discontinuities (matchings) at edges (see Definition 2.1.1). But now, we have established a different interpretation of the data: the matching between two triangles represent the Deck transformation which relates the reference triangles of the covering where the parameterization is described as a function $\varphi' \in \hat{S}_h(M'_h)^2$. The advantage of this interpretation is that the rotational discontinuities are eliminated and the frame field decouples into two vector fields. The problem is therefore reduced to the simplified QuadCover setting, but with a different underlying surface.

It remains to notice that the QuadCover algorithm and the results are independent of the choice of reference triangles. If a different reference triangle is chosen, then the matchings will change, but this has no effect on the topology of the covering, the position or index of branch points (see Section 2.4.4). The covering surface is uniquely determined (up to enumeration of the layers) by knowing its branch points.

2.4.4 Branch Points

In vector field theory, the topology of a given field is mainly described by its singular points, e.g. the location of sinks, sources and vortices. A singularity p is

categorized by its index which measures the angle deficit when tracing a vector on an infinitesimally small loop γ around p, divided by 2π. The index of a vector field singularity is always an integer number.

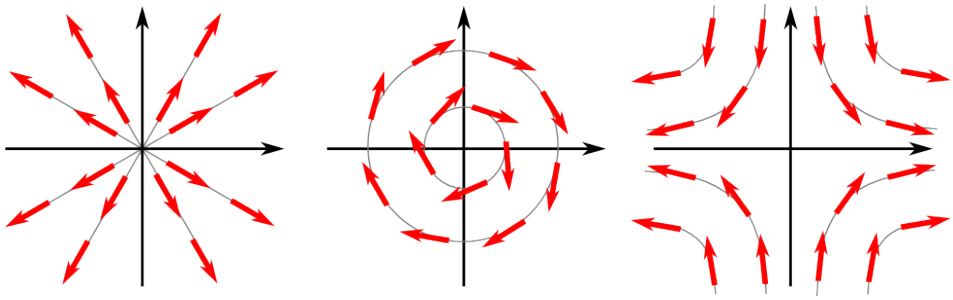


FIGURE 2.15: Vector fields with singularity in $(0,0)$. Left: Source, index=1. Middle: Vortex, index = 1. Right: Saddle, index = -1.

Singularities of frame fields behave differently since frame vectors are allowed to flip. Thus, when tracing a vector along a loop around p, one may end up with a rotated version of the vector, see Fig. 2.16. The index of a frame field singularity is therefore an integer multiple of $1/4$.

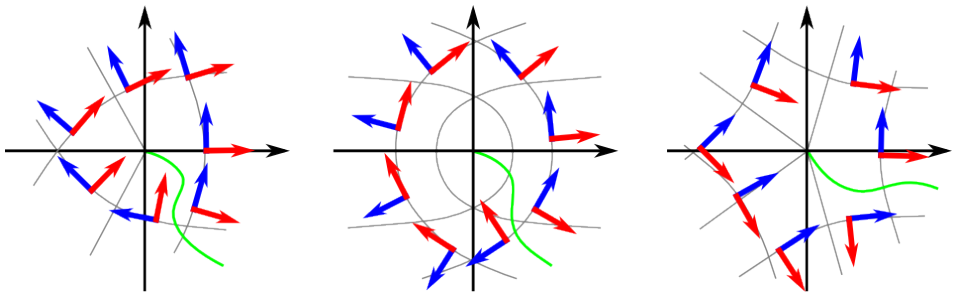


FIGURE 2.16: frame fields with singularity in $(0,0)$. The representation frame changes across the green curve (matching is $\neq 0$ there). Left: index=1/4. Middle: index = 1/2. Right: index = -1/4.

Let p be a singularity of the frame field $X = (U, V)$, e.g. with an index of $1/4$. When the vector U is traced along a loop γ once around p, it matches V. Therefore, if γ is lifted to a curve γ' on the covering surface with $\pi(\gamma') = \gamma$, it is not a closed loop: moving around the singularity once ends on the next top layer. The point p is therefore a branch point where the covering has the local structure of a helix (as in Fig. 2.11, middle).

In the discrete setting, branch points always occur at vertices. Despite the fact that the index is a geometric property of the frame field in the vicinity of a vertex, the fractional part of the index can be estimated only by knowing the matchings:

Definition 2.4.6 *Let p be a vertex with ordered triangles $t_0, \ldots, t_k$ in its vertex*

star. The ***fractional index*** *of p is defined by:*

$$ind_f(p) = \frac{1}{4}\left(\sum_{i=0}^{k} r_{i,i+1} mod\ 4\right). \tag{2.37}$$

p corresponds to a branch point in the covering M_h' if and only if the fractional index is not 0. The ramification index is $n(p) = 4$ if $\mathrm{ind}_f(p) \in \{1, 3\}$ and $n(p) = 2$ if $\mathrm{ind}_f(p) = 2$. The fractional index is therefore a property of the covering surface.

Notice that the fractional index is equivalent to the fractional part of the classical vector field index. The integer part is determined by the index of the vector field on M'.

It finally turns out that branch points are introducing restrictions to parameterizations:

Theorem 2.4.4 *Given a global parameterization φ and a branchpoint $p \in M_h$. Then, the value $\varphi(p)$ is uniquely determined by the gaps at adjacent edges to p.*

Proof. Let γ be a small positively oriented loop around p which passes through the triangles $t_0, \ldots, t_k, t_0$. Then, the values $\varphi(p)$ in different triangles are related by the transition (see Eqn. (2.2)):

$$\begin{aligned} \varphi_0(p) &= \mathbf{J}^{r_{01}}\varphi_1(p) + w_{01}, \\ \varphi_1(p) &= \mathbf{J}^{r_{12}}\varphi_2(p) + w_{12}, \\ &\ldots \end{aligned} \tag{2.38}$$

Plugging in each equation in the successor yields:

$$\varphi_0(p) = J^{4\cdot \mathrm{ind}_f(p)}\varphi_0(p) + w, \tag{2.39}$$

with some $w \in \mathbb{Z}^2$ depending on the gaps at intermediate edges. If p is regular, then w must be 0. Otherwise, the parameterization is given by:

$$\varphi_0(p) = (\mathrm{Id} - J^{4\cdot \mathrm{ind}_f(p)})^{-1} w. \tag{2.40}$$

□

Notice that eqn. (2.40) gives a direct relation between the gaps w_{ij} and the image of the parameterization at singularities. Inparticular, by computing the inverse matrix it becomes clear that all singularities are mapped to $\frac{1}{2}\mathbb{Z}^2$ if all $w_{ij} \in \mathbb{Z}^2$.

2.4.5 Algorithm

In this section, we will formulate the full QuadCover algorithm. The principles are the same as in the simplified QuadCover, but evaluated on the covering surface.

Representation of a Parameterization. Given M_h, a parameterization is represented by its values φ at vertices (in all adjacent triangles) together with matchings r_{ij} – which implicitly define a covering surface M'_h – and gaps w_{ij} for all edges.

As in the simplified version of QuadCover, there is much redundancy in the choice of gaps. In a similar way, we restrict all gaps in the interiour of a simply connected disk D to 0. Singularities can be considered as infinitesimally small holes and must lie on the boundary of D. To achieve this, we have modified the method by Erickson and Whittlesey [Erickson 2005] for surfaces with boundary and singularities:

The surface M_h is cut open by a cut-graph G such that $D = M_h \setminus G$ is simply connected. For closed surfaces of genus g, the method [Erickson 2005] computes a system of $2g$ many shortest loops, whose union is a cut graph. Once we have more than one boundary component (or singularity), each additional boundary component needs one path to connect it with G. Thus, in presence of $b > 1$ boundary components (or branch points) we need $2g + b - 1$ paths in total. In our extension of the method we combinatorially identify all boundary vertices and branch points into one root point R. On this surface (now without any boundary), we apply the method of [Erickson 2005] with B as the base point. When we undo the identification of boundary points, the paths which looped through R now turn into paths that connect boundary components and branch points.

Notice, that the cut graph is computed on M_h, thus it cuts the covering M'_h into 4 simply connected pieces – however this is no problem, the cut surface may contain more than one component.

Once, G is computed, gaps at edges in $M_h \setminus G$ are set to 0. Denote the cut paths by γ_k, $k \in \{0, \dots, 2g + b - 2\}$. Similar to as in Section 2.3.3, we can represent each global parameterization by coefficients $u_i, v_i \in \mathbb{R}$, $w_{u,k}, w_{v,k} \in \mathbb{Z}$ such that

$$\varphi = \sum_{v_i \in V \setminus B} \left(u_i \Phi'_{2i} + v_i \Phi'_{2i+1} \right) + \sum_{\gamma_k \in G} \left(w_{u,k} \hat{\Phi}'_{2k} + w_{v,k} \hat{\Phi}'_{2k+1} \right), \tag{2.41}$$

where B is the set of branch points. Φ'_{2i}, Φ'_{2i+1} are the hat functions for each regular vertex p. They are the 2D equivalent of the nodal hat functions Φ_i: In one adjacent triangle to p, set $\Phi'_{2i}(p) = \left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right)$ (resp. $\Phi_{2i+1}(p) = \left(\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right)$). In all other triangles of the vertex star, the values of the basis functions are rotated according to the given matchings such that it turns into a continuous function on the covering. $\hat{\Phi}'_{2k}$ and $\hat{\Phi}'_{2k+1}$ are path-based basis functions with a constant gap along path γ_k. They are defined similarly to Eqn. (2.30) with a constant value of $\hat{\Phi}'_{2k} = \left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right)$ (resp. $\hat{\Phi}'_{2k+1} = \left(\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right)$) at the right side of path γ_k. If the path crosses edges with non-vanishing matching, the value must be rotated accordingly. Furthermore, if γ_k starts or ends in a singularity, then its value in this singularity is uniquely determined by Eqn. (2.40). See Fig. 2.17 for an illustration of the basis functions.

Local Step. For the local step, a non-conforming function g^* is computed which is given by its values on edge midpoints. It is represented by coefficients $g^*_{u,i}, g^*_{v,i}$

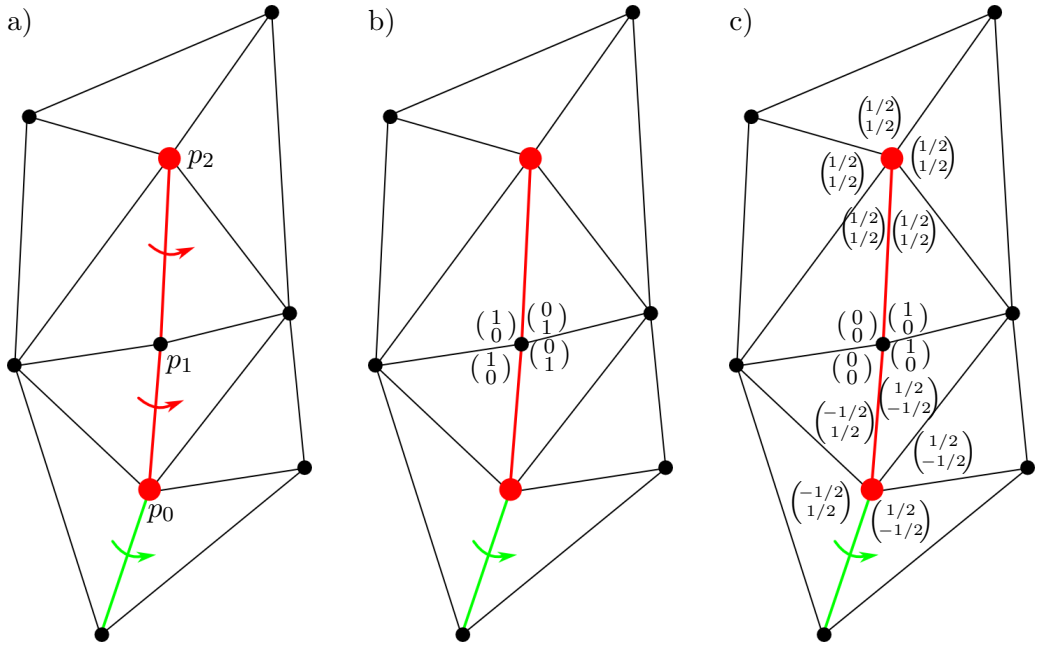


FIGURE 2.17: a) Triangle mesh with two singularities p_0 and p_2. In direction of the drawn arrows, the matching is 1 on red edges, 2 on the green edge and 0 on black edges. b) Values of the nodal basis function Φ'_2 on p_1. It vanishes on all other vertices. c) Values of the path basis function $\hat{\Phi}'_0$ for a cut path γ_0 formed by the vertices (p_0, p_1, p_2). The gap is $\binom{1}{0}$ at the edges of γ_0.

with:

$$g^* = \sum_{\text{edges } e_i} \left(g^*_{u,i} \Psi'_{2i} + g^*_{v,i} \Psi'_{2i+1} \right).$$

Here, Ψ'_{2i}, $\Psi'_{2i+1} : M_h \to \mathbb{R}^2$ denote the non-conforming basis functions on the covering. They are $(1,0)$ (resp. $(0,1)$) at the edge e_i in one adjacent triangle t_j and $\mathbf{J}^{r_{jk}}(1,0)$ (resp. $\mathbf{J}^{r_{jk}}(0,1)$) in the opposite triangle t_k. At all other edge midpoints, the basis functions are $(0,0)$.

Given a frame field X as input, the local step consists of solving the linear system of equations:

$$\mathfrak{L}^* \vec{g}^* = \mathfrak{c}^*_X. \tag{2.42}$$

The entries of the Laplace matrix are: $\mathfrak{L}^*_{ij} = \int_{M_h} \langle \nabla\Psi'_i, \nabla\Psi'_j \rangle \, \mathbf{d}A$, and those of the right vector are: $(\mathfrak{c}^*_X)_i = \int_{M_h} \langle \nabla\Psi'_i, X \rangle \, \mathbf{d}A$. Basically, the matrix is almost similar to the non-conforming Laplace matrix L^* from Eqn. (2.21), but each entry is replaced by a 2x2-matrix which incorporates the rotation. The right vector contains the curl* of the input field X.

Algorithm 3: Local Step

Input: triangle mesh M_h, guiding field $X = (U_i, V_i, r_{ij})$
Output: cut graph G, parameterization $(u, v) \in \hat{S}^2_{G,h}$

1 Solve Eqn. (2.42) for $\vec{g}^*$
2 Compute $(\nabla u, \nabla v) = (U_i, V_i) - \nabla g^*$
3 Compute arbitrary cut graph $G = \bigcup \gamma_i$
4 Integrate ∇u, ∇v on $M_h \setminus G$ respecting the matchings

Global Step. The last stage of QuadCover assures the given integer constraints. It is similar to the global step of the simplified QuadCover but on the covering, i.e. with representation of a parameterization as in Eqn. (2.41).

The gaps $w_{u,k}, w_{v,k}$ are measured for each path γ_k and snapped to its nearest integer. The global step then consists of solving the system:

$$\mathfrak{L}\vec{\tilde{u}} = \mathfrak{d}_X \tag{2.43}$$

The conforming 2d Laplace matrix has entries: $\mathfrak{L}_{ij} = \int_{M_h} \langle\, \nabla\Phi'_i, \nabla\Phi'_j \,\rangle \,\mathbf{d}A$. The entries of the divergence vector are:

$$\begin{aligned}(\mathfrak{d}_X)_i = &\int_{M_h} \left\langle \nabla\Phi'_i, X \right\rangle \mathbf{d}A \\ &- \sum_{\gamma_k \in G} \left([w_{u,k}] \int_{M_h} \left\langle \nabla\hat{\Phi}'_{2k}, \nabla\Phi'_i \right\rangle \mathbf{d}A + [w_{v,k}] \int_{M_h} \left\langle \nabla\hat{\Phi}'_{2k+1}, \nabla\Phi'_i \right\rangle \mathbf{d}A \right).\end{aligned} \tag{2.44}$$

$\vec{\tilde{u}}$ is the solution vector containing the coefficients (u_i, v_i) from Eqn. (2.41).

Algorithm 4: Global Step

Input: parameterization $(u, v) \in \hat{S}^2_{G,h}$
Output: globally coherent parameterization $(\tilde{u}, \tilde{v}) \in \hat{S}^2_{G,h}$

1 Compute gaps $w_{u,k}, w_{v,k}$ of (u, v) at paths γ_k
2 Round gaps to nearest integer
3 Solve Eqn. (2.43) for $\vec{\tilde{u}}$

Assure a Pure Quadrilateral Mesh. Notice that due to the construction of the basis functions $\hat{\Phi}_k$, the value of each component in a global parameterization at a singularity is always an integer multiple of 1/2 (see Eqn. (2.40)). If the parameterization is used for quad remeshing, then a singularity lies either exactly on a corner of the quad grid, on an edge midpoint (which is only possible if the fractional index is 1/2) or on the midpoint of a quadrilateral, turning the quad into an n-gon, $n \neq 4$.

