Parameterizing Singularities of Positive Integral Index

Matthias Nieser, Konrad Polthier *

Abstract

Classical surface parameterization algorithms often place singularities in order to enhance the quality of the resulting parameter map. Unfortunately, singularities of positive integral index (as the north pole of a sphere) were not handled since they cannot be described with piecewise linear parameter functions on a triangle mesh. Preprocessing is needed to adapt the mesh connectivity. We present an extension to the QuadCover parameterization algorithm [KNP07], which allows to handle those singularities.

A singularity of positive integral index can be resolved using bilinear parameter functions on quadrilateral elements. This generalization of piecewise linear functions for quadrilaterals enriches the space of parameterizations. The resulting parameter map can be visualized by textures using a rendering system which supports quadrilateral elements, or it can be used for remeshing into a pure quad mesh.

1 Introduction

Classical algorithms for surface parameterization often use a global map, which flattens a given surface and transfers it to a 2d parameter domain. During the last years, approaches came up which allow the placement of singularities (or cone points). Singularities are necessary in order to minimize overall distortion of the parameterization.

For a given parameterization method, the amount of distortion is mainly determined by the location and type of singularities. Typical algorithms first

*Supported by DFG Research Center Matheon "Mathematics for key technologies"

place some singularities and hold them fixed during the subsequent optimization. An accurate singularity placement belongs to the main problems for parameterization.

The QuadCover algorithm also works in this manner. The singularities are taken from a given input frame field (e.g. from principle curvatures).

A special case arises for singularities of index +1. They cannot be resolved by a piecewise linear parameter function, since this would require its gradient to increase to infinity. All existing parameterization algorithms using PL triangle functions, have to face the same problem.

1.1 Previous work

Surface parameterization is an active research area. We will shortly discuss early and recent work which are closely related. More complete lists can be found in [HPS08, FH05].

Early global parameterization methods were introduced by Haker, Gu and Yau [HAT+00, GY03] and others. They studied conformal parameterizations which preserve angles at the cost of possibly large length distortions. Angles and lengths can not be preserved at the same time on general surfaces.

Other methods like Tong et al. [TACSD06] or Lai at al. [LJX+08] allow singular points which enlarge the space of harmonic functions used for parameterization. A good placement of singular points is an ongoing problem. [BCGB08] and [SSP08] present two approaches for an automatic placement of singularities which are suited for conformal parameterizations.

With the QuadCover algorithm [KNP07] we built upon an idea from Ray et al. [RLL+06], which use two orthogonal input fields as guiding directions for the parameter lines. The singularities are then taken from the guiding field. The idea of QuadCover is to find a parameterization whose gradient matches this field as well as possible.

1.2 Contributions

Typical parameterization algorithms use piecewise linear parameter functions on triangle meshes. Unfortunately, this function space is too rigid for describing vortex-like singularities. It is impossible to represent a vortex with piecewise linear texture maps on a triangle grid.

We propose a procedure for handling singularities of positive integer index. We therefore introduce bilinear texture maps on quadrilateral elements

and use them for the construction of such a singularity. The resulting parameterization can be visualized by rendering systems which support bilinear textures on quads. Independent from rendering, the parameterization can still be used for quad remeshing.

We do not give an algorithm for placing singularities. As in the classical QuadCover algorithm [KNP07], they are just taken as the singularities of the input field. The construction of a good input field is still an unsolved problem.

The paper is structured as follows: The overview and the main ideas of QuadCover are outlined in Sect. 2. The algorithm intensively uses branched covering spaces. A short introduction into coverings is given in Sect. 3. Sect. 4 describes QuadCover in detail.

We show the relation between the index of a frame field singularity and a branch point of the covering. We introduce the problem which occurs with singularities of positive integral index. We then describe an extension of the QuadCover algorithm which solves the problem (Sect. 5). Finally, Sect. 6 shows some results of the extended algorithm.

