To appear in ACM TOG 31(4).

Dual Loops Meshing: Quality Quad Layouts on Manifolds

Marcel Campen* RWTH Aachen University
David Bommes* RWTH Aachen University
Leif Kobbelt* RWTH Aachen University

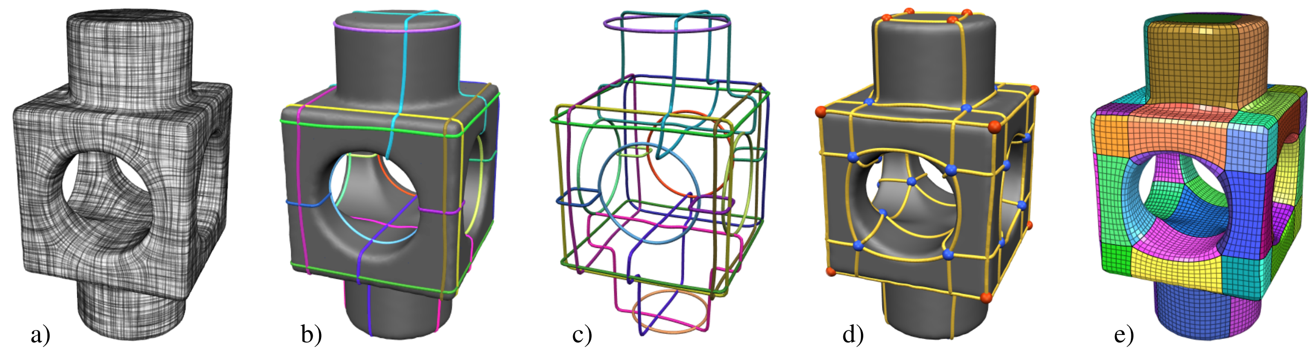


Figure 1: *Overview of our* Dual Loops Meshing *approach that constructs coarse all-quadrilateral patch layouts with high geometric fidelity. Guided by a field of principal curvature directions (a), a collection of transversally crossing loops on the surface is generated in a geometry-aware manner (b, c). These loops partition the surface into polygonal regions whose valences are intimately related to their total curvature. By dualizing the graph formed by these loops, we obtain a coarse all-quadrilateral layout (d) whose irregular nodes are located in geometrically plausible regions. This layout can, for instance, be exploited for the generation of quad meshes with a high-level patch structure (e).*

Abstract

We present a theoretical framework and practical method for the automatic construction of simple, all-quadrilateral patch layouts on manifold surfaces. The resulting layouts are coarse, surface-embedded cell complexes well adapted to the geometric structure, hence they are ideally suited as domains and base complexes for surface parameterization, spline fitting, or subdivision surfaces and can be used to generate quad meshes with a high-level patch structure that are advantageous in many application scenarios. Our approach is based on the careful construction of the layout graph's combinatorial dual. In contrast to the primal this dual perspective provides direct control over the globally interdependent structural constraints inherent to quad layouts. The dual layout is built from curvature-guided, crossing loops on the surface. A novel method to construct these efficiently in a geometry- and structure-aware manner constitutes the core of our approach.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling;

Keywords: quad mesh, base complex, edge flow, simplification, segmentation, parameterization

*e-mail: (campen|bommes|kobbelt)@cs.rwth-aachen.de

1 Introduction

In recent years a wealth of automatic quad-meshing techniques has been presented [Ray et al. 2006; Dong et al. 2006; Kälberer et al. 2007; Huang et al. 2008; Bommes et al. 2009; Zhang et al. 2010]. These enable the construction of high-quality meshes – with controlled element alignment, orientation, sizing, aspect ratio, etc. In conjunction with the inherent ability of orienting elements according to principal curvature directions the (local) regularity and (low-level) structuredness of quad meshes provide benefits over traditional triangle meshes in numerous application areas. So far, however, significantly less attention has been paid to the high-level, global structure of the resulting meshes. On this level, each quad mesh implicitly defines a unique coarsest underlying *patch layout* with quadrilateral patches, also termed *base complex* or *singularity graph* (cf. Figures 2 and 1e). Its patch borders are defined by those sequences of edges which, within the quad grid, form straight connections between irregular vertices [Bommes et al. 2011].