Many applications require a parameterization whose grid of parameter lines consists of quadrilaterals only. All elements in the parameterization are quads if the singularities are on integer positions, i.e. we just have to enforce that $w_{u,k}, w_{v,k} \in 2\mathbb{Z}$. Therefore, instead of rounding the gaps to the nearest integer, they are rounded to the nearest point in $2\mathbb{Z}$.

An equivalent way to enforce an all-quadrilateral mesh is to use QuadCover as described above, but with all input frames scaled by 0.5. The result is a parameterization which is twice as coarse. A subsequent scaling of the parameterization by 2 (which actually corresponds to a $1:4$ subdivision of the mesh) leads to the same result where all gaps are in $2\mathbb{Z}$.

Chapter 3

Frame Field Generation

The quality of a parameterization produced with QuadCover depends heavily on the input frame field. It describes the desired alignment of the parameter lines. For a given triangle mesh M_h, a frame field consists of two vectors in the tangent plane of each triangle plus the matchings $r_{ij} \in \mathbb{Z}$ between all adjacent triangles t_i, t_j (see Definition 2.1.3). The matchings relate adjacent vector frames and encode the location and fractional index of singularities.

The generation of good frame fields is essential. The requirements on a "good" frame field actually depends on the application. Common demands are:

Minimal deviation from orthonormal coordinate frames in order to minimize metric distortion of the parameterization.

Minimal curl since the co-gradient part is subtracted in the local stage of QuadCover. If curl is low, then the original frame directions will be maintained in the parameterization.

Alignment to visually appealing directions. This is desired especially if the parameterization is used for design purposes.

Thoughtful singularity placement. Most commonly, distortion in the parameterization occurs in the vicinity of singularities, especially if those are placed in adverse locations.

There exist several methods for frame field generation (see Section 1.5). In this section, we present a frame field design framework which is based on ideas of different techniques [Kälberer 2007, Bommes 2009, Nieser 2011a]. The generated frame fields are optimized for generating parameterizations with low distortion.

3.1 Principal Curvature Fields

The input to field-based parameterization algorithms are commonly frame fields whose vectors point in the principal curvature directions. They are the naturally recognized directions on curved surfaces and align to surface features and to sharp edges. Furthermore, the quadrilaterals of meshes whose edges are aligned in principal curvature direction tend to be more planar.

Algorithm 5: Generate Principal Curvature Field

Input: triangle mesh M_h

Output: guiding fields (U, V), matchings r_{ij}

1 Compute principal curvature directions per triangle

2 Between adjacent triangles t_i, t_j, compute best fitting matching r_{ij}

In step 1, a discrete approximation to the principal curvature directions is computed in each triangle. In our implementation, we used a method similar to that by Hildebrandt and Polthier [Hildebrandt 2004], but this step can also be exchanged by any other approach, e.g. [Hildebrandt 2011, Pottmann 2007, Cohen-Steiner 2003]. Note that we need principal curvature frames per triangle, not per vertex as commonly produced.

The algorithm is based on computing the shape operator $S(t_i)$ in each triangle t_i. The frame vectors are then defined as two perpendicular eigenvectors of $S(t_i)$ scaled to unit length. First, the shape operator is computed on each edge e_{ij} as 3×3 matrix:

$$S(e_{ij}) := H_{ij}(\vec{e}_{ij} \times N_{ij})(\vec{e}_{ij} \times N_{ij})^t, \tag{3.1}$$

with the following notation:

- $\vec{e}_{ij} := v_{\text{to}} - v_{\text{from}}$ is the (oriented) edge vector of e_{ij} which connects the two incident vertices $v_{\text{from}}, v_{\text{to}}$.
- $N_{ij} := \frac{N_i + N_j}{\|N_i + N_j\|}$ denotes the edge normal which is computed as average of the triangle normals N_i, N_j of t_i, t_j.
- $H_{ij} := 2\|\vec{e}_{ij}\| \cos \frac{\Theta_{ij}}{2}$ is the integrated mean curvature at the edge.
- $\Theta_{ij} := \angle(N_i, N_j)$ is the dihedral angle between triangles t_i, t_j.

Second, the shape operator in a vertex is the sum over $S(e_{ij})$ of all adjacent edges, and the shape operator in a triangle t_i is averaged over its three vertices. Thus, for short:

$$S(t_i) := \sum_{e_{jk} \in E} \varrho_{jk} S(e_{jk}), \quad \varrho_{ij} = \begin{cases} 2/3, & e_{jk} \text{ is an edge of } t_i \\ 1/3, & e_{jk} \text{ and } t_i \text{ share exactly 1 vertex} \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

Finally, two orthogonal eigenvectors (U_i, V_i) of $S(t_i)$ are computed and scaled to unit length. For maintaining the positive orientation of the frame, eigenvectors are chosen such that $V_i := \mathbf{J}U_i$. In general, if $S(t_i)$ is not a multiple of the identity matrix, then there are four possibilities of choosing U_i (pointing in minimum or maximum curvature direction and the sign of the vector). For parameterization, it does not matter which of these directions is taken since the difference is just a relabeling, the produced covering surface is exactly the same (see Section 2.4.3).

Step 2 of Algorithm 5 chooses a matching r_{ij} for all edges determining which vector U_i, V_i, $-U_i$, $-V_i$ in t_i is combinatorially matched to U_j in t_j, and therefore how the parameter lines of the parameterization are locally connected. Since we want the parameter lines to be as smooth as possible, it is a canonical choice to match those vectors with the smallest angle in between, i.e.:

$$r_{ij} := \left[\frac{\angle(U_j, U_i)}{\pi/2} \right] \tag{3.3}$$

This choice might not be globally optimal since the matchings determine the singularities and a noisy input field will cause far too many singularities. However, it defines an initial field for further processing.

3.2 Smoothing Frame Fields

Principal curvature fields are very stable in areas where the curvature is high in one specific direction and low in the perpendicular direction. However, in umbilic or planar regions of the surface, the shape operator becomes a multiple of the identity and there is no unique principal curvature direction. Thus close to an umbilic or planar region the field is not reliable and very prone to noise, see Fig. 3.1.

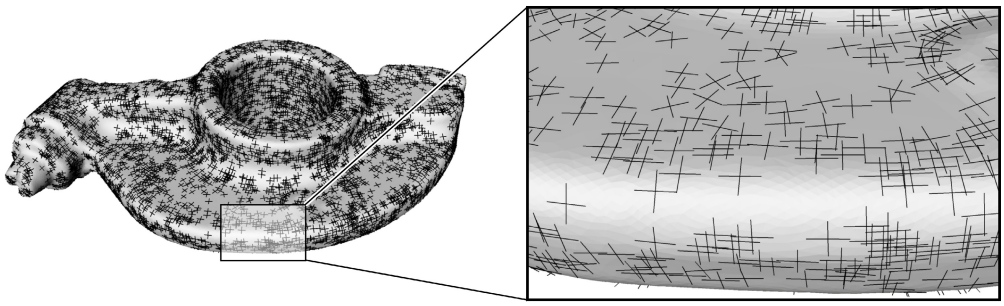


FIGURE 3.1: Principal curvature directions, computed with the algorithm from Section 3.1. The directions are unstable in planar or spherical regions.

To accommodate this issue, we identify regions where the shape operator is reliable. The field is then extended into the non-reliable region (or equivalently, it is smoothed by taking the reliability into account). The method is similar to that in [Bommes 2009], but extended to continuous reliability weights in contrast to having only trusted and non-trusted regions. Here, reliability is defined on each triangle t_i as:

$$\omega_i := \frac{\|\kappa_{i,1} - \kappa_{i,2}\|}{\max_{t_j} \|\kappa_{j,1} - \kappa_{j,2}\|} \tag{3.4}$$

where $\kappa_{i,1}, \kappa_{i,2}$ are the principal curvature values, resp. the eigenvalues of the shape operator $S(t_i)$. In practice, we keep only the most reliable 10% of all frame directions; all remaining weights are set to 0, which means that the corresponding frame directions are completely ignored.

Given an initial frame field (U_i, V_i), the smoothing procedure computes a new field whose frames are a rotated version of (U_i, V_i) by an angle α_i in the current triangle. Hereby, an energy $E := E_{\text{smooth}} + \tau E_{\text{align}}$, $\tau \in \mathbb{R}^+$ is minimized. The smoothing energy E_{smooth} minimizes the intrinsic angle between adjacent frames. It is defined by:

$$E_{\text{smooth}}(\alpha_i, r_{ij}) := \sum_{\text{edge } e_{ij}} \|\beta_{ij} + \alpha_j - \alpha_i - \frac{\pi}{2} r_{ij}\|^2 \tag{3.5}$$

where $\beta_{ij} \in (-\pi, \pi]$ is the intrinsic angle between U_i and U_j.

The alignment to the original field is measured by the alignment energy E_{align} which is given by:

$$E_{\text{align}}(\alpha_i, r_{ij}) := \sum_{\text{triangle } t_i} \omega_i \|\alpha_i\|^2. \tag{3.6}$$

Both energies are linearly combined with a parameter τ. The user chooses between smooth fields ($\tau = 0$) or maintaining curvature directions ($\tau \to \infty$).

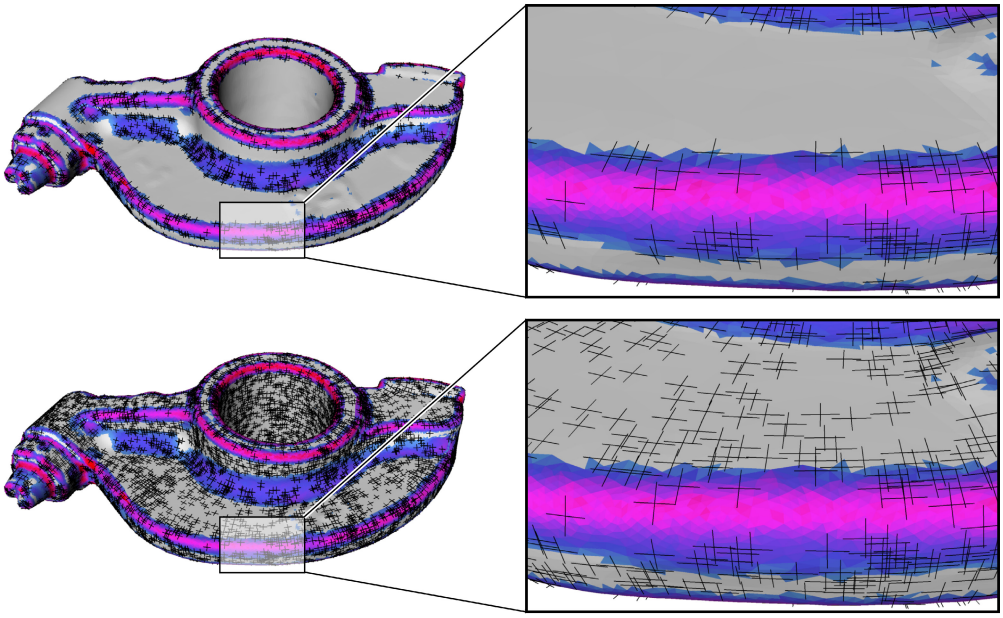


FIGURE 3.2: Computation of a stable curvature field. The colors represent the reliability function ω. Top: In regions where ω is too small (white color), τ is set to 0 and therefore those vectors are ignored. Bottom: Extended frame field.

This is a quadratic energy which has exactly one minimum for the α_i if the matchings are held fix. If matchings are variable, then some of the variables are redundant and we obtain several solutions. Since matchings are restricted to integer values, it turns into an NP hard optimization problem like the global step of QUADCOVER.

We therefore minimize the energy in a similar way as done with the QUADCOVER energy in Section 2.3.4: First, compute a spanning forest on the dual of M_h, using all triangles with $\omega_i \neq 0$ as root. Second, for all adjacent triangles t_i, t_j of the

spanning forest, restrict the matchings r_{ij} to 0. This simply removes redundant variables from the system. Third, minimize the energy (see Eqn. (3.5)) for all α_i and remaining r_{ij}. Finally, round all r_{ij} to the nearest integer and minimize the energy again with fixed matchings.

3.3 Optimizing Frame Fields

The outcome of the algorithm described in Section 3.1 and 3.2 is a smooth curvature-aligned frame field. Notice that the smoothing method relies on a heuristic for snapping the integer variables which does not provide the best locations for singularities. In practice, either clusters of singularities appear which are very close together or the location of singularities is not optimal, especially if aesthetics of the parameterization is important. In this section, a method for optimizing the location of singularities is proposed either automatically or by incorporating user requests.

A common artifact is distortion of the parameterization in the vicinity of a singularity (see Fig. 3.3, left). The parameterization "pushes" from one side and "pulls" from the other side of the singularity resulting in compression and stretching of parameter lines. If the singularity is allowed to move in direction of the stretched part, then the parameterization will relax.

Another issue considers the desired number of singularities. This number can be reduced by merging several singularities into a new one, and the index is just summed up. Depending on the application, this is a very useful operation since it produces smooth parameterizations with only few irregularities. However, the price is that the initial directions are altered and surface features are less respected (see also Fig. 5.6 for an example with hexagonal parameterization). Another price is that merging several singularities of index $1/4$ or $-1/4$ may generate higher order singularities, i.e. with an index of $1/2$ (see Fig. 3.3, right). For quad parameterizations, an index $1/2$-singularity corresponds to a hanging node with only 2 adjacent quadrilaterals. In order to produce quad meshes without hanging nodes, one can easily modify the following algorithm and disallow merging into an index $1/2$ singularity. All other operations are allowed, such as merging $1/4$, $-1/4$ into 0; merging $-1/4$, $-1/4$ into $-1/2$, and so on.

Singularity Movement. With our framework, one can move a singularity by changing the matchings locally. Let p be a singularity of index $k/4$ and γ be a path on edges which starts at p and ends at some other vertex q. The singularity is moved along γ into q by altering the matchings at all edges e_{ij} of γ. If t_i is the triangle at e_{ij} on the right side of γ and t_j the left one, then r_{ij} is replaced by $r_{ij} - k$.

This operation modifies the covering surface and moves the branch point from p to q, but it does not affect the vectors itself. In fact, the identification of frame vectors across γ is changed, and the frame field becomes discontinuous. Therefore, we have

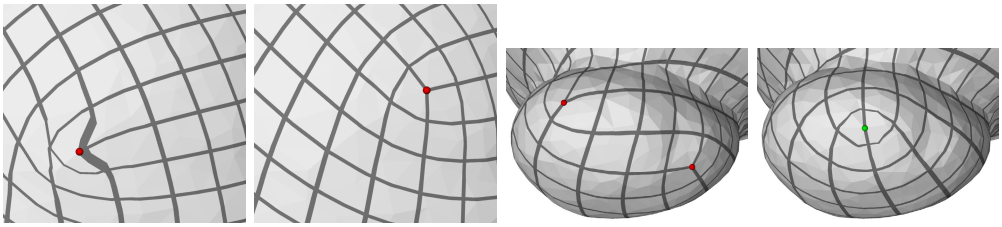


FIGURE 3.3: Singularity Optimization. Left: Parameterization with distortion in the vicinity of a singularity. Moving the singularity into right top direction relaxes the stress. Right: Two singularities (index 1/4) moved into the same location become merged to a higher order singularity (index 1/2). Notice that index 1/2 singularities are incident to only 2 quadrilaterals.

to minimize the energy E from Section 3.2 again (for the unknowns α_i, holding the matchings fix) in order to obtain a smooth frame field with desired singularities, see Fig. 3.4.

The output field is smooth as long as the path γ only passes through non-trusted regions, i.e. where alignment weights ω_i (see Section 3.2) are 0, since otherwise the alignment energy would produce a discontinuous seam where the matching has been changed.