2 Overview

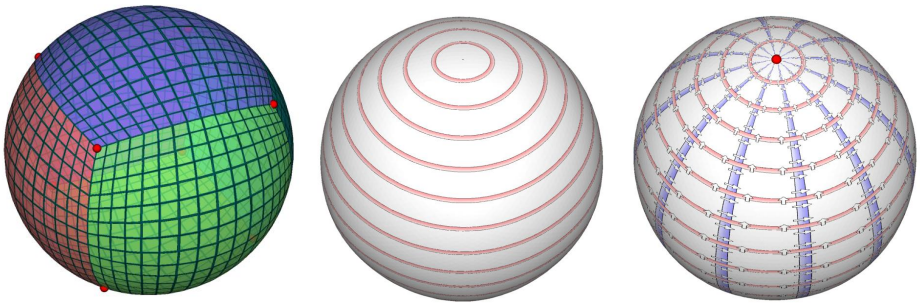

Figure 1: Different parameterizations of a sphere. **Left:** Classical algorithms may produce 8 singularities which are equally distributed. **Middle:** Use a radial guidance frame field in QuadCover. Since index 1 singularities cannot be handled in the classic way, the resulting u component nearly vanishes everywhere. Only the v component behaves well. **Right:** The two singularities of index 1 are correctly resolved with the presented method.

Given a smooth manifold M with an atlas of charts $\{U_i\}$. A global parameterization $f_i : U_i \to \mathbb{R}^2$ maps all charts into a flat (u, v)-domain. The

parameter lines are the preimages of the unit grid $\mathbb{Z} \times \mathbb{R}$ (u_i lines), $\mathbb{R} \times \mathbb{Z}$ (v_i lines). They induce a quadrilateral structure in each chart.

Whenever charts overlap, the parameter functions are related by transition functions ξ_{ij}. In order to ensure, that the quad structure is globally continuous, we restrict all transition functions to leave the unit grid invariant. It is constrained to be of the form

$$\xi_{ij}(u, v) := f_j(f_i^{-1}(u, v)) = J^{r_{ij}} \begin{pmatrix} u \\ v \end{pmatrix} + d_{ij}, \quad r_{ij} \in \mathbb{Z}, d_{ij} \in \mathbb{Z}^2, \qquad (1)$$

where J is the rotation by 90 degrees in counter-clockwise direction. We call these maps *grid automorphisms*. The numbers r_{ij} decide, whether the u lines in chart U_i correspond to u, v, $-u$ or $-v$-lines in U_j. They are called *matchings* between the charts. The vectors d_{ij} encode a translational offset and are called *gaps* (see Fig. 2).

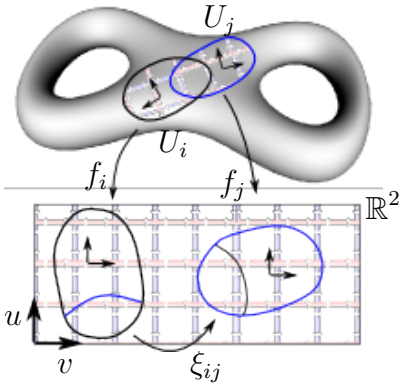

Figure 2: Smooth manifold with two charts and matching $r_{ij} = 1$.

Frame fields. The parameterization is guided by a so-called *frame field*. In each chart, it is just a collection of two vector fields. The goal is to find a parameter map, whose gradients matches up with the given frames as well as possible.

The gradients of the parameter map in different charts are related by:

$$\begin{pmatrix} \nabla u_j \\ \nabla v_j \end{pmatrix} = J^{r_{ij}} \begin{pmatrix} \nabla u_i \\ \nabla v_i \end{pmatrix} \qquad (2)$$

Thus, the ordered list $(\nabla u, \nabla v, -\nabla u, -\nabla v)$ in U_j is the same as in U_i, but cyclically shifted $-r_{ij}$ times. In general, the gradients therefore do not describe global continuous vector fields on M. Whenever u- and v-gradients

are flipped in different charts, the corresponding vectors must be identified. It turns out, that there is an elegant way to describe this setting using vector fields on covering spaces (see Sect. 3).

The algorithm. The QuadCover algorithm takes any frame field as input and generates a parameter function whose gradients matches up with the input field as well as possible. In practice, the principle curvature directions are often used as starting frame field, but the user can start with any field. Details about the algorithm are given in Sect. 4.