In practice, the quality of this patch layout of a quad mesh is of interest for numerous reasons. It is desired to be coarse and simple (while still appropriately respecting the underlying geometry) in order to enable the application of efficient adaptive and multilevel solver schemes in the context of quad-based Finite Element simulation and to expose the high level of structuredness expected by applications like mesh compression and Fourier or wavelet based processing. In the field of character animation designers are interested in quad meshes with good so-called *edge-flow* (a loose concept closely related to geometry-aware, simple base complexes) as these tend to reduce deformation artifacts and distortions. In this context a simple layout can furthermore directly be exploited as convenient domain for purposes of texture and detail mapping. This leads us to further applications, not directly involving quad meshes, that benefit from simple quad layouts in their own right: for instance, such layouts provide appropriate domains for rectangular surface parameterization, reverse engineering, higher-order patch (e.g. NURBS) fitting, subdivision surfaces, and other scenarios.

Many state-of-the-art quad-meshing techniques contain no aspect aimed at considering the global structure, hence often meshes with

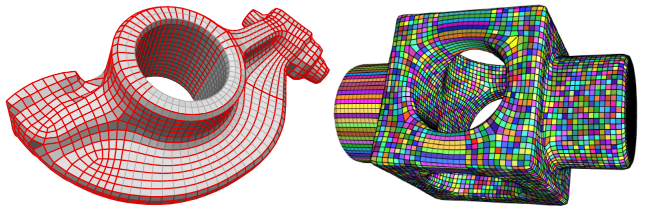


Figure 2: *Typical quad meshes created by a field-guided parameterization approach [Bommes et al. 2009] and their base complexes. On the left the patch borders are highlighted. On the right patches are visualized by individual colors – Figure 1e shows a very similar quad mesh of this object (same resolution, same number and type of irregular vertices) created by our method which has a much simpler base complex.*

very complex patch layouts (cf. Figure 2) are produced, unless sharp surface features or additional structural constraints [Myles et al. 2010] fortuitously enforce a simple one. It may be tempting to try "forcing" these methods into creating coarse layouts by prescribing large quad sizes such that the resulting mesh itself can serve as layout ab initio. However, this is hardly possible since (in contrast to usual quad-meshing assumptions) quads of widely differing sizes and aspect ratios are required to achieve a simple, coarse layout while simultaneously respecting a geometry-aware irregularity constellation (cf. Figure 12). As these patch parameters (size, aspect ratio, orientation) interdepend in a complex global manner, approaches relying on their local a priori determination [Zhang et al. 2010; Kovacs et al. 2011] are not suitable either.

In this paper we analyze the fundamental structural constraints and geometric requirements which constitute the general class of quad layouts and, based on that, present a novel method that directly constructs geometry-aware layouts on manifolds. In this context, structural simplicity and geometric fidelity of the layout tend to be opposing objectives and need to be balanced appropriately. In essence, our method is based on the careful construction of the combinatorial dual of the quad layout in a way that is sensitive to the input's geometry and ensures structural consistency. Figure 1 provides a quick overview over the stages of the procedure.

Our key contributions are:

• **A novel framework** to describe, identify, and manipulate structurally valid and geometrically faithful duals of quad layouts.

• **Efficient methods** for the geometry-aware construction of embedded loops that are the key components of the dual layouts.

• **A practical greedy algorithm** based on the framework to automatically and efficiently perform the layout construction.