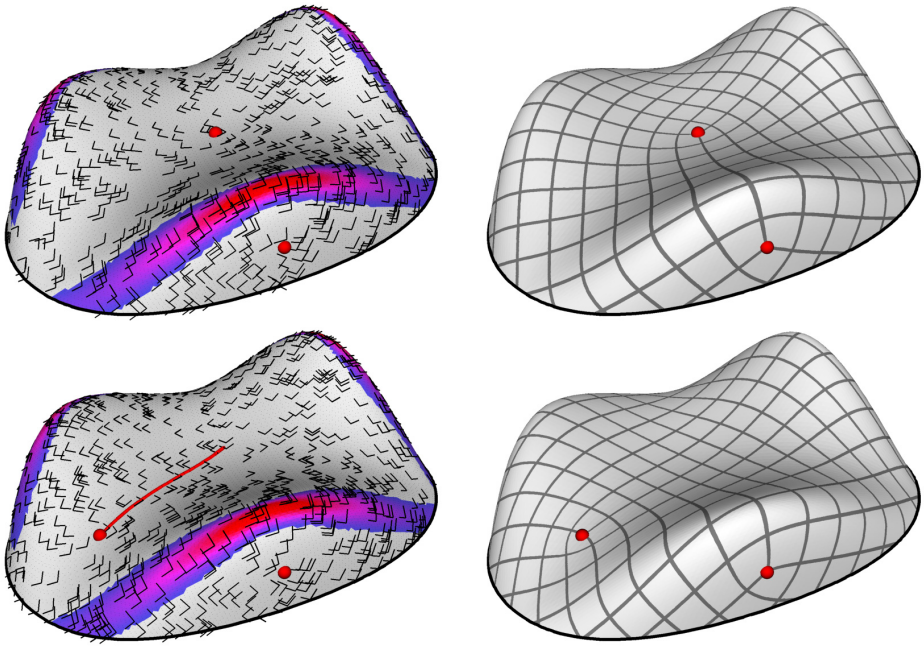


FIGURE 3.4: Top: Initial field and parameterization. The colors represent the trust function of the frame field. Bottom: Singularity was moved along the red path. The parameterization adapts to the new position.

We have integrated this singularity prescription in our implementation. After an

automatic parameterization, the user may pick an individual singularity and click on a destination vertex. A shortest edge-based path γ between both vertices is computed and the singularity is moved along γ. Merging of singularities is also possible, if γ ends on another singularity.

Relaxation. We implemented a simple heuristic for automatic singularity placement. After parameterization, each singularity is optimized locally by moving it successively to all adjacent vertices. The value of the current position is measured by the amount of curl of the frame field, since the curl measures how much the parameterization after the local QuadCover step deviates from the guiding field. Furthermore, if the distance between two singularities becomes too close, those singularities will be merged.

This relaxation method finds positions of singularities which are a local minimum for the total amount of curl. This heuristic gives very nice results in practice and singularities tend to move into the direction where the parameterization relaxes most (see Fig. 3.5).

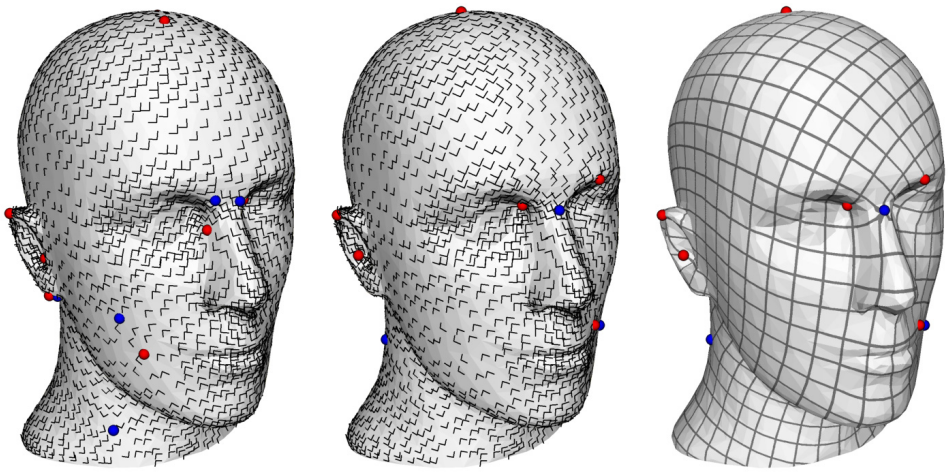


Figure 3.5: Left: Initial frame field generated from curvatures. Middle: Frame field after relaxation. Right: Parameterization using the relaxed field.

Chapter 4

Surface Tiling

Surface Tiling Using Parameterization. A *tiling* or *tessellation* of the Euclidean two-plane is a repetitive pattern that fills the whole plane. Based on the implied symmetry, tilings are classified into 17 distinct wallpaper groups. A symmetry is an isometry which leaves the pattern invariant. There are four categories of isometries of the plane: translations, rotations, reflections (in a line) and glide reflections (which are combinations of translations and reflections).

A regular tiling of the plane has always translational symmetries by two linearly independent displacement vectors. Furthermore, according to the so-called crystallographic restriction theorem, rotations are only possible by 60, 90, 120 or 180 degrees. Here we are only interested in orientation-preserving symmetries, i.e. only translations and rotations. The related wallpaper groups are called p1, p2, p3, p4 and p6 according to the degree of rotational symmetry.

Parameterization of a two-dimensional manifold is closely related to surface tiling: a parameterization transfers a wallpaper tiling of the two-plane onto a surface in a canonical way. Such patterns on geometries are commonly used for design purposes and in architecture.

In order to obtain a regular tiling on the surface without any visible seams, the parameterization must be compatible with the symmetry group of the tiling: the transition between charts must be symmetries of the actual wallpaper tiling. E.g. the QuadCover algorithm described in Chapter 2 is compatible with wallpaper group p4 since it allows rotations of 90 degrees. It can be used to produce quad tilings or any other patterns with similar symmetry, e.g. 4/8 patterns, see Fig. 4.1.

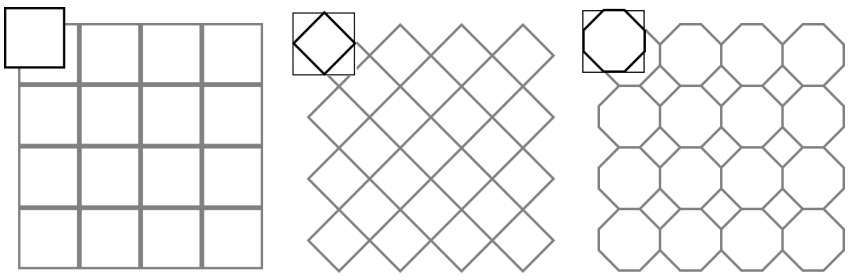


Figure 4.1: Different patterns which with 90 degrees rotational symmetry which are compatible with quadrilateral parameterizations.

This chapter focuses on different symmetry groups. Section 4.1 introduces hexag-

onal parameterizations (p6, p3) which can be used to generate regular triangle or hexagonal meshes. Stripe parameterizations allow 180 degrees rotations (p2) and are described in Section 4.2. Parameterizations with only translational symmetries (p1) are special since they have no singularities with fractional index. As a consequence, singularities are not resolved as branch points of the covering but as vector field singularities on the covering. Section 4.3 analyses this phenomenon and provides an algorithm for handling singularities of positive integer index, i.e. poles.

4.1 HexCover

Regular hexagonal patterns are one of the three regular patterns that can seamlessly tile a plane. They provide an optimal approximation to circle packings which have been linked to the wide appearance of hexagonal patterns in nature, such as honeycombs, insect eyes, fish eggs, and snow and water crystals, as well as in man-made objects such as floor tiling, carpet patterns, and architectural decorations (see Fig. 4.2).

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

FIGURE 4.2: Hexagonal patterns in nature: (a) honeycombs, (b) insect eyes, (c) snowflakes, (d) crystal, (e) naturally formed basalt columns.

Hexagonal parameterization has an important application in triangular remeshing, which refers to generating a triangular mesh from an input triangular mesh to improve its quality. (Note that triangular and hexagonal meshes are dual to each other, and triangular remeshing can also be used to perform hexagonal remeshing.) In triangular remeshing, it is often desirable to have all the triangles in the mesh being nearly equilateral and of uniform sizes, and the edges following the curvature and feature directions in the surface.

Automatic generation of a hexagonal parameterization from an input surface poses a number of challenges. First, unlike quadrangular parameterization whose parameter lines are parallel to either the major or the minor principal curvature directions, in hexagonal parameterization only one of the two directions can be used at each point on the surface. One must decide which direction to choose, and how to propagate such choices from a relatively small set of points to the whole surface to maintain the smoothness of the resulting parameterization. Section 4.1.1 discusses the generation of hexagonal frame fields.

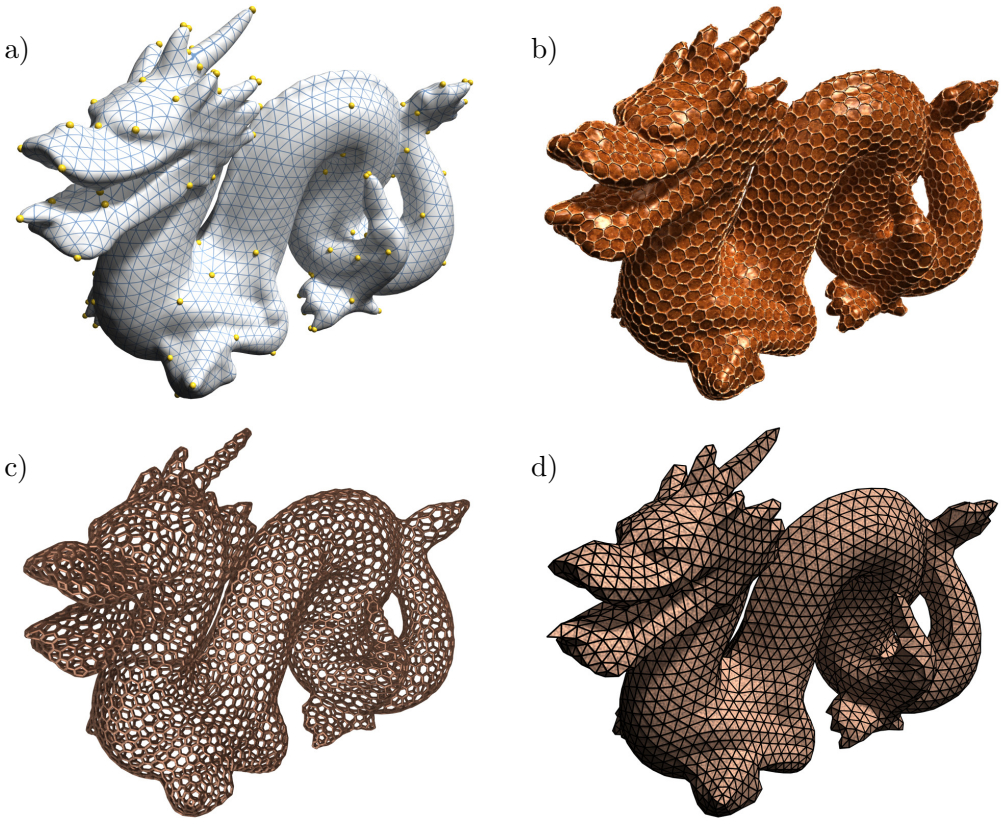


FIGURE 4.3: Hexagonal global parameterization (a), used for regular texture (b) and geometry pattern synthesis (c) with hexagonal patterns and for geometry-aware triangular remeshing (d).

Second, the continuity conditions developed for quadrangular parameterization along cut-paths are not appropriate for hexagonal parameterization (see Fig. 4.4). Section 4.1.2 explains how QUADCOVER is extended to generate global hexagonal parameterizations as described in [Nieser 2011a].

4.1.1 Hexagonal Frame Field Generation

Curvature-Aligned Frame Fields. In this section, we describe the pipeline for generating a geometry-aware hexagonal frame field, or 6-RoSy field, $X = (U, V)$ given an input surface M_h. In general, an N-symmetric frame field has a set of N frames in each triangle t_i: $R_N^k \cdot (U_i, V_i)$, $k \in \{0, \dots, N-1\}$ where (U_i, V_i) is one reference frame and R_N^k is the linear operator that rotates a given frame by $\frac{2\pi k}{N}$ in the local coordinates defined by the frame, i.e. $R_N^k \cdot (U_i, V_i) = (U_i \cos\frac{2\pi k}{N} + V_i \sin\frac{2\pi k}{N}, V_i \cos\frac{2\pi k}{N} - U_i \sin\frac{2\pi k}{N})$.

The relation between the reference frames in adjacent triangles t_i, t_j is given by matchings $r_{ij} \in \{0, \dots, N-1\}$: the frame (U_j, V_j) is combinatorially linked with

(a) (b) (c)

FIGURE 4.4: A quadrangular parameterization ensures that the discontinuity along the cut is invisible (a). The same parameterization is incompatible with a hexagonal pattern (b), which leads to seams (yellow). In this case a hexagonal parameterization is needed (c).

the frame $R_N^{r_{ij}} \cdot (U_i, V_i)$. Similar to the quadrangular case (see Chapter 2), the matchings introduce singularities with fractional index of $\frac{k}{N}$, $k \in \mathbb{Z}$.

To automatically compute a hexagonal frame field, we need to answer the question of what direction is assigned in each triangle depending on the principal curvature directions. In the final parameterization, parameter lines should follow principal curvature directions. Lines ignoring surface features cause "twisting" artifacts (see Fig. 4.5). We therefore need to solve the problem of matching the 6 directions of the hexagonal field with the four principal curvature directions.

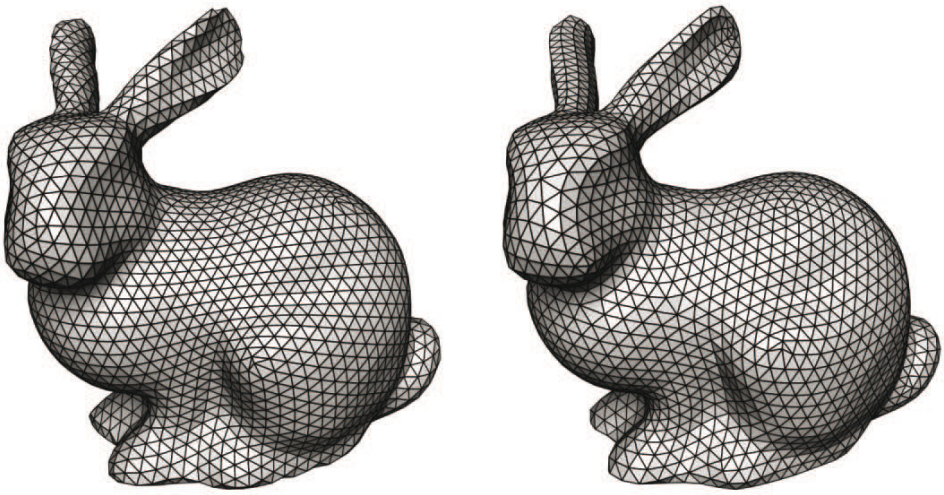


FIGURE 4.5: For remeshing, edges should follow principal curvature directions (right). Edges ignoring surface features (left) cause "twisting" artifacts (on the ears).

Classification of the Curvature Tensor. To align the parameter lines with feature lines such as ridges and valleys, we will choose the principal direction that has the most bending, i.e., maximum absolute principal curvature, as one of the directions in the hexagonal frame field.

Principal curvature directions are most meaningful in cylindrical and hyperbolic regions due to the strong anisotropy there. However, while purely hyperbolic regions

possess strong anisotropy, the absolute principal curvatures are nearly indistinguishable, thus making both principal curvature directions candidates. Moreover, the two bisectors between the major and minor principal curvature directions can also provide viable choices for the edge directions in hyperbolic regions. Due to the excessive choice of directions in hyperbolic regions and insufficient choice of directions in planar and spherical regions, we only generate frames in cylindrical regions and then extend the field to the whole surface.

Let $S(t_i)$ be the curvature tensor in one triangle and κ_1, κ_2 be the corresponding curvature values as computed in Section 3.1. The vector $\binom{\kappa_1-\kappa_2}{\kappa_1+\kappa_2}$ can be written in polar coordinates $\varrho\binom{\cos\Phi}{\sin\Phi}$, $\varrho \in \mathbb{R}^+$, $\Phi \in (-\pi, \pi]$ and classified into six special configurations. The first satisfying $\varrho(p) = 0$, i.e., geometry is locally planar. For the remaining five configurations we have $\varrho(p) > 0$. Respectively, they correspond to $\Phi(p) = \frac{\pi}{2}$ (spherical), $\Phi(p) = \frac{\pi}{4}$ (cylindrical), $\Phi(p) = 0$ (purely hyperbolic), $\Phi(p) = -\frac{\pi}{4}$ (inverted cylindrical), and $\Phi(p) = -\frac{\pi}{2}$ (inverted spherical). With this representation, we can classify any point p as being planar if $\varrho(p)$ is smaller than a given threshold δ, elliptical if $\Phi(p) \geq \delta$ and $|\Phi(p)| > \frac{3}{8}\pi$, hyperbolic if $\Phi(p) \geq \delta$ and $|\Phi(p)| < \frac{1}{8}\pi$, and cylindrical otherwise (see Fig. 4.6).