The resulting coordinates u and v can be used as texture map. Because of the coupling of u and v, the texture pattern has to be symmetric due to a rotation by 90 degrees. E.g. if you use an image containing a quadrangular grid, then the surface gets divided into quadrangles.

Special issues arise, when there are singularities. When tracing a small cycle around a singularity p, the frame vectors turn by either a whole number of revolutions (singularity of integral index) or just by a multiple of 90 degrees (singularity of fractional index $k/4$, $k \in \mathbb{Z}$). The singularities with fractional index can be naturally described with branch points in a covering space as described in Sect. 4.

Singularities of index $k \in \mathbb{N}$ in general cannot be represented with standard piecewise linear functions. The nature of these singularities is that the parameter function in its vicinity tends to infinity (e.g. the gradients of the longitudinal part of the polar parameterization on a sphere). It turns out, that the space of PL functions is not flexible enough to represent such singularities. A key idea is to integrate additional quadrilateral elements and therefore extend the space of PL functions to functions which are linear on each triangle and bilinear on each quad. As for PL functions, this is a fully consistent space of continuous functions, even if triangles and quads are mixed in the same geometry. The idea and the algorithm will be presented in Sect. 5.

3 Frame Fields

This section gives a formal definition of frame fields and describes their relation to covering spaces.

Definition 1 *Given a manifold M with charts U_i and matchings r_{ij}. A* frame field *on M is a collection of two vector fields $K_{i,0}$, $K_{i,1}$ in each chart*

U_i. Let $K_{i,2} := -K_{i,0}$, $K_{i,3} := -K_{i,1}$. All overlapping charts $U_i \cap U_j \neq \emptyset$ must hold: $K_{j,m} = K_{i,(m-r_{ij}) \bmod 4}$, $\forall m \in \{0,1,2,3\}$.

The algorithm uses the notion of branched covering surfaces for an equivalent description of frame fields as explained below.

3.1 Branched Covering Spaces

A frame field on the input surface can be seen as two vector fields on a covering surface. The advantage of this notion is, that standard vector field calculus can now be applied to frame fields.

Coverings. First, recall some definitions about Riemann surfaces, see [FK80, Jos02]. We first give an abstract definition of a covering and explain below how we actually construct one.

Definition 2 *Let M be a Riemann surface. A 4-sheeted covering M' of M is a Riemann surface with a local homeomorphism $\pi : M' \to M$ and the property: For each point $p \in M$, there exists a neighborhood U_p whose preimage $\pi^{-1}(U_p)$ is the union of exactly four pairwise disjoint topological disks. Fig. 3, left shows a 4-sheeted covering.*

In our setting, we allow some exceptional points p (branch points), where the preimage of a neighborhood of p is the union of less than four topological disks (e.g. as in Fig. 3, right)).

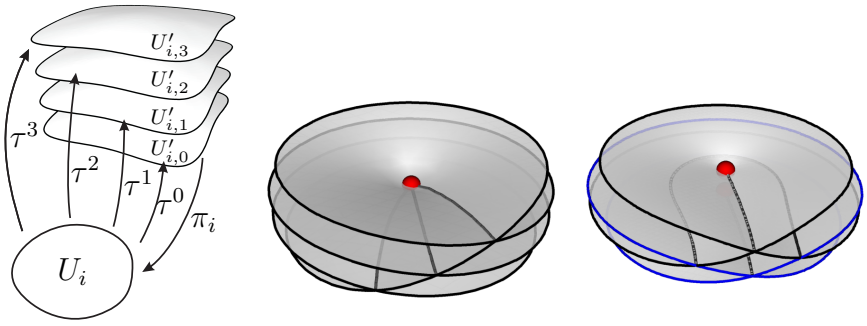

Figure 3: **Left:** Trivial 4-sheeted covering. **Right:** Different branch points.

Construction. We construct a covering of M as follows: From each chart U_i, make four copies (*layers*) and name them $U'_{i,k}$, $k \in \{0,1,2,3\}$. Let

$\pi_i : \bigcup_k U'_{i,k} \to U_i$ be the operator which projects the copies back to U_i and $\tau_{i,k} : U_i \to U'_{i,k}$ the inverse maps. The four layers $U'_{i,k}$ together with π_i is called the *4-sheeted trivial covering* of the chart U_i (Fig. 3, left).