2 Related Work

Quad Meshing State-of-the-art quadrangular remeshing methods, which are based on, e.g., periodic parameterizations [Ray et al. 2006], Morse-Smale theory [Dong et al. 2006; Huang et al. 2008], or mixed-integer optimization [Kälberer et al. 2007; Bommes et al. 2009], are usually targeted at creating meshes with uniformly sized elements. Extensions and modifications have been presented [Zhang et al. 2010; Kovacs et al. 2011] which allow for the incorporation of sizing and aspect ratio fields to adjust the element properties to local properties of the input geometry. Control over the base complex and its simplicity is not available or limited in these methods; those based on spectral surface analysis [Dong et al. 2006] do create meshes with a two-level hierarchy, but the uniformity of the coarser level largely restricts its overall simplicity.

Coarse Quad Layouts Eck and Hoppe [1996] proposed a base complex construction for the purpose of spline patch fitting based on pairing of the faces of a triangular base complex. A similar approach is chosen by Boier-Martin et al. [2004]. Control over the geometric fidelity is quite limited due to the nature of these approaches. Daniels et al. [2009] propose to convert a triangle mesh to a quad mesh by one step of Catmull-Clark subdivision and then simplify the result using quad mesh decimation techniques. Driving this decimation in a way that the resulting coarse mesh is geometrically faithful and irregular vertices end up in plausible locations similarly remains a problem. In this context Panozzo et al. [2011] use special kinds of pre-computed sizing fields (called *fitmaps*) to control a decimation process.

Due to the apparent complexity of the problem, several researchers proposed a manual [Bommes et al. 2008; Krishnamurthy and Levoy 1996] or semi-automatic, assisted [Tong et al. 2006; Tierny et al. 2011; Ji et al. 2010] creation of quad layouts. This approach to the problem can be advantageous as layout design decisions might depend on the intended use of the layout and cannot always be derived from geometry alone. However, manually designing a good all-quadrilateral layout turns out to be an intricate and cumbersome task even for experienced users due to the involved complex global interdependencies and structural constraints that are to be considered, underlining the utility and benefit of an automatic approach.

Automatic methods have been presented for related, simpler variants of the problem, like quad-dominant layouts [Marinov and Kobbelt 2004], polygonal layouts [Cohen-Steiner et al. 2004], or partitions with T-junctions [Eppstein et al. 2008; Myles et al. 2010].

Base Complex Simplification Recently, the problem has been approached "reversely": instead of creating a layout from scratch, one starts from a fine quad mesh base complex and attempts to successively simplify it by iteratively adjusting its structure. Bommes et al. [2011] and Tarini et al. [2011] achieved respectable results with this strategy. In Section 7 we discuss the respective advantages and disadvantages in comparison to our technique.

Distantly related are general quad mesh decimation methods, e.g. [Daniels et al. 2008], that are, however, not specifically targeted at considering the base complex.

3 Problem Definition

We first introduce some basic definitions and terminology and then state our objective in more detail. A *polygonal patch layout* Q is a 2-dimensional cell complex embedded on an orientable 2-manifold $\mathcal{M}$. Its 0-cells are called *nodes*, its 1-cells *arcs*, and its 2-cells *patches*. The term *valence* signifies the number of arcs incident to a node or patch. If all patches have valence 4, the layout is a *quad layout*. In this context, nodes of valence 4 are called *regular*, all others *irregular*. At regular nodes, two pairs of *opposite* arcs can be defined in the intuitive way. A maximal chain of successively opposite arcs is called *separatrix* if it starts and ends at (not necessarily distinct) irregular nodes. Figure 3 illustrates these definitions.