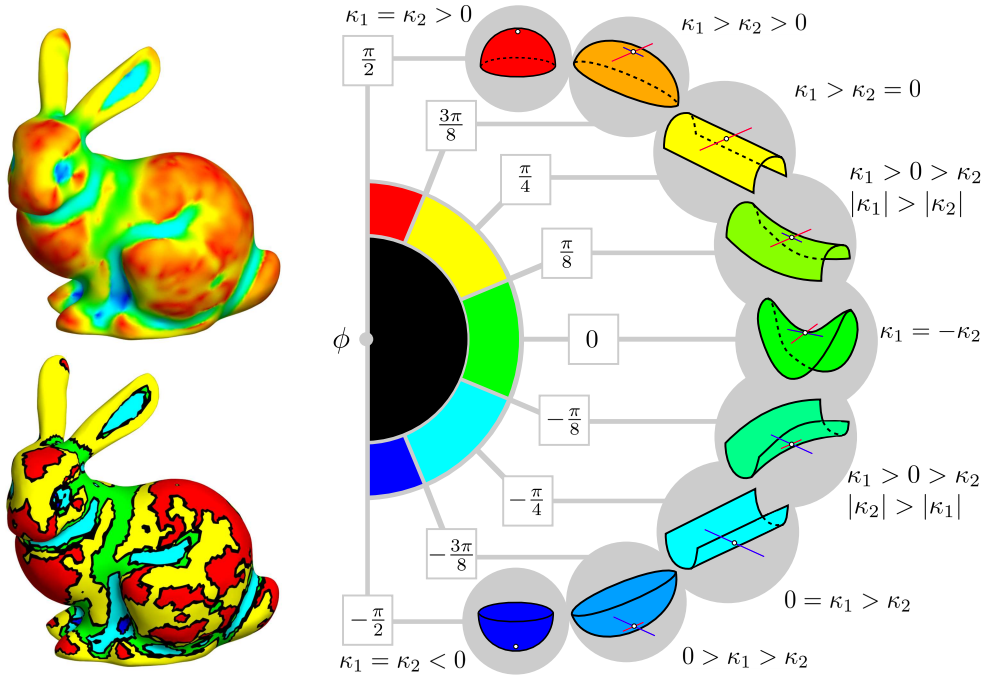


FIGURE 4.6: Surface classification scheme to determine directional constraints. $\phi \in [-\pi/2, \pi/2]$ is color mapped to the [blue,red] arc in HSV color space: Left top: continuous mapping. Bottom: binned classification. The legend (right) shows surfaces patches which are locally similar to points with given values.

Field Generation. Given the classification, we select the maximum direction as reference vector in triangles where $\Phi > 0$ and the minimum direction where

$\Phi < 0$. Recall that the directions in the output field specify the gradients in our resulting parameterization, and we wish one of the isolines of the parameters to be orthogonal to the direction in which the surface is bending the most. Finally, we propagate the directions in the cylindrical regions into non-cylindrical regions (planar, spherical, hyperbolic) using the same energy minimization approach as described in Section 3.2.

In addition to this automatic technique, the editing and frame field optimization methods described in Section 3.3 also apply to hexagonal frame fields.

4.1.2 Hexagonal Parameterization

In this section the second stage of the pipeline is described, which constructs a hexagonal global parameterization from an input triangular mesh with a given hexagonal frame field. We will first describe the differences of hexagonal parameterization to the quadrangular case.

Hexagonal Parameterization and Energy. Given a mesh M_h with $|T|$ triangles, a *global parameterization* $\varphi : M_h \to \mathbb{R}^2$ *respecting an N-rotational symmetry* is a collection of linear maps $\{\varphi_i \,|\, 1 \leq i \leq \|T|\}$ with the following property. For any adjacent triangles t_i and t_j we have:

$$\varphi_j(p) = R_N^{r_{ij}} \varphi_i(p) + w_{ij}, \quad \forall p \in t_i \cap t_j. \tag{4.1}$$

with matchings $r_{ij} \in \{0, \ldots, N-1\}$ and gaps $w_{ij} \in G_N$ in some integer lattice G_N.

In the quadrangular case ($N = 4$), translational discontinuities w_{ij} are required to be on the set of Gauss integers $G_4 := \{(a, b)^T \,|\, a, b \in \mathbb{Z}\}$. Therefore a texture image with 90 degrees rotational symmetry around any point in G_4 is continuously mapped onto the surface.

Hexagonal parameterization ($N = 6$) is similar, except that in this case the texture image needs to respect hexagonal rotational symmetries. A canonical choice is a hexagonal or triangular pattern as shown in Fig. 4.7 (c). The texture image has an aspect ratio of $1 : \sqrt{3}$ and tiles the plane seamlessly. It is furthermore invariant under rotations of $\frac{\pi}{3}$ around the center of each hexagon. The set of these center points is known as the *Eisenstein integer lattice*, shown in Fig. 4.7 (d):

$$G_6 := \left\{ a \begin{pmatrix} 1 \\ 0 \end{pmatrix} + b \begin{pmatrix} 1/2 \\ \sqrt{3}/2 \end{pmatrix} \,\middle|\, a, b \in \mathbb{Z} \right\}. \tag{4.2}$$

Besides the rotational invariance, the hexagonal grid also remains invariant under translations by any vector in G_6. The discontinuities are therefore not visible if all gaps w_{ij} are in G_6 because of the repeating structure of the texture image.

HexCover and Covering Spaces. A hexagonal frame field is actually equivalent to two multi-valued vector fields on the surface. Similar to the setting in Section 2.4,

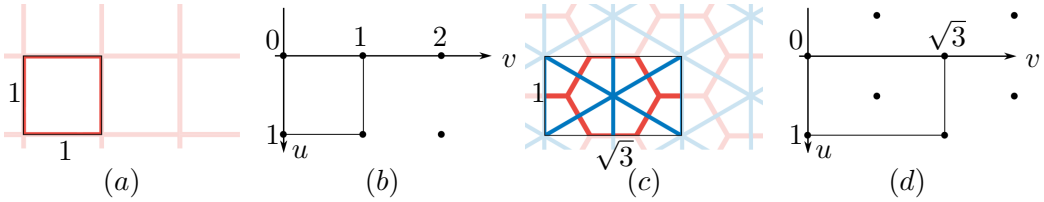


FIGURE 4.7: (a) Tileable texture with quadrangular rotational symmetry. (b) Integer lattice G_4. (c) Tileable texture with hexagonal rotational symmetry. (d) Eisenstein integer lattice G_6.

we make use of a covering space, which transforms a hexagonal frame field into two vector fields on a 6-fold covering surface of M_h.

Every triangle t_i in M_h has six corresponding triangles in M_h': $t'_{i,0}, \ldots, t'_{i,5}$. The frame field on M_h is lifted to the covering by distributing the six vectors onto the six copies, i.e., $U'_{i,j} = R_6^j U_{i,0}$, $V'_{i,j} = \mathbf{J}U'_{i,j}$ where $U_{i,0}$ is one of the six directions of the hexagonal field in t_i. For adjacent triangles t_i, t_j in M_h, the corresponding copies are combinatorially connected, depending on the matching r_{ij}. The triangles $t'_{i,k}$, $k \in \{0, \ldots, 5\}$ are thereby connected with $t'_{j,k+rij \bmod 6}$ (Fig. 4.8).

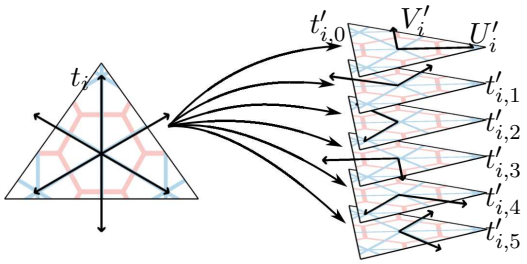


FIGURE 4.8: Left: Triangle t_i with 6-RoSy field. Right: 6-fold covering of t_i with vector fields F'_u, F'_v.

Notice that also in the hexagonal case, the covering is just used as theoretical construct and is not explicitly computed in the algorithm.

Algorithm. The algorithm for computing a hexagonal parameterization is similar to QuadCover, but with two main differences:

1. The matching numbers are up to modulo 6 instead of 4. The **J**-operator in Eqn. (2.2) turns into a rotation R_6 by 60 degrees.

2. To enforce the integer constraints, the gaps w_{ij} are modified by rounding them to the nearest point in G_6 instead of G_4.

Figure 4.9 shows the hexagonal parameterization of two minimal surfaces using HexCover.

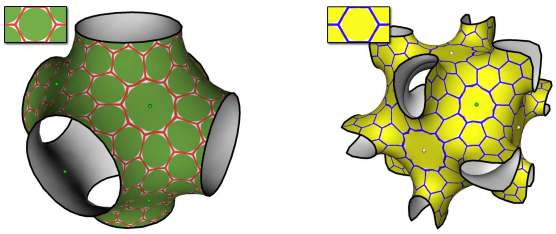


FIGURE 4.9: Minimal surfaces. Left: Schwarz surface with 8 singularities of index $-1/2$. Right: Neovius surface with 8 index $-1/2$ and 6 index -1 singularities.

4.2 Stripe Parameterization

Stripe parameterizations can be used for mapping texture images which are symmetric by rotations of 180 but not necessarily 90 degrees, such as stripe textures. A natural domain are tubular surfaces where the minimum and maximum curvature directions decouple in most areas into two *directional fields*. Mathematically these surfaces can be described as thickened graphs, and the calculated parameterization stripe will follow either around the tube, along the tube axis or a spiraling combination of both.

Tubular surfaces appear in many application areas such as networks of blood vessels and neurons in medicine, or tube and hose systems in industrial environments. Often a tubular structure must be recovered and segmented from noisy scan data. QUADCOVER can be efficiently used for automatic stripe parameterization of tubular surfaces given as triangle meshes. An additional benefit of the stripe parameterization is the enhanced visualization of the underlying geometric structure. The method for computing stripe parameterizations was first described in [Kälberer 2010].

Differences. Similar to HEXCOVER, the algorithm for computing stripe parameterizations differs from QUADCOVER by another symmetry of the texture pattern and a different integer property. Transition functions follow Eqn. (4.1) for $N = 2$ and the gaps are restricted to $w_{ij} \in \mathbb{Z}^2$.

Conceptually, stripe parameterizations are a special case of quad parameterizations where all indices of singularities are integer multiples of $1/2$, i.e. indices like $1/4$ or $-1/4$ are not allowed. The frames of guiding field for stripe parameterization are defined up to the sign $\pm(U_i, V_i)$ in each triangle t_i. Therefore, it decouples into two (directional) vector fields $\pm U_i$ and $\pm V_i$. The covering simplifies to a two-sheeted covering surface (see Fig. 4.10).

Field-Generation. For frame field generation, a similar method as in Section 4.1.1 can be used: choose the minimum curvature vector in all cylindrical regions and extend this field onto the whole surface. If we consider tubular surfaces only, this is especially stable since then the minimum curvature direction provides a stable field nearly everywhere.

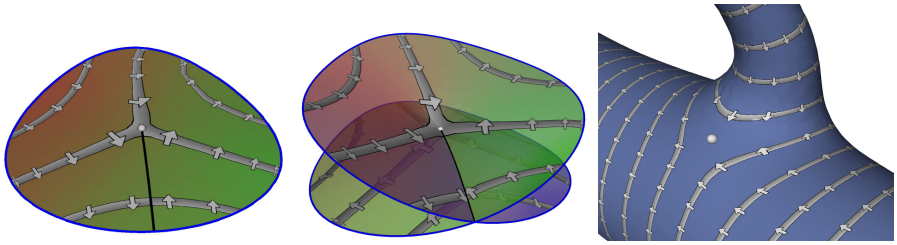


FIGURE 4.10: Left: Stripe parameterization with branch point. The isolines of the u function and its gradient vectors are drawn. Middle: The same parameter function on the 2-sheeted covering (the covering surface is not embedded, it has self-intersections). Right: Branch point on a parameterized tube object.

Parameterization. The parameterization step is done similarly to QUADCOVER (Section 2.4). Notice that the matrices of the systems of equations may be decomposed into two (identical) matrices of half dimension since the u and v component of the parameterization never interact. The computing time can therefore be speed up by using the smaller matrix for the u part and then for the v part.

Examples. Here are some examples of tubular surfaces with a stripe parameterization. The surface in Fig. 4.11 shows a human blood vessel which contains parts with a very large tube radius as well as very filigree branches. Regardless of this difference in the scaling, the stripe density stays nearly constant everywhere.

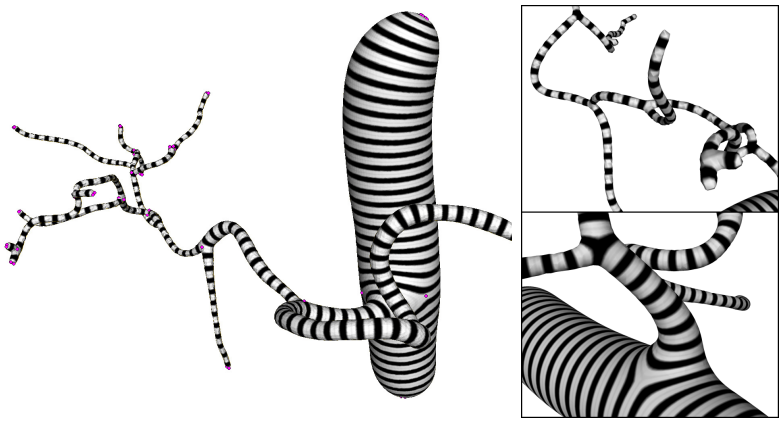


FIGURE 4.11: Parameterized blood vessel, captured by MRT. Courtesy of Fraunhofer MEVIS.

Figure 4.12 shows a complex neuron model of genus 23, captured using confocal microscopy. The produced parameterization has very little distortion even on this complicated object. The stripe pattern helps to perceive the complicated shape of the neuron.

Twisted Stripes. There are two different possibilities of generating twisted stripes with QUADCOVER. First, take a stripe parameterization and locally rotate each frame by any angle in its triangle. Then parameterize again using the twisted field.

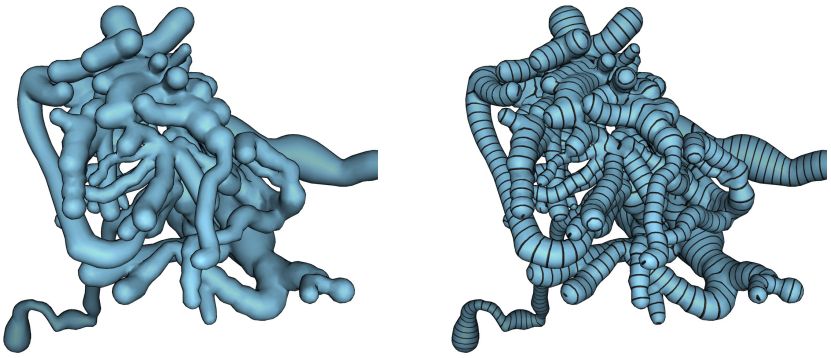


FIGURE 4.12: Parameterized neuron by courtesy of Freie Universität Berlin, Department of Neurobiology.

The second technique is based on the fact that the stripes in the texture image do not necessarily point along the u axis. The only requirement is that the image is 180 degrees rotational invariant. If a given stripe texture image is given, consider the transformation $\binom{u}{v} \mapsto \begin{pmatrix} n & -k \\ k & n \end{pmatrix} \binom{u}{v}$ with $k, n \in \mathbb{N}^+$. It is basically a scaling and rotation of the texture space (e.g. $k = n = 1$ rotates by 45 degrees) such that the new texture is still invariant under translations in $\mathbb{Z}^2$. Thus, without computing a new parameterization, the underlying texture image can just be replaced in order to adjust the twisting behaviour of the stripe pattern.

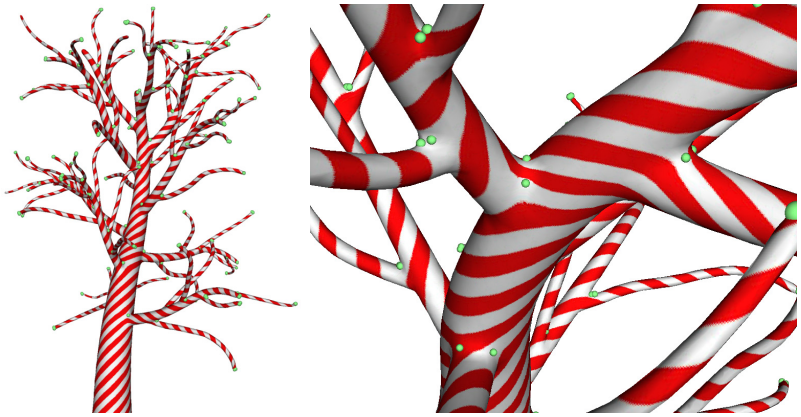


FIGURE 4.13: Tree model with diagonal candy cane stripes. Singularities are marked in green.