In the next step, we glue these layers at the overlaps of the adjacent charts together. For each pair of charts the layers may be glued in four different ways, which is defined by the corresponding matching r_{ij}.

Definition 3 *A* covering surface induced by given matchings r_{ij} *is uniquely defined by the following construction:*

Let (U'_i, π_i) be 4-sheeted trivial coverings of all charts U_i. The covering surface is given as the union of all U'_i where the following points are identified: For all overlapping charts U_i, U_j, identify the points $\tau_i^k(p)$ with $\tau_j^{(k-r_{ij}) \bmod 4}(p)$, $p \in U_i \cap U_j$, $k \in \{0,1,2,3\}$ (see Fig. 4, left).

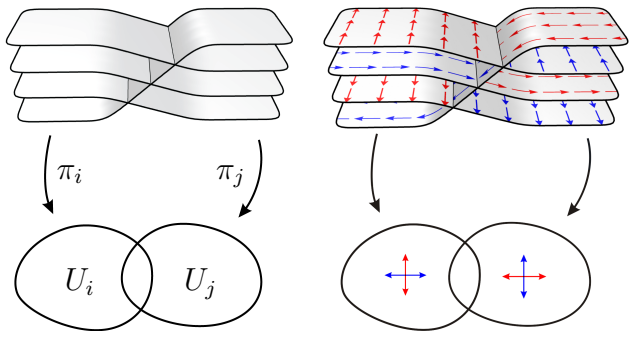

Figure 4: **Left:** Patching two coverings together with matching $r_{ij} = 1$. **Right:** A frame field lifted to a vector field on the covering.

Since the trivial coverings of charts have no branch points and the charts cover the surface, we cannot get any branch point by this construction. Instead, they can be invented by removing single points from the surface. Depending on the matchings of charts in its vicinity, the covering could be extended to a branched covering there.

3.2 Vector Fields on Covering Spaces

In this section, we show how frame fields can be described as vector fields on a covering surface. It allows us to apply the classical theory for vector fields to frame fields.

A frame field $(K_{i,0}, K_{i,1})$ on M with matchings r_{ij} canonically lifts to two vector fields (K'_0, K'_1) on the covering which is induced by r_{ij}. In each chart, define the vectors on its trivial covering as follows: For all $p \in U'_{i,k}$ set $K'_0(p) := K_{i,k}(\pi_i(p))$ and $K'_1(p) := K_{i,(k+1)\bmod 4}(\pi_i(p))$, $k \in \{0, 1, 2, 3\}$.

The result are two globally well defined vector fields K'_0, K'_1 on M', since the layers of the covering are connected in the same way as the vector fields permute when another chart is chosen (compare Def. 1 and 3, Fig. 4, right).

Moreover, the vector fields are symmetric on the layers. I.e. if $K'_0(p)$ and $K'_1(p)$ are the vectors in a given point in layer 0, then the vectors in all other layers are: $(K'_1(p), -K'_0(p))$, $(-K'_0(p), -K'_1(p))$ and $(-K'_1(p), K'_0(p))$.

Discretization. Each triangle of the mesh is considered as a chart. The transition function between two adjacent triangles is fully determined by the matching and the translation vector associated to their common edge (see Eqn. (1)).There is no need to really compute the covering surface. It is implicitly given by the matchings.

Branch points are located at vertices. Let p be a vertex and all elements in its star are enumerated clockwise from 0 to $n-1$. The matchings at incident edges to p are given by $r_{i,(i+1) \bmod n}$. Let $t_p := ((\sum_i r_i) \bmod 4)$ be the *type* of the vertex. Branch points are characterized by $t_p \neq 0$. This means, starting somewhere in the neighborhood of p and walking around the vertex ends on a different layer in the covering. Fig. 3 shows branch points of type 1 and 2.

Discrete frame fields are piecewise constant. In each triangle, two vectors are stored. Together with the matching numbers, a discrete frame field is thereby uniquely described.