The *quality* of a quad layout is a measure that, to some extent, depends on its purpose. Still, we can clearly identify a set of properties that well describe the general suitability of a layout from the application point of view:

- **Geometric fidelity:** patches should map to planar rectangles with low parametric distortion
- **Structural simplicity:** the number of patches should be small

The first point implies that it is advantageous if patches are close to developable, their corner angles close to right, and their arcs

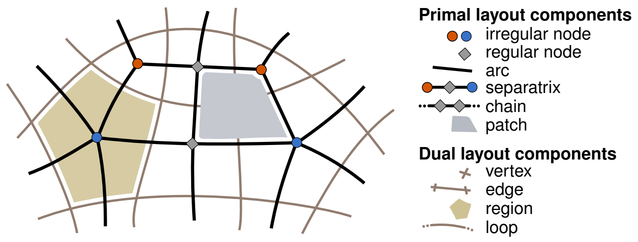


Figure 3: *Illustration of a layout's primal and dual components.*

close to geodesic. These are important criteria as strongly deviating patches tend to lead to distortions or degradations in one form or the other in most applications. The second point expresses the general objective of *domain simplicity*. In many cases these two objectives are opposing – our objective is to find layouts with a good balance in this regard. We can already conclude that aligning patch arcs with principal curvature directions and positioning nodes with suitable valences in regions of certain total curvatures helps in achieving high quality layouts. Numerous of the mentioned quad remeshing and layout construction methods are based on this insight.

3.1 Motivation

Before we go into detail on our method, we provide some insight into our motivation for ultimately working in the dual setting.

Probably the most obvious attempt to construct a quad layout given a set of nodes with prescribed valences consists in connecting these by incrementally constructing the separatrices (i.e. arcs or arc chains) from node to node. It turns out, however, that the structural consistency requirements ("only valence-4 patches") are hard to control when proceeding in this local, incremental manner: whenever a cycle in the graph of nodes and separatrices is closed during the process, strict conditions relating the valence of its patches to the number and valence of inlying nodes have to be met – otherwise new irregular nodes would have to be introduced in order to still be able to obtain a quad-only layout. Besides the fact that these additional nodes might be geometrically implausible (causing distortion), they further imply additional separatrices that are to be inserted. Hence, it can become difficult to generate a complete layout without introducing an excessive number of irregular nodes. This primal, separatrix-based layout construction approach is furthermore highly order-dependent: early greedy decisions can render a layout unacceptable long before this is realized.

3.2 Our Approach

The key observation that allows us to avoid the problems of separatrix-based approaches is the fact that the layout consistency is much easier ensured when not working in the primal but in the dual setting. In more detail, this means we do not directly construct the primal layout graph of separatrices connecting irregular nodes but its combinatorial dual, which we eventually dualize to obtain the primal. We construct this dual graph from the arrangement of a set of crossing loops (closed curves) on the surface (cf. Figure 1 b,c for an example). Murdoch et al. [1997] already emphasized that the dual view of a quad mesh as intertwined loops is advantageous because it directly exhibits the global connectivity constraints involved. The central question now is how such a dual layout can be generated from the infinitely dimensional space of loops. In Section 4 we explain, in a *continuous setting*, the theory behind our construction of arrangements of loops on manifolds that allows us to achieve layouts with geometric fidelity as well as structural consistency and simplicity. Afterwards, in Section 5, we describe the implementation in a *discretized setting* on triangulated manifolds.

4 Dual Loops

The combinatorial dual D of a quad layout Q is a 4-regular cell complex, i.e. every dual *vertex* has four incident dual *edges*. Hence, at every vertex there are two pairs of *opposite* edges, and the set of all edges uniquely decomposes into a disjoint collection of cycles of successively opposite edges. Note that these dual edge cycles correspond to (possibly non-simple, i.e. self-crossing) cyclic quad strips in Q. It is this dual view of a quad layout as a collection of dual cycles that builds the fundament of our approach: we construct an arrangement of loops on the surface which then constitute the dual cycles forming a dual graph D. The loop intersections define this graph's vertices and the loop segments between any two intersections define its edges. The *regions* of the induced partition of $\mathcal{M}$ consequently define its faces (cf. Figure 3).