4.3 Singularities of Positive Integral Index

4.3.1 Fractional vs Integral Index

For QUADCOVER parameterizations, the location of singularities with fractional index are determined by the matchings r_{ij}. They are defined as vertices where the frame vectors exchange when tracing a small loop around the point. Such

singularities are resolved as a branch point in the covering surface.

Singularities whose index is an integer behave fundamentally different. If a sphere e.g. is parameterized in polar coordinates, then all matchings are 0; there is no flip of the parameter lines anywhere (see Fig. 4.14, left). These singularities do not introduce any branch points since walking around the pole ends up on the same layer as started with. The covering itself does not "see" those singularities; they just arise as gradient field singularities *on* the covering.

As a direct consequence, once the matchings are known, the location of all fractional singularities is fixed whereas the location of integer singularities is still variable and will be optimized as well.

A special case are singularities with positive integral index. Let $\varphi = (u, v)$ be a parameterization and the gradient field ∇v has a vortex in point p (like the north pole of the sphere, see Fig. 4.14, right). Unfortunately, vortices cannot be described using a standard PL parameter map on triangles because of the following reason: If γ is a small path around p, then the path integral $\int_\gamma \nabla v \, ds$ is always a constant number, which stays the same as the curve gets contracted. Therefore, in the near vicinity of p, the gradients must tend to infinity (like an irrotational vortex). It is impossible to represent such a function as a piecewise linear function on a triangle mesh.

Hence, if there are vortices present in the input frame field, the parameterization algorithm ignores it and produces a parameter function which is far from the guiding field (see Fig. 4.14, left). By the way, this issue arises in all parameterization approaches based on a similar PL setting, like [Tong 2006, Bommes 2009]. They all rely on having no singularities of positive integer index which is sufficient in most cases. However, if for some reason a pole is desired, or if the sum of the prescribed fractional indices is not consistent with the Euler characteristic of the surface, then singularities of integral index cannot be avoided and special treatment is necessary for avoiding a degenerated situation like in Fig. 4.14, left.

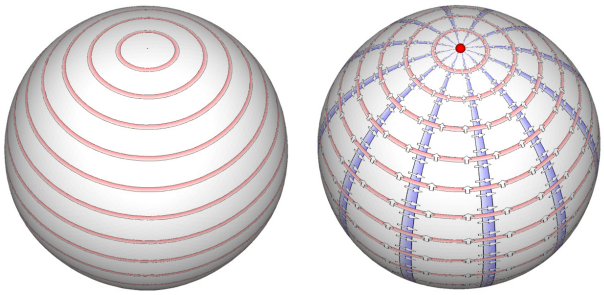


FIGURE 4.14: Left: Resulting parameterization if a representation with piecewise linear functions is used. The v component nearly vanishes everywhere. Right: Polar parameterization (u, v) of a sphere.

In Section 4.3.2 a method [Nieser 2009] is described which handles those singularities

and therefore avoids degenerated parameterizations.

4.3.2 Handle Singularities of Positive Integral Index

Use quads. A regular triangle mesh is too rigid for representing a piecewise linear vortex-like parameter function. Thus, we locally remesh the surface to add some quadrilateral elements. The shape of the surface stays the same, but the quads allow a more complex structure of the parameter function.

Let p be a vertex, where the frame field has an index of $k \in \mathbb{N}^+$. We locally change the mesh and cut an infinitesimal small hole at p. The vertex star is remeshed as shown in Fig. 4.15. All triangles in the star become degenerated quads with one edge of length 0.

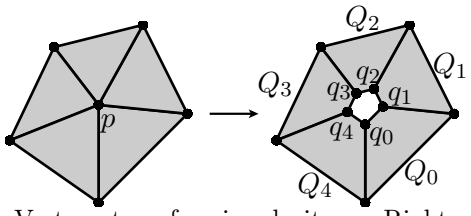


FIGURE 4.15: Left: Vertex star of a singularity p. Right: Combinatoric of the remeshed vertex star. It consists of 4 quads Q_i. The inner vertices q_i are geometrically located in the same point p, thus one edge of each quad has zero length.

Scalar functions on a mesh with quads are not longer forced to be piecewise linear. Here, we use functions which are linear on each triangle and bilinear on each quadrangle. Given function values at the vertices of a quad, the parameterization is then given by the unique bilinear function, which interpolates these values. The resulting function is continuous (up to discontinuities introduced by gaps), even if triangles and quads are mixed in the same mesh.

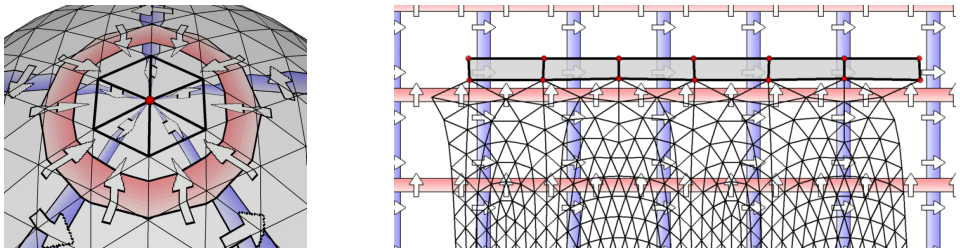


FIGURE 4.16: Left: Bilinear texture on a coarse mesh with triangles and quads. Right: Image of the elements in texture space. The quads in the vertex star of the singularity are marked in grey.

The advantage of using quadrilaterals is that the singular vertex is now represented by a set of edges which can be mapped to a polygonal line in texture space. Figure 4.16 shows a parameterization of a pole.

Approximation. The original QUADCOVER algorithm works on triangles. It

would be straight forward to generalize it to quadrangles. The only difference is that the function space for energy minimization changes. Although, an adaption of the optimization algorithm to work with bilinear functions turns out to be complicated, since the computation of the derivatives of the energy requires to solve a non-linear integral. Instead, we simplified the problem and approximated the optimal solution. This approximation replaces the quads by triangles again, but with an altered connectivity. Thus the standard QuadCover algorithm can be used. The quads are then used afterwards for the final description and visualization of the result. The outline of the modified QuadCover is listed in Algorithm 6.

Algorithm 6: Modified QuadCover algorithm, Input: Guiding frame field

```
1: for all vertices p do
2:     Measure index of frame field at p
3:     if (index mod 1 == 0) and (index > 0) then
4:         Store vertex p in array specialSingularities
5:         Cut all outgoing edges from p open.
6: Run original QuadCover algorithm
7: for all p in specialSingularities do
8:     Replace all adjacent triangles to p by a quadrilateral
9:     Compute texture coordinates for the quads
```

Lines 1–5 do a local remeshing at each vertex p with positive integer index. All adjacent edges to p are cut open generating a hole in the surface (see Fig. 4.17). In line 6, QuadCover is applied to the triangle surface. Figure 4.18 shows the texture domain of the example from Fig. 4.16.

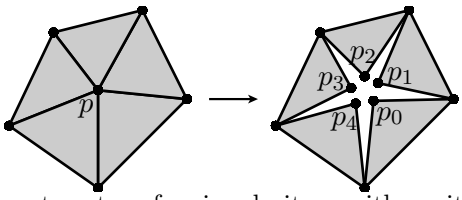


Figure 4.17: Each vertex star of a singularity p with positive integer index will be cut open. The right mesh shows the new combinatoric, the inner vertices p_i are geometrically located at p.

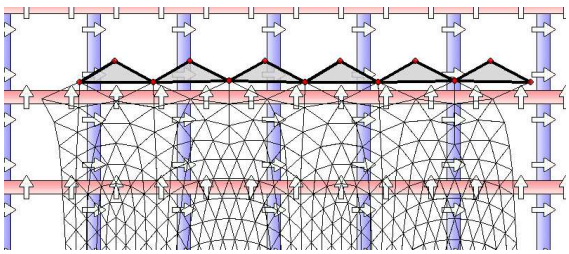


Figure 4.18: Texture domain after parameterization with modified mesh connectivity. The surface from Fig. 4.16, left is taken as input.

In lines 7–9, the singularities are remeshed again. Each triangle of the vertex star (which was previously cut open) is now replaced by a quadrangle. All quads are connected as in the situation of Fig. 4.15, right.

It remains to compute the texture coordinates for the created vertices q_i. They are obtained by just averaging the texture coordinates of the old vertices p_i (from Fig. 4.17, right). In quadrangle Q_j, compute the texture coordinates as:

$$\begin{aligned} f_j(q_j) &:= 1/2\,(f_{j-1}(p_{i-1}) + w_{i-1,i} + f_j(p_j)) \\ f_j(q_{j+1}) &:= 1/2\,(f_j(p_j) + f_{j+1}(p_{i+1}) + w_{i+1,i}) \end{aligned} \tag{4.3}$$

where $w_{i-1,i}$ is the translational part of the transition (gaps) between chart Q_{i-1} and Q_i, see Eqn. (2.2).

4.4 Constraints and Boundary Conditions

An important feature for parameterization software is the possibility to adhere to special user given constraints. Commonly, parameter lines should follow sharp creases in the surface or stick to the boundary. Here we consider two different types of constraints:

Geometrical Constraint: Force a parameter line to follow a given path δ on the surface exactly, e.g. running along a crease or the boundary.

Combinatorial Constraint: Constrain the connectivity of the parameter, e.g. two singularities are forced to be connected by a parameter line, but the exact path is not given.

Geometrical constraints are the stronger ones and some geometrical constraints implicitly force combinatorial constraints. If e.g. a geometrical constraint is given by a curve δ which connects two singularities, then this also restricts the combinatorics of the parameter grid.

Here, we provide an algorithm for processing these kinds of constraints. As far as we know, this is the very first approach to explicitly enforce combinatorial constraints to parameterizations, whereas geometrical constraints are also used in other methods [Bommes 2009, Zhang 2010]. An advantage of our approach is the splitting of constraints into a pure combinatorial and a geometrical part. This allows further processing, e.g. one can test if several constraints contradict by checking the rank of the combinatorial matrix C_c (see next section).

Section 4.4.1 describes how constraints are formulated and integrated into QuadCover. Section 4.4.2 shows how geometrical and combinatorial constraints are actually implemented.

4.4.1 QuadCover with Constraints

Given a parameterization φ_i on a triangulated manifold M_h. As formulated in Section 2.4.5, the parameterization is represented according to Eqn. (2.41) using the following data:

1. Symmetry order $n \in \mathbb{N}$. $n = 4$ for quadrangular-, $n = 6$ for hexagonal-, $n = 2$ for stripe parameterization or any other symmetry order. In the following, denote R_n as operator which rotates a given vector in the plane by $2\pi/n$.
2. Matchings $r_{ij} \in \{0, \dots, n-1\}$ on edges. They define the fractional index of all vertices and therefore the position of singularities.
3. Cut graph G consisting of a set of N cut paths $G = \{\gamma_k, k \in \{0, \dots, N-1\}\}$.
4. Gaps $w_{u,k}, w_{v,k}$ for each cut γ_k
5. Parameter values u_i, v_i at each regular vertex p_i.

The parameterization $\varphi_i = (u_i, v_i)$ is piecewise linear and may be discontinuous across edges. Remember, that adjacent triangles t_i,t_j are related by:

$$\varphi_{|t_i} = R_n^{r_{ij}} \varphi_{|t_j} + w_{ij}. \tag{4.4}$$

Here, the gap w_{ij} is equal to the (oriented) sum of gaps w_k associated to all cut-paths γ_k which proceed over the edge between t_i and t_j. QuadCover generates a parameterization by optimizing the free variables $u_i, v_i \in \mathbb{R}$ and $(w_{u,k}, w_{v,k}) \in G_N$ with an integer lattice G_N (see Sect. 4.1.2).

Geometrical Constraints. Geometrical constraints are defined by a set of ν curves $\delta_\kappa : [0,1] \to M_h$, $\kappa \in \{0, \dots, \nu-1\}$ which w.l.o.g. run over edges of M_h (if a curve traverses through a triangle, we can subdivide the triangle and δ_κ becomes a curve on edges). The curve may be open or a closed loop ($\delta_\kappa(0) = \delta_\kappa(1)$).

We desire that each δ_κ is exactly represented by a parameter line. For a quadrangular parameterization, we therefore constraint one parameter value (u or v) along each δ_κ to a value $c_\kappa \in \mathbb{Z}$ which becomes a new variable in our system. In general for an arbitrary symmetry order, we want to constrain either the u component of φ or the u component of any rotated version $(R_n^\lambda \varphi)_u$, $\lambda \in \{0, \dots, n-1\}$ to c_κ (see Fig. 4.19).

Let p_0 be the start vertex of δ_κ and t_0 be the first triangle on the right side of δ_κ, we constrain

$$(R_n^\lambda \varphi(p_0)_{|t_0})_u \equiv c_\kappa. \tag{4.5}$$

With plugging in Eqn. (2.41), this becomes a linear equation in the coeffients u_0, v_0 if all gaps $w_{u,j}$, $w_{v,j}$ and the constraint value c_κ are known.

Since the parameter map φ is in general not continuous everywhere, the constraint must be altered along δ_κ according to the matching and gaps (if δ_κ crosses any cut paths). Therefore, by tracing all vertices and corresponding triangles on the right

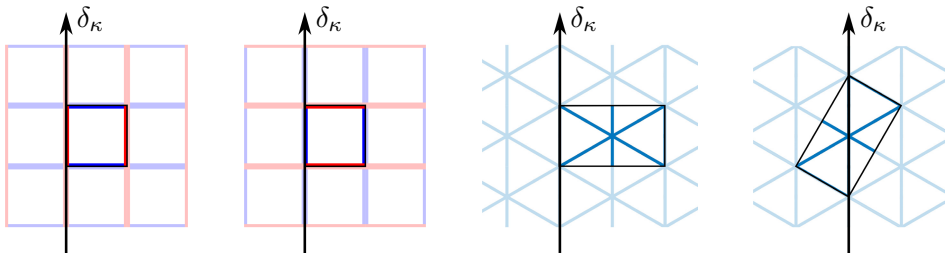


FIGURE 4.19: From left to right: quadrilateral parameterization with u-lines fitting to the constraint δ_κ; fitting v-lines; hexagonal parameterization with fitting u-lines; with fitting 60-degrees-rotated u-lines.

side of δ_κ, we can establish one linear equation for each vertex p_i on δ_κ of the form:

$$\left(R^\rho \begin{pmatrix} u_i \\ v_i \end{pmatrix} + \sum_{\gamma_k \in G} R^{\rho_k} \begin{pmatrix} w_{u,k} \\ w_{v,k} \end{pmatrix} \right)_u \equiv c_\kappa. \tag{4.6}$$

with coefficients ($\rho \in \mathbb{Z}, \rho_k \in \mathbb{Z}$). All these linear equations are collected into a system:

$$C_g \vec{u} = b, \tag{4.7}$$

where $\vec{u} \in \mathbb{R}^{2|V|}$ contains all real valued coefficients u_i, v_i and $b \in \mathbb{R}^{2|V|}$ is a vector which depends on all integer coefficients $w_{u,k}$, $w_{v,k}$ and on c_κ.

In the 1st step of QUADCOVER, we compute a local parameterization φ as before ignoring any given constraints. After that, we measure the value c_κ as the average value on the right side of every path δ_κ (attending the discontinuities along the path as described above). When gaps are rounded, the values c_κ are also snapped to the nearest integer (resp. to the nearest integer-multiple of $\sqrt{3}$ if $n = 6$).

In the 2nd step of QUADCOVER, the gaps are known and Eqn. (4.7) is used as linear constaint during energy minimization (e.g. by using Lagrange multiplier).

Combinatorial Constraints. A combinatorial constraint is a linear condition on the integer variables in our system, i.e. the gaps $w_{u,k}, w_{v,k}$ plus the constraint values c_κ. They alone determine the combinatorial layout of the net, i.e. the exact number of parameter lines and how they are connected to each other. Thus a combinatorial constraint prescribes conditions on this layout. We write a combinatorial constraint by the linear equation:

$$C_c \vec{w} = 0 \tag{4.8}$$

where $\vec{w} = \{w_{u,0}, w_{v,0}, w_{u,1}, w_{v,1}, \ldots, w_{v,N-1}, c_0, \ldots c_{\nu-1}\} \in \mathbb{R}^{2N+\nu}$ is a vector containing all integer variables of our system.