4 QuadCover Parameterization Algorithm

This section explains shortly, how the QuadCover algorithm works. For further details, refer to [KNP07].

Compute potential function. Given a surface M together with a frame field $(K_{i,0}, K_{i,1})$. Equivalently, given a covering surface M' with two vector fields (K'_0, K'_1). The parameterization $(u', v') : M' \to \mathbb{R}^2$ can be projected back to $(u, v) : M \to \mathbb{R}^2$ by taking the values in one of the layers. It does not matter which one, because the parameter lines in all layers will be congruent.

First pass. QuadCover uses a variational approach in order to find a parameterization which fits best possible to the given frame field. More

precisely, the energy $E(u', v') = \int_{M'} (\|\nabla u' - K_0'\|^2 + \|\nabla v' - K_1'\|^2)\, dA$ gets minimized. The functions u', v' live in a space of PL functions, which may be discontinuous at the edges (because of Eqn. (1)). The difference of function values at both sides of an edge must be a constant (called *gap* at an edge).

In practice, u' (and equivalently v') is found using a discrete Hodge-Helmholtz decomposition of the vector fields K_0' (resp. K_1'). They can uniquely be written as $K_k' = P_k + C_k + H_k$, $k \in \{0, 1\}$ with a potential P_k, a copotential C_k and a harmonic vector field H_k. Any pair of functions u', v' satisfying $\nabla u' = P_0 + H_0$, $\nabla v' = P_1 + H_1$ minimizes the energy.

QuadCover integrates the vector fields $P_k + H_k$ in each chart (triangle) of the covering separately. The exact translation constant is left open at this stage. The resulting map (u', v') is in general not injective, the images of triangles may overlap. The common edge between adjacent triangles (of the covering) is by construction parallel and of same length in texture space.

Second pass. For getting a valid global parameterization, it remains to ensure that all gaps are $\in \mathbb{Z}^2$ (Eqn. (1)). QuadCover decreases the degrees of freedom by translation, such that all triangles are connected. Thus, all gaps become 0, except at few edges which form a cut graph of the mesh. A cut graph is a set of paths γ_i which cut the surface to a topological disk.

The second pass is based on the following observation: along each path γ_i of this cut graph, the gap is always a constant d_i. This is because the derivative of the function is locally integrable. Note, that there is an exception if two paths γ_i, γ_j merge and run on top of each other. In this case, the gap turns into $d_i + d_j$. For further details, see [KNP07].

The algorithm first computes the gaps $d_i \in \mathbb{R}^2$ for all cut paths. Then, a special parameter function (h', k') is computed, which is harmonic and whose gaps at cut paths are exactly $[d_i] - d_i$. The maps h' and k' are uniquely described by this property up to global constant summand. The final parameterization is given as $(u' + h', v' + k')$ and satisfies all needed conditions.

5 Singularities

5.1 Singularities in QuadCover

A characteristic of each parameterization is the location and type of its singularities. The placement of singularities is important for low metric distortion.

Fractional index. For vector fields, the type of a singularity can be

measured by its index. Take a closed path, which runs counterclockwise around a singularity p. The index $ind_p(X)$ of a vector field X at p is defined as the number of whole revolutions of the vectors along the path. The index is always an integer number.

When dealing with frame fields, the index is not constrained to be integer since tracking a vector along a closed path may not necessarily end up in the same frame vector. Therefore, the index can be multiples of 1/4, (Fig. 5).

In QuadCover, we detect singularities from the input frame field (in most cases the principle curvature field). There is a difference about the handling of singularities with integer index and those with fractional index. Integer indices just appear as vector field singularities on the covering. Singularity of fractional index are resolved using a branch point. Remember from Sect. 3.1 that the type t_p of a branch point can be seen as the layer shift when walking around the branch point once. During the construction of the covering, a branch point of type $t_p = 4(i \bmod 1)$ is placed at each point, where the frame field has a non-integer singularity of index i. If you track one vector around the branch point, you end up on a different layer, i.e. the parameter lines exchange or flip the sign.