Principally, the concept of dualization has no inherent geometric component. But note that, while dualization of graphs is a purely combinatorial concept, dualization of surface embedded cell complexes additionally has a topological component, involving, e.g., the homotopy classes of the embedded edges, which affect the topology of the dual. For our setting this means that the *detailed* geometry of the dual loops we construct does not play a major role for the final primal layout Q that is obtained by dualization of D, but we still have to construct these dual loops in a geometry-aware manner as their embedding directly implies the topology of D. Hence, we can furthermore very naturally exploit the geometry of the dual created in this manner to derive an initial embedding for the primal. This implies a loose geometric relation between primal and dual, e.g. a primal arc is often roughly orthogonal to its dual edge, a dual edge strip runs through a primal quad strip roughly parallel to its boundary arcs, etc., which we can exploit in the following.

4.1 Loop Arrangements

Let us consider which properties an arrangement of loops needs to fulfill in order to induce a valid dual graph in the described way:

(1) loop intersections are transversal (i.e. non-tangential),
(2) no three loops intersect in one point,
(3) each region is bounded by at least 2 loop segments,
(4) each region has disc topology.

The first two properties guarantee that each intersection induces a unique valence 4 vertex, the third ensures that no dual face has a valence < 2, which would result in primal layout nodes with valence 1 that are most often considered unacceptable; if necessary, we can also raise this lower bound to disallow valence 2 nodes. The fourth point ensures a valid embedding with topologically simple faces.

If these properties are fulfilled, at least a valid dual graph is induced. Additionally we are interested in quality. Considering the abovementioned geometric relation between dual and primal elements the quality criteria listed in Section 3 can roughly be translated from the primal to the dual setting as follows:

(5) loop intersections should be close to orthogonal,
(6) loops should follow principal curvature directions,
(7) each region's valence should reflect its total curvature,
(8) loops should be short and few in number.

Item (5) encourages rectangular patches, (6) and (7) encourage principal curvature alignment and send irregular primal nodes to surface regions of suitable curvature, allowing for low patch-rectangle mapping distortions. Item (8) directly translates into structural simplicity as the total length of primal arcs and dual edges is closely related in practice. Note that these items are soft requirements promoting quality while (1)-(4) are mandatory conditions.

The loop construction process has to be driven very carefully to yield arrangements which fulfill (1)-(4) and support (5)-(8). As an example of the delicacy, consider that once any irregular region has been created by a set of loops, this irregularity can never be eliminated again by the addition of further loops; they can only split the region, but irregular sub-regions inevitably remain.

4.2 Principal Curvature Fields

Constructing loops in a way that they are aligned with directions of minimal and maximal curvature directly serves fulfilling properties (1), (5), and (6) (and indirectly also (7)). These directions can conveniently be represented as a *cross field* (i.e. 4-symmetry direction field) [Ray et al. 2008; Ray et al. 2009; Palacios and Zhang 2007] on $\mathcal{M}$. In order to base our construction on globally consistent directions that are also well-defined in umbilic regions, in practice we make use of the interpolation approach proposed by Bommes et al. [2009] to create consistent smooth cross fields. Figure 1a shows a visualization of such a field. We will see that this representation also serves in fulfilling the other requirements as the singular points of the cross field additionally encode integral curvature information in their indices, hence indicate favorable positions and valences for irregular nodes. In detail, due to the curvature-driven construction (and due to the Poincaré-Hopf theorem for cross fields [Ray et al. 2008]), the singularities can intuitively be understood as local representatives of an imagined clustering of the surface's total curvature: a singularity with index i represents $2\pi i$ of total curvature. In essence, it is this relation that will ultimately serve to fulfill (7). A singularity index i is an integer multiple of $\frac{1}{4}$ intuitively signifying the "angular defect" (as fraction of 2π) of the field when turning around the singularity once. In the quad mesh context a corresponding node's valence k is thus naturally related to it by $i = 1 - \frac{1}{4}k$ [Kälberer et al. 2007; Bommes et al. 2009]. Hence, indices $\frac{1}{4}$ and $-\frac{1}{4}$ are by far the most common in our context.