A geometrical constraint as explained above sometimes enforces a combinatorial constraint: While tracing the curve δ_κ all vertices on the curve are constrained by Eqn. (4.6) with coefficients (ρ, ρ_k, c_κ). It may happen that a vertex is already constrained (e.g. by another geometrical constraint) with different coefficients

$(\rho, \rho'_k, c'_\kappa)$, then we get a combinatorial constraint:

$$c_\kappa - \left(\sum_{\gamma_k \in G} R^{\rho_k} \begin{pmatrix} w_{u,k} \\ w_{v,k} \end{pmatrix} \right)_u \equiv c'_\kappa - \left(\sum_{\gamma_k \in G} R^{\rho'_k} \begin{pmatrix} w_{u,k} \\ w_{v,k} \end{pmatrix} \right)_u . \tag{4.9}$$

In a similar way, we get a combinatorial constraint whenever δ_κ touches a singular vertex, since the parameter values at singularities are uniquely determined by the gaps (see Eqn. (2.40)).

In summary, if we start with a given set of curves δ_κ and set up geometrical constraints, we get one combinatorial constraint each time a curve touches a singularity and each time, two curves meet (or one curve meets itself, e.g. if it is a closed curve) and the coefficient ρ of both curves is similar.

We assemble all combinatorial constraints in the matrix C_c and assure $C_c \vec{w} = 0$ when rounding the integer variables. We do this by performing a Gauss method on C_c and eliminate variables which depend on the remaining ones. Notice that C_c has only integer entries, we therefore do not run into numerical problems. The free variables are then snapped to the integer grid and the dependent ones are defined by the linear system (and are of course also integer).

Algorithm 7: QuadCover with constraints

Input: triangle mesh, guiding frame field, constraint curves δ_κ
Output: global parameterization

1 Perform local QuadCover step and obtain parameterization φ with coefficients $u_i, v_i, w_{u,k}, w_{v,k}$
2 Compute constraint values c_κ as average of φ along δ_κ
3 Assemble geometrical and combinatorial constraints in matrices C_g and C_c
4 Perform Gauss elimination on C_c
5 Round gaps $w_{u,k}$, $w_{v,k}$ and c_κ to integer grid satisfying $C_c \vec{w} = 0$
6 Perform global QuadCover step under linear constraints $C_g \vec{u} = b$

4.4.2 Using Meta-Graphs

The extended QuadCover algorithm with constraints can be applied in different ways. The user can prescribe geometrical constraints by providing a set of curves δ_κ. A parameter line will snap to each curve (see Fig. 4.20, top).

A very nice application is that we can also skip the geometrical constraint, i.e. obmit the linear constraints in step 6 of Algorithm 7. The geometrical alignment of the parameter line is then fully ignored, but the topology is enforced, i.e. if the path $\delta\kappa$ connects two singularities, then the parameterization contains a parameter line connecting them as well (see Fig. 4.20, f). If δ_κ is closed, then the parameterization will contain a closed parameter line which is homotopic to δ_κ, avoiding a spiraling effect as shown in Fig. 4.21.

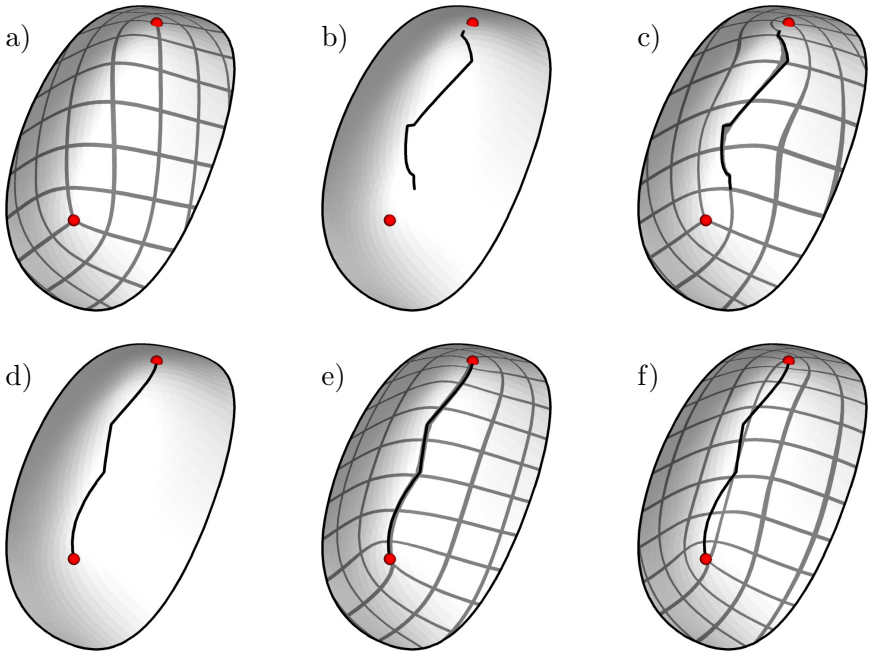


FIGURE 4.20: Demonstration on using constraints in QUADCOVER. All parameterizations use the same guiding frame field. a) Parameterization without any constraints. The singularities (red) are not connected by a parameter line. b) Arbitrary (jaggy) curve δ_0 drawn by the user. c) Parameterization with δ_0 as geometrical constraint. No combinatorial constraints are enforced. d) Arbitrary curve δ_1 connecting the singularities. e) Parameterization with δ_1 as geometrical constraint. It involves a combinatorical constraint which enforces that both singularities are connected by a parameter line. f) Parameterization using only the combinatorial constraint induced by δ_1. The parameterization is overall smooth and does not depend on the geometry of δ_1.

We have now established a general framework for handling constraints: The user can provide a set of constraint paths δ_κ (called initial meta-graph). For each path, he defines if it is a combinatorial constraint or even a geometrical constraint. The meta-graph may e.g. consist of geometrical features of the surface (as sharp creases that should be maintained) or just be an initial decomposition of the surface in order to keep the parameterization more simple. A good meta-layout helps to generate coarser parameterizations.

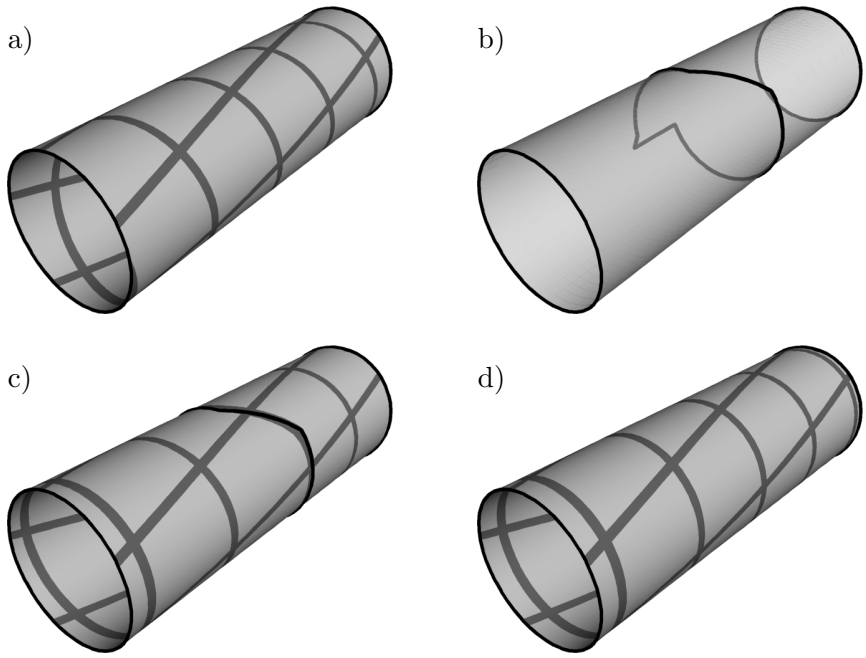


FIGURE 4.21: Parameterized cylinder using a twisted frame field. a) Parameterization without constraints. Parameter lines form a spiraling pattern. b) Constraint path δ_0 which forms a closed loop. c) Parameterization with geometrical constraints. d) Parameterization with combinatorial constraints only. Notice how the lines in axis direction remain unaffected by the constraint, whereas the traversal lines close up.

Chapter 5

Parameterization Results

This section shows some results of parameterized and remeshed surfaces using QuadCover. The algorithm has been implemented in Java using the *JavaView* software library [JavaView]. Results are shown in Section 5.1 whereas Section 5.2 compares QuadCover with different approaches. Limitations and future work are regarded in Section 5.3.

5.1 Examples of QuadCover and HexCover

The QuadCover technique was tested on various kinds of shapes. In general, it produces highly regular and smooth patterns.

Figure 5.1 shows a quadrangular parameterization on the rockerarm model. The frame field is visualized using a LIC technique for frame fields, introduced by Palacios and Zhang [Palacios 2011]. Singularities are highlighted in blue (index of $-1/4$) and yellow (index of $1/4$).

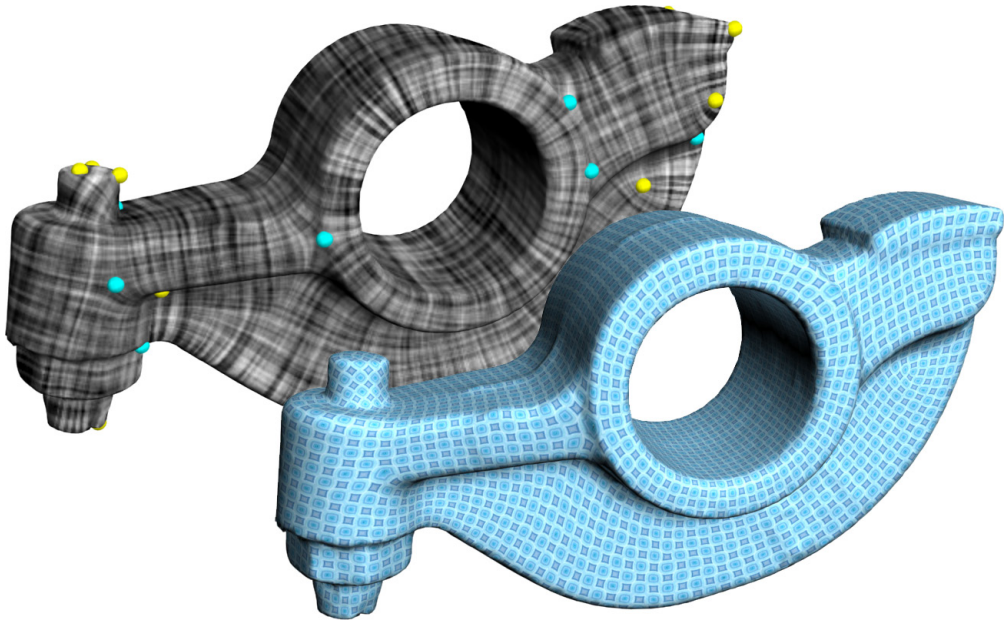


Figure 5.1: Geometry-aware 4-RoSy field and corresponding texture tiling.

The pegaso model (Fig. 5.2) is a closed surface of genus 6. Notice that the parameterization contains isolated triangles and pentagons. They correspond to a singularity which lies in the center of the element. As explained in Section 2.4.5, a pure quad mesh can always be achieved by generating a parameterization which is twice as coarse and by performing one subdivision step which bisects each n-gon into n quadrangles.

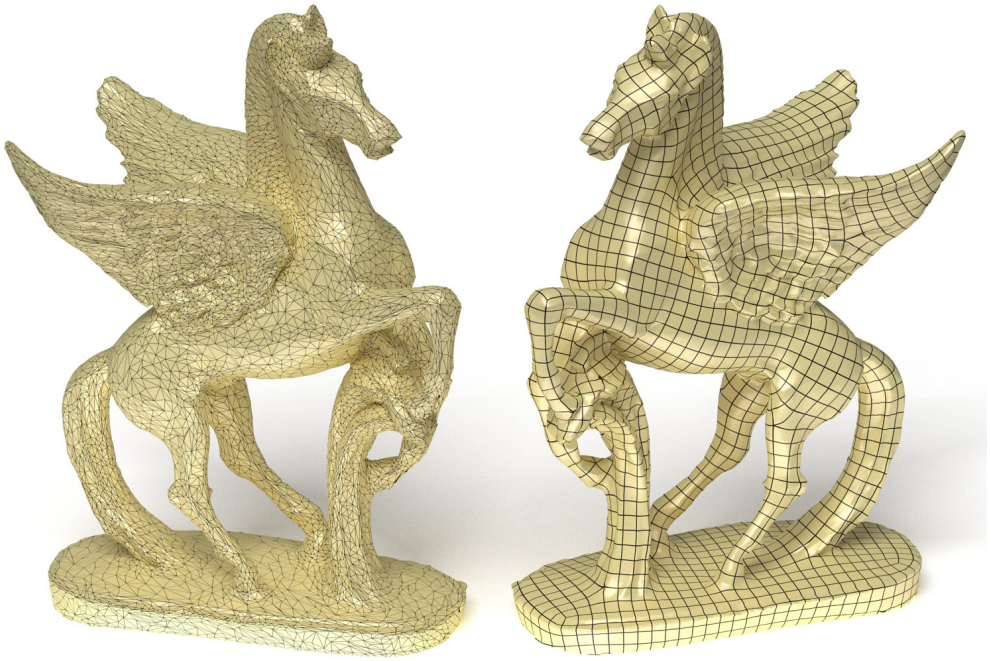


FIGURE 5.2: Pegaso. Left: Input triangle mesh. Right: Quadrilateral parameterization.

Hexagonal parameterizations are shown in Fig. 5.3. Figure 5.4 demonstrates the ability of QUADCOVER to generate meshes with adaptive sized elements. The vectors of the guiding frame field are scaled by some arbitrary user-defined scaling function. In order to obtain a finer resolution in curved regions, the scaling function can be locally defined according to the curvature in each point.

The control over singularities is an important design element for generating parameterizations. Figure 5.6 shows that the number of singularities impacts the smoothness of the parameterization with more singularities reproducing more feature details of the surface. However metric distortion also increases if more singularities are used as can be represented with the actual mesh resolution (see Fig. 5.6. Choosing the number of singularities can be considered as a tradeoff between smoothness of mesh elements and feature preservation.

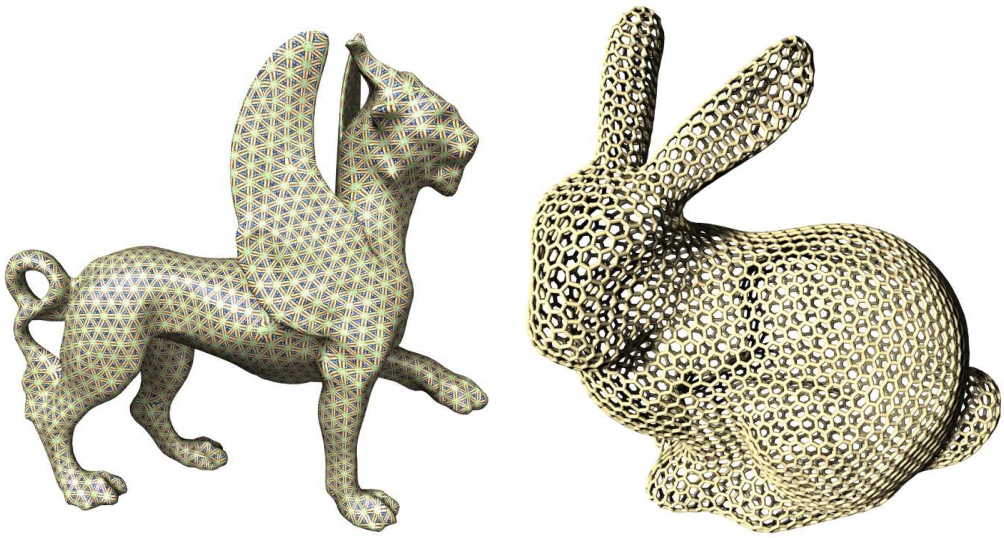


FIGURE 5.3: Hexagonal parameterization visualized with a texture (left) and using a geometry pattern (right).

5.2 Comparison to Other Techniques

Quadrangular Remeshing. Comparisons to other parameterization approaches show that QUADCOVER produces very competitive results (see Fig. 5.7). As the figure illustrates the curvature alignment of [Ray 2006] has also been achieved by QUADCOVER, but the number of irregular points is drastically reduced and the mesh is restricted to quadrilaterals. In contrast to the methods of [Tong 2006] and [Dong 2006], QUADCOVER is suited to handle arbitrary locations of branch points, as it does not restrict the branch points to be the corners of some coarse meta-mesh.