Figure 5 shows the branch points of different singularities. For index $1/4$ or $-1/4$, the 4 layers of the covering are connected at the branch point (of type 1 or 3) forming a cyclic spiral. For index $1/2$ or $-1/2$, two spirals (with two layers each) are formed (branch point of type 2).

Branch points increase the topologically complexity of the cover. They can be described by cutting an infinitesimally small hole into the surface. This enlarges the fundamental group of the surface and therefore enriches the wealth of parameter functions.

The location of a branch point on the surface is fix during the whole optimization. The reason is that only the parameter function on the covering gets optimized, not the covering surface itself.

Integer index. Singularities of integer index do not have any affect on the topology of the covering. The north pole of a sphere for example (Fig. 1, right) does not lead to a branch point since walking around the pole ends up on the same layer. The same is true for a saddle of index -1.

The nature of singularities with integer index is completely different. Since they are not represented by the covering itself, they just arise as vector field singularities of the gradients. This may occur automatically during the optimization.

Positive integer index. There is a special case for singularities with

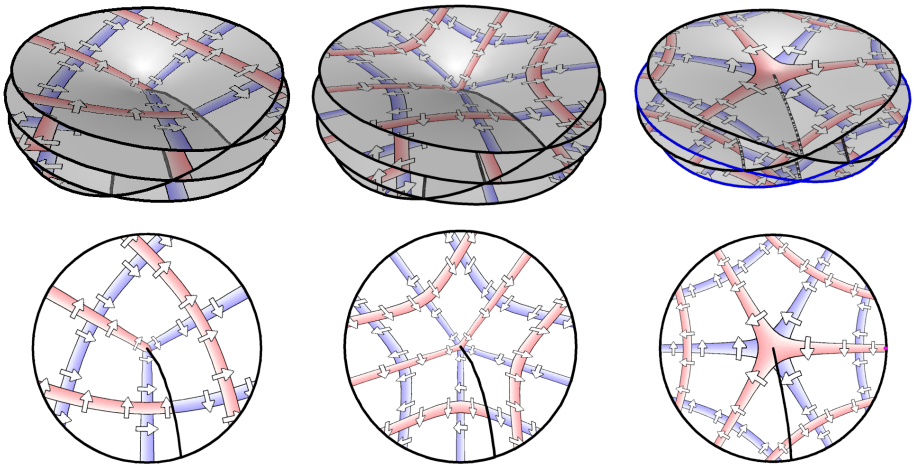

Figure 5: Parameterizations with different singularities and their coverings. Left: Index = 1/4, Middle: Index = −1/4, Right: Index = −1/2.

positive integer index. Take a u function which has a local maximum at point p, thus ∇u has index 1 there. In a good parameterization, the gradients of u and v are approximately perpendicular to each other. In this case, ∇v would be a vortex around p.

Unfortunately, vortices cannot be described with a standard PL parameter map on triangles because of the following reason: If γ is a curve which runs around p, then the path integral $\int_\gamma \nabla v ds$ is 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 not possible to represent such a function as a piecewise linear function on a triangle mesh.

Hence, if the input frame field has a singularity of index 1, the parameterization algorithm ignores it and produces a parameter function which is far from the guiding field (see Fig. 1, left). We now explain a method, which handles this case and therefore provides better parameterizations.

5.2 Handle singularities of positive integral index

Use quads. A regular triangle mesh is too rigid for a piecewise linear vortex-like parameter function. Thus, we locally remesh the surface and invent 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 $k \in \mathbb{N}$. Remesh the vertex star as displayed in Fig. 6. All triangles in the star become degenerated quads, whereas two vertices are geometrically in the same location as p.

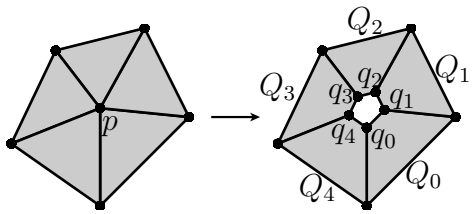

Figure 6: **Left:** Vertex star of a singularity p. **Right:** Combinatoric of the remeshed vertex star. It consists of 4 quads Q_i. The inner vertices q_i are geometrically located in the same point p, thus the quads are degenerated.