4.3 Branched Coverings

Locally and away from singularities, a smooth cross field F_C on $\mathcal{M}$ can be decomposed into four *direction fields*. This does not hold globally. However, as shown by Kälberer et al. [2007] for the case of frame fields, we can construct a *four-sheeted branched covering surface* $\mathcal{M}_4$ of $\mathcal{M}$ with branch points induced by the singularities, and canonically lift the cross field F_C on $\mathcal{M}$ to a single smooth direction field F_D on $\mathcal{M}_4$. Analogously, we can construct a *two-sheeted branched covering surface* $\mathcal{M}_2$ of $\mathcal{M}$ and lift the cross field to a single smooth *line field* F_L on it. Figure 4 illustrates this schematically. Note that $\mathcal{M}_4$ is also a two-sheeted branched covering surface of $\mathcal{M}_2$. In the following we will deal with curves and loops on $\mathcal{M}_4$; note that they project uniquely onto $\mathcal{M}_2$ and $\mathcal{M}$ by means of the respective covering maps – for brevity we will not always mention this projection explicitly but simply refer to them and their projection images by the same name. These covering constructs significantly simplify our further elaborations by allowing us to work with cross, line, or direction fields as needed.

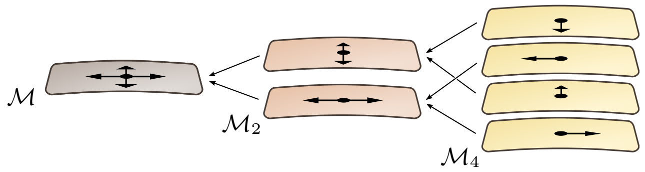
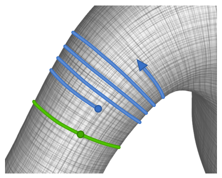


Figure 4: *Illustration of a small region of $\mathcal{M}$ with its branched coverings $\mathcal{M}_2$ and $\mathcal{M}_4$ and the cross, line, and direction field.*

4.4 Loops on Branched Coverings

Now consider what happens when we attempt to create a principal direction aligned loop through a point p on $\mathcal{M}$ by tracing. We choose one of the four directions from the local cross field by lifting p onto one of the four sheets of $\mathcal{M}_4$, trace a curve through the direction field F_D on $\mathcal{M}_4$ until we get back to the starting point, and project the curve from $\mathcal{M}_4$ to $\mathcal{M}$. The obvious problems are that the curve might never return, and if it does, it will probably be very long and far too complex to provide a useful loop for our purpose – we have to trade some deviation from the field for shortness. This is illustrated here on a cylindrical structure: the blue curve exactly follows the field and spirals over the surface, whereas the green curve slightly deviates from the field so as to be able to form a short loop.

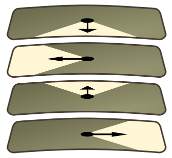


Let $l : [0, b] \to \mathcal{M}_4$ denote a regular curve in arc-length parameterization which does not run exactly through a branch point (where the direction field is not defined). We define a combined measure c_α that rates such curves based on their direction fidelity and shortness, balanced by the parameter $\alpha \in [1, \infty)$:

$$c_\alpha(l) = \int_0^b \sqrt{\cos^2\theta(s) + \alpha^2 \sin^2\theta(s)}\, ds$$

where $\theta(s)$ is the angle between the curves' tangent and the desired direction $F_D(l(s))$ at $l(s)$. A similar measure has also been employed by Tarini et al. [2011] in the primal setting to rate separatrices. We are interested in non-trivial loops (closed curves) which minimize this measure for a prescribed α (called *minimal loops*) subject to certain constraints that we introduce in the following. As extreme cases, $\alpha = \infty$ leads to the abovementioned complex minimal loops, whereas $\alpha = 1$ leads to field oblivious geodesics. Values around 30 proved to provide a good balance for our purpose.