Table 5.1 shows that QUADCOVER exhibits the smallest edge length variation, at the cost of higher angular deviation. The angular deviation can be reduced significantly by discarding the curvature alignment term in the smoothing step, see Sect. 3.2. But then, more wrinkles in the final quad mesh are introduced where parameter lines do not follow high curvature.

	vertices	irreg. vert.	RSD edge	RSD angle
[Ray 2006]	6355	314	25.0%	10.7%
[Tong 2006]	6576	34	28.3%	12.6%
[Dong 2006]	7202	26	30.8%	7.8%
QUADCOVER	6535	37	18.2%	14.8%

TABLE 5.1: The number of total and irregular vertices of the models shown in Fig. 5.7, as well as the relative standard deviation of their edge lengths and vertex angles.

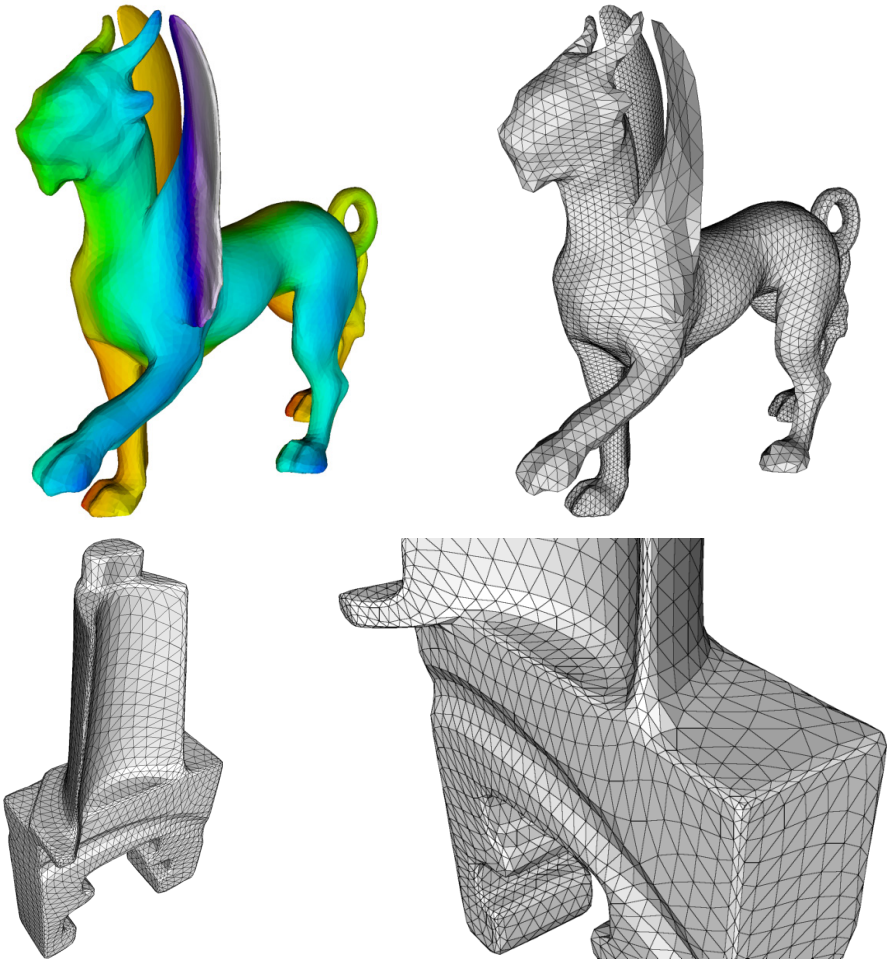


FIGURE 5.4: Adaptive sizing of triangles. Top: Feline model. Scaling function is linear along the y-axis (left). The triangles adapt to this varying scale (right). Bottom: Blade model. Frames are scaled by the absolute maximal principal curvature value.

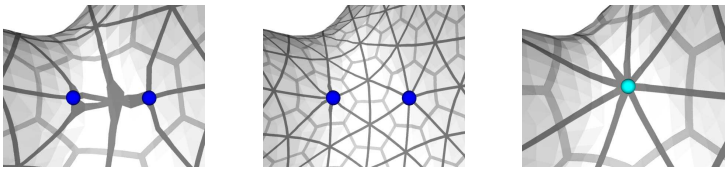


FIGURE 5.5: Singularities which are closer than the grid size may force the parameterization to degenerate locally (left). This artifact can be avoided by either choosing a finer grid size (middle) or by merging nearby singularities (right), e.g. as described in Section 3.3.

A comparison with the method from Tong et al. [Tong 2006] is shown in Fig. 5.8.

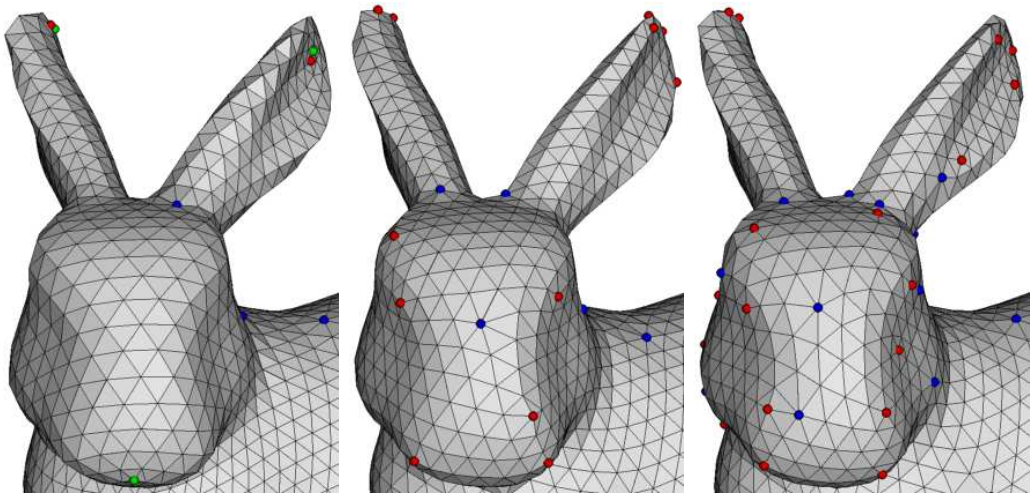


FIGURE 5.6: Triangular remeshing with 23, 65, and 151 singularities.

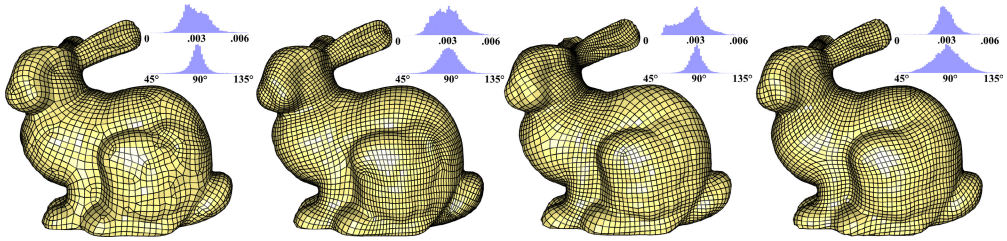


FIGURE 5.7: Comparison of remeshing results of the Stanford bunny. Models were produced by [Ray 2006], [Tong 2006], [Dong 2006] and QUADCOVER. The upper histogram next to each model shows the distribution of edge lengths, the lower histogram represents angle distribution.

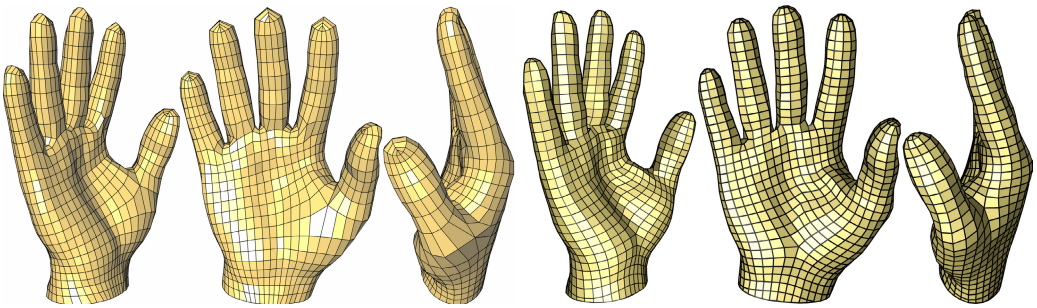


FIGURE 5.8: Comparison of the remeshed hand model, remeshed with [Tong 2006] (left 3 images) and with QUADCOVER (right 3 images). The hand model is courtesy of Pierre Alliez. Except from setting preprocessing parameters, no interaction was involved.

Triangular Remeshing. For triangular remeshing, Figure 5.9 compares the results of the foot and Venus models with HexCover with that of [Alliez 2002] and [Yan 2009]. The frame field for the HexCover parameterization was produced with the automatic singularity clustering method described in [Nieser 2011a].

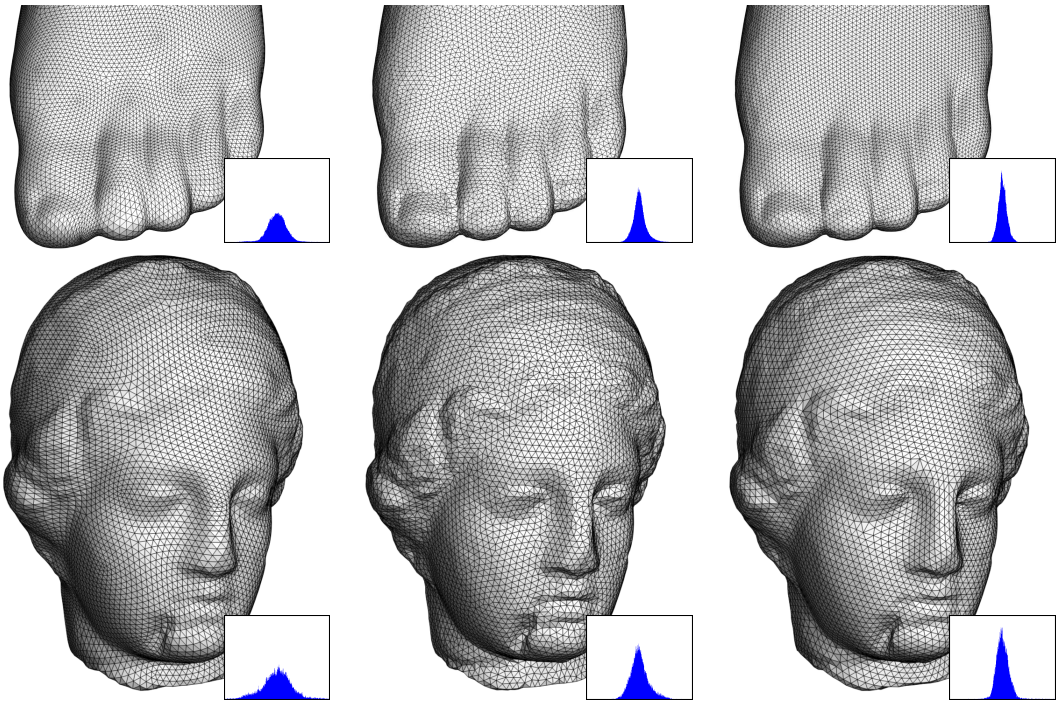


Figure 5.9: Comparison of HexCover (right) to those from [Desbrun 2002] (left) and [Yan 2009] (middle). The histograms show occurring inner angles (on the X-axis from 0 to $\pi/3$. For each model, the scale on the Y-axis is the same.

	vertices	irreg. vert.	RSD edge	RSD angle
Foot [Desbrun 2002]	23968	146	24.6%	19.9%
Foot [Yan 2009]	20000	3287	11.2%	12.3%
Foot HexCover	26081	13	10.5%	8.5%
Venus [Desbrun 2002]	6541	38	32.5%	29.1%
Venus [Yan 2009]	6000	1449	16.2%	17.3%
Venus HexCover	5791	36	13.9%	11.4%

Table 5.2: The number of total and irregular vertices of the models shown in Fig. 5.9, as well as the relative standard deviation of their edge lengths and vertex angles.

Table 5.2 provides the quality statistics of all tested models and the comparison. Notice that the HexCover method has better overall triangle aspect ratios (larger minimum angle, smaller maximum angle, and small standard deviation of angles)

than [Alliez 2002]. In addition, it tends to produce edge directions that better align with the features in the mesh (such as along the nose of the Venus).

Mixed Integer Quadrangulation. The mixed integer quadrangulation (MIQ) algorithm [Bommes 2009] is an approach based on QuadCover. The MIQ approach proposes a different way of generating a frame field from a sparse set of vectors. After field generation, the MIQ algorithm then also aligns the parameterization to a guiding frame field by minimizing the energy functional from Eqn. (2.4).

Both approaches consider a similar space of parameterizations, but use a different representation. The formulation of the MIQ approach uses the parameter values $\varphi_j(p_i) \in \mathbb{R}^2$ of all vertices p_i in each adjacent triangle t_j ($6 \cdot |T|$ many variables). Since there is huge redundancy, a constraint is defined per edge which assures a constant gap along the edge. The constraints are resolved using Lagrange multipliers, the system to solve has therefore dimension $6 \cdot |T| + 2 \cdot |E|$.

At this point, the representation used by QuadCover has an advantage since it explores a minimal set of variables: the gaps $w_{u,k}, w_{v,k} \in \mathbb{Z}$ and coefficients $u_i, v_i \in \mathbb{R}$ for all regular vertices (see Section 2.4.5). The total number of variables is $2 \cdot |V| + 4g - 2$ (where g is the genus of the surface). No constraints are needed since all redundancy is eliminated.

Also the formulation of the mixed integer problem is similar in both approaches: In MIQ, the integer variables are the parameter values at singularities whereas QuadCover assures integer gaps. As stated in Theorem 2.4.4, the coordinates of singularities are uniquely determined by the $w_{u,k}, w_{v,k}$. Assuring integer parameter values at singularities is equivalent to assuring integer gaps.

The main difference between both methods is the rounding heuristic for minimizing the energy under integer constraints. Instead of directly rounding all integer variables at once as done in QuadCover (see Section 2.4.5), the MIQ solver iterates between rounding an integer variable and solving the system while rounded integer variables are held fix.

In general, the MIQ rounding strategy should provide a better approximation to the solution of the NP-hard integer problem than the direct rounding technique does. However, all our tests show that the enhancement is relatively little and both rounding techniques produce very similar results. We conjecture that the reason behind this is the use of a shortest cut graph G. It appears that shorter paths γ_k give the variables w_k a more local influence, hence directly rounded integer variables become more accurate.

On the other side, computing a parameterization with QuadCover is much faster, since only two linear systems of equations need to be solved. The MIQ algorithm solves as many systems as there are integer variables (using different matrices each time). The dimension of the linear system is larger with more redundant variables and linear constraints. The fast rounding technique of QuadCover is especially strong if a parameterization is generated several times, e.g. for singularity opti-

mization as described in Section 3.3.

5.3 Limitations and Future Work

The parameterization approach presented in this thesis is highly robust and provides parameterizations which are very uniform and smooth by the click of a button.

One main issue is still unsolved which is the problem of flips or overfoldings. If the frame field contains much curl then the result of the local QuadCover stage is typically distorted. Similarly, if the vectors of the input field have a small scale (in order to produce especially coarse parameterizations), then the global QuadCover stage typically introduces distortion when rounding the integer variables. In both cases, if distortion becomes too high, the image of some triangles under the parameterization may flip, i.e. change their orientation, and the parameterization is no longer injective.

Flipped triangles cause problems when a parameterization is used for remeshing. Up to now, there is no method which guarantees a flip-free parameterization.

A very interesting topic is the generalization of QuadCover to 3D volumes, computing a regular hexahedral mesh from a given tetrahedral mesh (see Fig. 5.10). A first approach [Nieser 2011b] already defines the proper notion and formulates the parameterization algorithm CubeCover. However, the generation of suitable 3D frame fields turns out being far more challenging than in two dimensions and remains an open problem for future work.

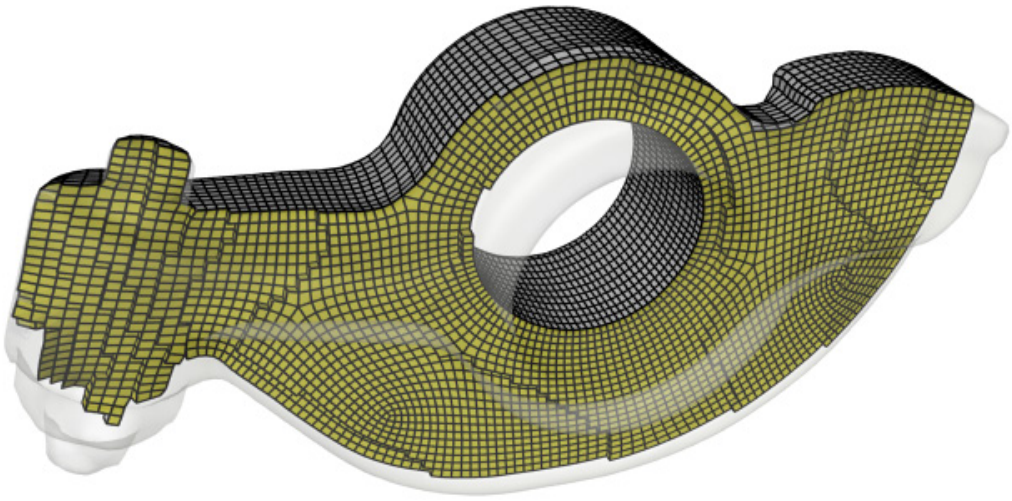


Figure 5.10: Inner volume of the rockerarm model, tessellated into hexahedra with CubeCover.