Scalar functions on a mesh with quads are not longer forced to be piecewise linear. We extend the space to functions, which are linear on each triangle and bilinear on each quadrangle. Given function values at the vertices of a quad, the function itself is then given by the unique bilinear function, which interpolates these values. The resulting function is continuous, even if triangles and quads are mixed in the same mesh.

Parameter functions on meshes with triangles and quads can be used as texture map. If the rendering system supports bilinear textures, one can easily display singularities of integral index, see Fig. 7.

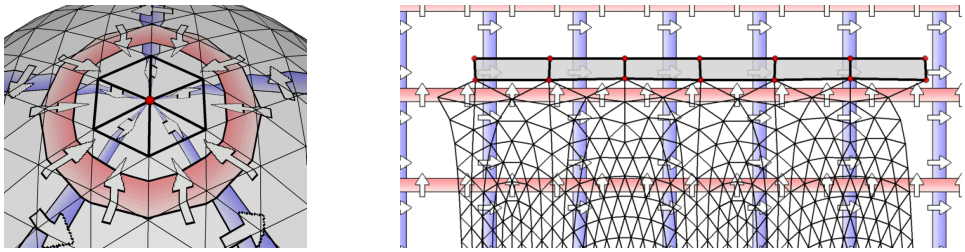

Figure 7: **Left:** Bilinear texture on a coarse mesh with triangles and quads. **Right:** Image of the elements in texture space. The quads in the vertex star of the singularity are marked in grey.

Using quadrilaterals, one can now map a polygon in texture space to a single point on the surface. All the blue parameter lines which goes through one of the quadrangles in Fig. 7 run into the singularity.

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 turned 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 extended QuadCover is listed in Algorithm 1.