In order to obtain loops which are not only short and field-aligned, but furthermore can be combined to structurally valid loop arrangements we impose the following constraint: we restrict the set of non-trivial regular loops on $\mathcal{M}_4$ to the set L of *admissible* loops that fulfill $\theta(s) \in [-\frac{\pi}{4}, \frac{\pi}{4})$, i.e. we limit tangent deviations from the principal curvature directions. This effectively partitions the tangent space at each point into four "sectors", each describing the admissible loop tangents on one of the four sheets of $\mathcal{M}_4$. For $\alpha \gg 1$ minimal loops are very unlikely to violate this anyway, but imposing it as hard constraint allows us to guarantee important properties in the following. Furthermore, we define an arrangement of loops on $\mathcal{M}_4$ that have no intersections on $\mathcal{M}_2$ to be $\mathcal{M}_2$*-simple*. Note that $\mathcal{M}_2$-simple arrangements of loops meeting $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4})$ contain no tangential intersections, thus fulfill (1). This is illustrated in Figure 5 and elaborated in the proof of Theorem 4.1.



The following two theorems state the further beneficial properties of the $\theta(s) \in [-\frac{\pi}{4}, \frac{\pi}{4})$ constraint in conjunction with $\mathcal{M}_2$-simplicity. Proofs can be found in the appendix. Let $I(R)$ denote the sum of singularity indices lying in a region R:

Theorem 4.1 *If a region $R \subset \mathcal{M}$ (bounded by loop segments) with disc topology in an $\mathcal{M}_2$-simple arrangement of admissible loops has k loop intersections on its boundary, then $I(R) = 1 - \frac{1}{4}k$.*

This relation between singularity indices and region valences implies an indirect relation between a region's total curvature and its valence since the cross field singularities loosely represent local curvature integrals by their indices. This relation "region valence $k \leftrightarrow$ index $1 - \frac{1}{4}k$" is exactly the desired one mentioned in Section

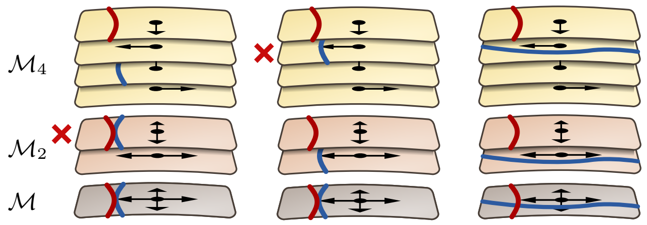


Figure 5: *Left: two loop segments (red, blue) have tangential contact on $\mathcal{M}$; this is possible as $\mathcal{M}_2$-simplicity is violated. Middle: the loops are $\mathcal{M}_2$-simple, but the $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4})$ constraint is violated allowing for a tangential intersection. Right: $\mathcal{M}_2$-simplicity and the θ-constraint together only allow for transversal intersections.*

4.2. Ultimately, this is beneficial for the quality of our final primal patches as the valences of irregular primal nodes will "fit" the geometry. It further shows that no unnecessary irregular nodes will arise as regions with index sum 0 are always regular (i.e. bounded by four loop segments).

Now, the idea underlying the dual layout construction is to choose loops from the set L of admissible loops such that singularities get separated from each other, aiming at having each singularity lie in a separate irregular region – all regions free from singularities shall be regular. We can achieve this separation in the following sense:

Theorem 4.2 *In an $\mathcal{M}_2$-simple arrangement of loops any region R with $I(R) > \frac{1}{4}$ containing multiple singularities of index $\frac{1}{4}$ or $\frac{1}{2}$, and any region R with $I(R) < -\frac{1}{4}$ containing multiple singularities of index $-\frac{1}{4}$ can be split into regions with smaller $|I(R)|$ by adding further admissible loops without introducing irregular regions outside R and without violating $\mathcal{M}_2$-simplicity.*

This means we can guarantee that singularities can suitably be separated by admissible loops for most practically relevant cases. This is a beneficial result as it shows that we are usually not bound to accept irregular nodes of very high or low valence in the primal layout that are not explicitly required by corresponding surface features. The theorem further extends to several cases which include multiple higher-order singularities (which generally are quite rare in practice). Note that even an exceptional case where singularities cannot be separated completely does not cause any structural problems: any region, even if it contains multiple singularities, simply turns into a single primal node during primalization (cf. Section 4.6).