Bibliography

[Alliez 2002] Pierre Alliez, Mark Meyer and Mathieu Desbrun. *Interactive Geometry Remeshing*. In Proceedings of Siggraph, pages 347–354, 2002. 76, 77

[Ben-Chen 2008] Mirela Ben-Chen, Craig Gotsman and Guy Bunin. *Conformal Flattening by Curvature Prescription and Metric Scaling*. Computer Graphics Forum, vol. 27, no. 2, pages 449–458, 2008. 8

[Bommes 2009] David Bommes, Henrik Zimmer and Leif Kobbelt. *Mixed-Integer Quadrangulation*. In Proceedings of Siggraph, pages 77:1–77:10, 2009. 6, 8, 9, 43, 45, 61, 64, 77

[Cohen-Steiner 2003] David Cohen-Steiner and Jean-Marie Morvan. *Restricted Delaunay Triangulations and Normal Cycles*. Symposium on Computational Geometry, pages 237–246, 2003. 44

[Crane 2010] Keenan Crane, Mathieu Desbrun and Peter Schröder. *Trivial Connections on Discrete Surfaces*. Proceedings of SGP, vol. 29, no. 5, pages 1525–1533, 2010. 9

[Desbrun 2002] Mathieu Desbrun, Mark Meyer and Pierre Alliez. *Intrinsic Parameterizations of Surface Meshes*. In Proceedings of Eurographics, pages 209–218, 2002. 7, 76

[Dong 2006] Shen Dong, Peer-Timo Bremer, Michael Garland, Valerio Pascucci and John C. Hart. *Spectral Surface Quadrangulation*. In Proceedings of Siggraph, pages 1057–1066, 2006. 7, 73, 75

[Erickson 2005] Jeff Erickson and Kim Whittlesey. *Greedy Optimal Homotopy and Homology Generators*. In Symposium on Discrete Algorithms, pages 1038–1046, 2005. 26, 39

[Farkas 1980] Hershel M. Farkas. Riemann surfaces. Springer, New York, 1980. 30

[Fisher 2007] Matthew Fisher, Peter Schröder, Mathieu Desbrun and Hugues Hoppe. *Design of Tangent Vector Fields*. In Proceedings of Siggraph, 2007. 9

[Floater 1997] Michael S. Floater. *Parametrization and Smooth Approximation of Surface Triangulations*. Computer Aided Geometric Design, vol. 14, pages 231–250, 1997. 7

[Floater 2005] Michael S. Floater and Kai Hormann. *Surface Parameterization: a Tutorial and Survey*. In Advances in Multiresolution for Geometric Modelling, pages 157–186. 2005. 7

[Forster 1999] Otto Forster. Lectures on Riemann surfaces. Graduate Texts in Mathematics. Springer, fourth édition, 1999. 30

[Fulton 1995] William Fulton. Algebraic topology: a first course. Springer-Verlag, 1995. 15

[Gauß 1828] Carl Friedrich Gauß. *Disquisitiones Generales Circa Superficies Curvas*. 1828. 3

[Gu 2003] Xianfeng Gu and Shing-Tung Yau. *Global Conformal Surface Parameterization*. In Proceedings of SGP, pages 127–137, 2003. 7

[Haker 2000] Steven Haker, Sigurd Angenent, Allen Tannenbaum, Ron Kikinis, Guillermo Sapiro and Michael Halle. *Conformal Surface Parameterization for Texture Mapping*. Transactions on Visualization and Computer Graphics, vol. 6, pages 181–189, 2000. 7

[Hertzmann 2000] Aaron Hertzmann and Denis Zorin. *Illustrating Smooth Surfaces*. Proceedings of Siggraph, pages 517–526, 2000. 9

[Hildebrandt 2004] Klaus Hildebrandt and Konrad Polthier. *Anisotropic Filtering of Non-Linear Surface Features*. Computer Graphics Forum, vol. 23, no. 3, pages 391–400, 2004. 44

[Hildebrandt 2011] Klaus Hildebrandt and Konrad Polthier. *Generalized shape operators on polyhedral surfaces*. Computer Aided Geometric Design, vol. 28, no. 5, pages 321–343, 2011. 44

[Hormann 2000] Kai Hormann and Günther Greiner. *MIPS: An Efficient Global Parametrization Method*. In Curve and Surface Design: Saint-Malo 1999, pages 153–162. 2000. 7

[Hormann 2008] Kai Hormann, Konrad Polthier and Alla Sheffer. *Mesh Parameterization: Theory and Practice*. no. 11, pages v+81, 2008. 7

[JavaView] Konrad Polthier, Klaus Hildebrandt, Felix Kälberer, Matthias Nieser and Ulrich Reitebuch. *JavaView – Interactive 3D Geometry and Visualization Software*, 1999 – 2011. 71

[Jin 2004] Miao Jin, Yalin Wang, Shing-Tung Yau and Xianfeng Gu. *Optimal Global Conformal Surface Parameterization*. In Proceedings of VIS, pages 267–274, 2004. 7

[Kälberer 2007] Felix Kälberer, Matthias Nieser and Konrad Polthier. *QuadCover – Surface Parameterization using Branched Coverings*. Computer Graphics Forum, vol. 26, no. 3, pages 375–384, 2007. 6, 8, 9, 43

[Kälberer 2010] Felix Kälberer, Matthias Nieser and Konrad Polthier. *Stripe Parameterization of Tubular Surfaces.* In Topological Methods in Data Analysis and Visualization. Theory, Algorithms, and Applications., Mathematics and Visualization. Springer Verlag, 2010. 6, 8, 58

[Kharevych 2006] Liliya Kharevych, Boris Springborn and Peter Schröder. *Discrete Conformal Mappings via Circle Patterns.* Transactions on Graphics, vol. 25, pages 412–438, 2006. 7

[Lai 2010] Yu-Kun Lai, Miao Jin, Xuexiang Xie, Ying He, Jonathan Palacios, Eugene Zhang, Shi-Min Hu and Xianfeng Gu. *Metric-Driven RoSy Field Design and Remeshing.* Transactions on Visualization and Computer Graphics, vol. 16, pages 95–108, 2010. 9

[Lambert 1772] Johann Heinrich Lambert. *Anmerkungen und Zusätze zur Entwerfung der Land- und Himmelscharten.* 1772. 4

[Lamotke 2005] Klaus Lamotke. Riemannsche flächen. Springer, 2005. 30

[Lévy 1998] Bruno Lévy and Jean-Laurent Mallet. *Non-Distorted Texture Mapping for Sheared Triangulated Meshes.* In Proceedings of Siggraph, pages 343–352, 1998. 7

[Lévy 2002] Bruno Lévy, Sylvain Petitjean, Nicolas Ray and Jérome Maillot. *Least Squares Conformal Maps for Automatic Texture Atlas Generation.* In Proceedings of Siggraph, pages 362–371, 2002. 7

[Li 2006] Wan-Chiu Li, Bruno Vallet, Nicolas Ray and Bruno Levy. *Representing Higher-Order Singularities in Vector Fields on Piecewise Linear Surfaces.* Transactions on Visualization and Computer Graphics, vol. 12, pages 1315–1322, 2006. 9

[Munkres 1999] James Munkres. Topology. Prentice Hall, 1999. 15

[Needham 2000] Tristan Needham. Visual complex analysis. Oxford University Press, 2000. 30

[Nieser 2009] Matthias Nieser and Konrad Polthier. *Parameterizing Singularities of Positive Integral Index.* In Mathematics of Surfaces XIII, volume 5654, pages 265–277. Springer Verlag, 2009. 6, 8, 61

[Nieser 2011a] Matthias Nieser, Jonathan Palacios, Konrad Polthier and Eugene Zhang. *Hexagonal Global Parameterization of Arbitrary Surfaces.* Transactions on Visualization and Computer Graphics, vol. 99, no. PrePrints, 2011. 6, 8, 9, 43, 53, 76

[Nieser 2011b] Matthias Nieser, Ulrich Reitebuch and Konrad Polthier. *CubeCover – Parameterization of 3D Volumes.* In Proc. SGP, 2011. 78

[Palacios 2007] Jonathan Palacios and Eugene Zhang. *Rotational Symmetry Field Design on Surfaces.* In Proceedings of Siggraph, 2007. 9

[Palacios 2011] Jonathan Palacios and Eugene Zhang. *Interactive Visualization of Rotational Symmetry Fields on Surfaces.* Transactions on Visualization and Computer Graphics, vol. 17, pages 947–955, 2011. 9, 71

[Pietroni 2010] Nico Pietroni, Marco Tarini and Paolo Cignoni. *Almost Isometric Mesh Parameterization through Abstract Domains.* IEEE Transaction on Visualization and Computer Graphics, vol. 16, no. 4, pages 621–635, 2010. 7

[Polthier 2002] Konrad Polthier. *Polyhedral Surfaces of Constant Mean Curvature.* phdthesis, Technische Universität Berlin, 2002. Habilitation thesis. 14, 18

[Polthier 2003] Konrad Polthier and Eike Preuss. *Identifying Vector Field Singularities using a Discrete Hodge Decomposition.* In Visualization and Mathematics III, pages 113–134. Springer, 2003. 22

[Pottmann 2007] Helmut Pottmann, Johannes Wallner, Yong-Liang Yang, Yu-Kun Lai and Shi-Min Hu. *Principal Curvatures from the Integral Invariant Viewpoint.* Computer Aided Geometric Design, vol. 24, pages 428–442, 2007. 44

[Praun 2000] Emil Praun, Adam Finkelstein and Hugues Hoppe. *Lapped Textures.* In Proceedings of Siggraph, pages 465–470, 2000. 9

[Ray 2006] Nicolas Ray, Wan Chiu Li, Bruno Lévy, Alla Sheffer and Pierre Alliez. *Periodic Global Parameterization.* Transactions on Graphics, vol. 25, pages 1460–1485, 2006. 8, 73, 75

[Ray 2008] Nicolas Ray, Bruno Vallet, Wan Chiu Li and Bruno Lévy. *N-Symmetry Direction Field Design.* Transactions on Graphics, vol. 27, pages 10:1–10:13, 2008. 9

[Ray 2009] Nicolas Ray, Bruno Vallet, Laurent Alonso and Bruno Lévy. *Geometry Aware Direction Field Processing.* Transactions on Graphics, 2009. 9

[Riemann 1851] Bernhard Riemann. *Grundlagen für eine allgemeine Theorie der Functionen einer veränderlichen complexen Grösse.* PhD thesis, Göttingen, 1851. 30

[Sheffer 2001] Alla Sheffer and E. de Sturler. *Parameterization of Faceted Surfaces for Meshing using Angle-Based Flattening.* Engineering with Computers, vol. 17, pages 326–337, 2001. 7

[Sheffer 2006] Alla Sheffer, Emil Praun and Kenneth Rose. *Mesh Parameterization Methods and their Applications.* Foundations and Trends in Computer Graphics and Vision, vol. 2, pages 105–171, 2006. 7

[Springborn 2008] Boris Springborn, Peter Schröder and Ulrich Pinkall. *Conformal Equivalence of Triangle Meshes.* In Proceedings of Siggraph, pages 77:1–77:11, 2008. 8

[Stephenson 1999] Kenneth Stephenson. *Approximation of Conformal Structures via Circle Packing.* In Proceedings of CMFT, volume 11, pages 551–582, 1999. 7

[Tarini 2004] Marco Tarini, Kai Hormann, Paolo Cignoni and Claudio Montani. *PolyCube-Maps.* ACM Trans. Graph., vol. 23, no. 3, pages 853–860, 2004. 7

[Theisel 2002] Holger Theisel. *Designing 2D Vector Fields of Arbitrary Topology.* Computer Graphics Forum, vol. 21, pages 595–604, 2002. 9

[Tong 2006] Yiying Tong, Pierre Alliez, David Cohen-Steiner and Mathieu Desbrun. *Designing Quadrangulations with Discrete Harmonic Forms.* In Proceedings of SGP, pages 201–210, 2006. 6, 7, 61, 73, 74, 75

[Turk 2001] Greg Turk. *Texture Synthesis on Surfaces.* Proceedings of Siggraph, pages 347–354, 2001. 9

[Tutte 1963] William T. Tutte. *How to Draw a Graph.* Proceedings London Mathematical Society, vol. 13, no. 1, pages 743–767, 1963. 7

[Wardetzky 2006] Max Wardetzky. *Discrete Differential Operators on Polyhedral Surfaces – Convergence and Approximation.* phdthesis, Freie Universität Berlin, 2006. phd thesis. 22

[Wei 2001] Li-Yi Wei and Marc Levoy. *Texture Synthesis over Arbitrary Manifold Surfaces.* Proceedings of Siggraph, pages 355–360, 2001. 9

[Yan 2009] Dong-Ming Yan, Bruno Lévy, Yang Liu, Feng Sun and Wenping Wang. *Isotropic Remeshing with Fast and Exact Computation of Restricted Voronoi Diagram.* In Proceedings of SGP, pages 1445–1454, 2009. 76

[Zadravec 2010] Mirko Zadravec, Alexander Schiftner and Johannes Wallner. *Designing Quad-dominant Meshes with Planar Faces.* Proceedings of SGP, vol. 29, no. 5, pages 1671–1679, 2010. 8, 9

[Zhang 2006] Eugene Zhang, Konstantin Mischaikow and Greg Turk. *Vector Field Design on Surfaces.* Transactions on Graphics, vol. 25, pages 1294–1326, 2006. 9

[Zhang 2007] Eugene Zhang, James Hays and Greg Turk. *Interactive Tensor Field Design and Visualization on Surfaces.* Transactions on Visualization and Computer Graphics, vol. 13, pages 94–107, 2007. 9

[Zhang 2010] Muyang Zhang, Jin Huang, Xinguo Liu and Hujun Bao. *A Wave-Based Anisotropic Quadrangulation Method.* Proceedings of Siggraph, pages 118:1–118:8, 2010. 8, 9, 64

Zusammenfassung

Die vorgelegte Arbeit beschäftigt sich mit dem Problem der Parametrisierung von simplizialen Flächen, d.h. deren Abbildung in die Ebene. Dabei sollen sich die Parameterlinien an vorgegebenen Richtungsfeldern orientieren, häufig den Hauptkrümmungsrichtungen der Fläche.

Richtungsfelder sind mehrwertige Funktionen. In dieser Arbeit wird die Äquivalenz zu einem (ein-wertigen) Vektorfeld auf einer Riemannschen Fläche gezeigt und damit die theoretische Grundlage für das QuadCover-Verfahren gelegt.

QuadCover generiert automatisch globale Parametrisierungen von beliebigen zweidimensionalen polyedrischen Mannigfaltigkeiten. Das resultierende Gitter aus Parameterlinien ist global stetig und erlaubt eine Neuvernetzung der Fläche in ein hochqualitatives Netz aus reinen Vierecken, Dreiecken oder Sechsecken.

Eine weitere Anwendung von QuadCover ist das Texturieren von Oberflächen. Diese kann mit beliebigen kachelbaren Mustern überzogen werden. Dabei muss die Parametrisierung mit der Rotationssymmetrie des Musters kompatibel sein. In der Arbeit wird gezeigt, wie mithilfe von QuadCover eine Fläche mit Vierecks-, Sechsecks-, Dreiecks- oder Streifenmuster gekachelt werden kann.

Ein Benutzer hat häufig spezielle Vorstellungen an eine gute Parametrisierung, so dass es notwendig ist, dass der Benutzer in den Parametrisierungsprozess eingreifen kann. QuadCover ermöglicht das manuelle Platzieren und Verschieben von Singularitäten auf der Fläche. Desweiteren können Kurven auf der Fläche vorgegeben werden, die geometrische Nebenbedingungen definieren, d.h. die Parametrisierung richtet sich exakt an der gegebenen Kurve aus. Ausserdem kann durch Vorgabe von kombinatorischen Nebenbedingungen Einfluss auf die Netztopologie genommen werden.