Algorithm 1 Modified QuadCover algorithm, Input: Guidance 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:    end if
 7: end for
 8: Run original QuadCover algorithm
 9: for all p in specialSingularities do
10:    Replace all adjacent triangles to p by a quadrilateral
11:    Compute texture coordinates for the quads
12: end for
```

Lines 1–7 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. 8. Then, the QuadCover algorithm is applied to the triangle surface. Fig. 9 shows the texture domain of the example from Fig. 7.

In lines 9–12, the singularities get 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. 6, 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. 8, right). In quadrangle Q_j, compute the texture coordinates as:

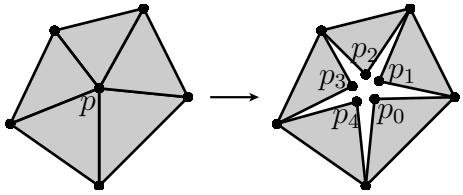

Figure 8: Each vertex star of a singularity p with positive integer index will be cut open. The right mesh shows the new combinatoric, the inner vertices p_i are geometrically located at p.

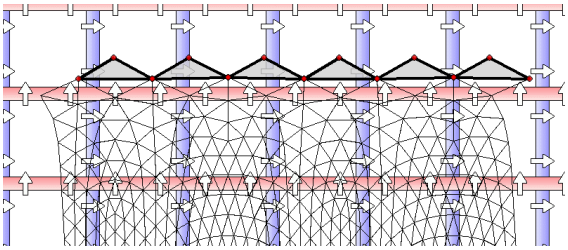

Figure 9: Texture domain after parameterization with modified mesh connectivity. The surface from Fig. 7, left is taken as input.

$$\begin{aligned} f_j(q_j) &:= 1/2\,(f_{j-1}(p_{i-1}) + d_{i-1,i} + f_j(p_j)) \\ f_j(q_{j+1}) &:= 1/2\,(f_j(p_j) + f_{j+1}(p_{i+1}) + d_{i+1,i}) \end{aligned} \tag{3}$$

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

6 Results

With this extension, QuadCover produces very stable parameterizations, even when singularities of integral index are present. All what we need is a frame field, which contains singularities at reasonable locations.

A good placement of singularities is still an open problem. The singularities from principle curvature fields are mostly nice, but their location is not very stable in nearly umbilic areas. Particularly, singularities of index 1 will mostly split up into 4 singularities of index 1/4 each.

We tested the algorithm on some user generated frame fields. Fig. 10, left shows the graph of the function $f(u, v) = \sin(u)\cos(v)$. A frame field was produced using the gradient field of the height function and its 90 degrees

rotated field. The parameterization has two singularities of index 1 and two saddles of index -1.

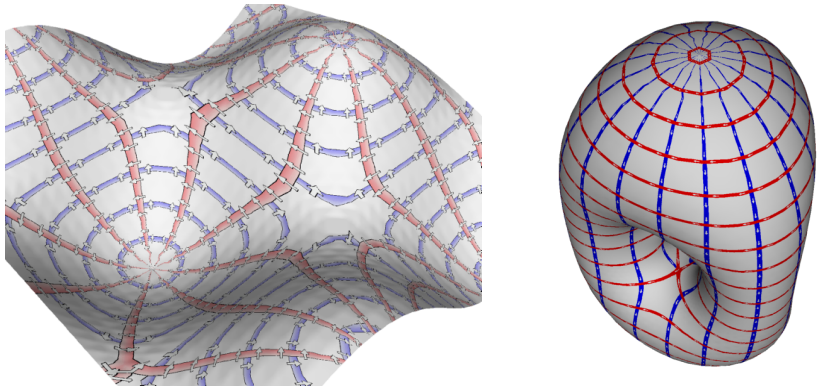

Figure 10: **Left:** The parameterization is guided by the gradient and cogradient of the height function. Singularities of index 1 naturally appear at all local minima and maxima. They are correctly detected and resolved by the method. The singularities of negative index at saddle points occur automatically, even if there is no parameter line running directly into the saddle. **Right:** Parameterization on a 3-fold Lawson surface. The hight function was used for this parameterization, too.

Fig. 10, right shows a Lawson surface. It is a constant mean curvature in $\mathbb{R}^3$ and is made out of 30k triangles. The parameterization took several seconds.

References

[BCGB08] M. Ben-Chen, C. Gotsman, and G. Bunin. Conformal flattening by curvature prescription and metric scaling. *Computer Graphics Forum*, 2008.

[FH05] M. Floater and K. Hormann. Surface parameterization: a tutorial and survey. In *Advances in Multiresolution for Geom. Modelling*. Springer Verlag, 2005.

[FK80] Hershel M. Farkas and Irwin Kra. *Riemann Surfaces*. Springer Verlag, 1980.

[GY03] Xianfeng Gu and Shing-Tung Yau. Global conformal surface parameterization. In *Symp. on Geom. Proc.*, pages 127–137, 2003.

[HAT+00] Steven Haker, Sigurd Angenent, Allen Tannenbaum, Ron Kikinis, Guillermo Sapiro, and Michael Halle. Conformal surface parameterization for texture mapping. *IEEE Trans. on Vis. and Comp. Graph.*, 6(2):181–189, 2000.

[HPS08] K. Hormann, K. Polthier, and A. Sheffer. Mesh parameterization: Theory and practice. In *SIGGRAPH Asia, Course Notes*, 2008.

[Jos02] Jürgen Jost. *Compact Riemann Surfaces*. Springer, 2002.

[KNP07] Felix Kälberer, Matthias Nieser, and Konrad Polthier. Quadcover - surface parameterization using branched coverings. *Comput. Graph. Forum*, 26, 2007.

[LJX+08] Yu-Kun Lai, Miao Jin, Xuexiang Xie, Ying He, Jonathan Palacios, Eugene Zhang, Shi-Min Hu, and Xianfeng David Gu. Metric-driven rosy fields design. Technical report, Tsinghua Univ., Beijing, China, 2008.

[RLL+06] Nicolas Ray, Wan Chiu Li, Bruno Lévy, Alla Sheffer, and Pierre Alliez. Periodic global parameterization. *ACM Trans. Graph.*, 25(4):1460–1485, 2006.

[SSP08] B. Springborn, P. Schröder, and U. Pinkall. Conformal equivalence of triangle meshes. *Siggraph, Proceedings*, 2008.

[TACSD06] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun. Designing quadrangulations with discrete harmonic forms. In *Proc. Eurographics*, 2006.