Let us briefly summarize which insights we gained regarding fulfillment of the eight required/desired properties: (1) and (2) are ensured by $\mathcal{M}_2$-simplicity in conjunction with the $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4})$ constraint, (5) and (6) are respected by minimizing c_α for $\alpha \gg 1$, (3) can be achieved due to Theorem 4.2 if there are no singularities of index $\geq \frac{3}{4}$ (which normally is the case in practice, and can even be ensured by dividing such singularities), (7) is fulfilled according to Theorem 4.1. The shortness part of (8) is governed by using c_α-minimal loops. For (4) "each region has disc topology" there is no theoretical guarantee that this can always be achieved on objects of non-zero genus; in fact, one can artificially create pathological cross fields such that loops with arbitrarily large $|\theta|$ are necessary to cut the surface to discs (cf. the cross field depicted in Figure 4 left in [Kälberer et al. 2007] for a basic example). These are, however, "unnecessarily unsmooth" and, in practice, do not arise from the employed principal directions guided field construction according to all our experiments. Rigorous guarantees could potentially be achieved by including suitable holonomy constraints [Lai et al. 2010] in the construction process in a future work.

In conclusion, we know that it should be *possible* to construct desired dual layouts from admissible loops, and the remaining question is *how* this can effectively be done, i.e. how to choose *few* admissible loops from the set L that fulfill $\mathcal{M}_2$-simplicity and suitably separate singularities. For this we propose a greedy strategy as described in the following section.

4.5 Greedy Loop Selection Strategy

The primary objective for loop selection is the separation of cross field singularities. In favor of layout simplicity this should be achieved with few loops. We construct the arrangement of loops in an incremental manner by iteratively adding further loops that do contribute to the goal of singularity separation. In this process, to ensure $\mathcal{M}_2$-simplicity, a new loop may not intersect the previously constructed ones on $\mathcal{M}_2$. Let $L(A)$ denote the subset of loops from L which, on $\mathcal{M}_2$, do not intersect the loops of an arrangement A.

For the purpose of deciding whether a loop contributes to singularity separation, imagine the set of all paths on $\mathcal{M}$ connecting a pair (a, b) of singularities without crossing any loop of an arrangement A (cf. Figure 6). As separation is a topological concept, we are not interested in these paths' geometry but solely in their homotopy classes. Let $H(A, a, b)$ denote the set of these classes, and $H(A) = \cup_{a \neq b \in S} H(A, a, b)$, where S is the set of all singularities. A loop $l \in L(A)$ is said to be (A, h)*-cutting* if $h \in H(A)$ but $h \notin H(A \cup \{l\})$. Intuitively, l actively increases the "level of separation" by cutting all connections of class h when adding it to A. Figure 6 illustrates this abstractly on a simple torus; Figure 9 shows a less abstract yet simple example (in the discretized setting).

Note that $H(A)$ can be infinite (and a single loop can be (A, h)-cutting for an infinite number of classes h). In Section 5.2, which deals with an implementation of the following algorithm, we describe how this is handled in practice.

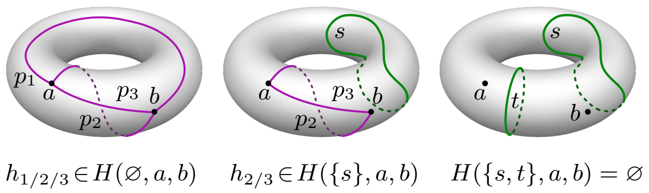

p_1
a
p_3
b
p_2
$h_{1/2/3} \in H(\varnothing, a, b)$
s
a
p_3
b
p_2
$h_{2/3} \in H(\{s\}, a, b)$
s
a
t
b
$H(\{s, t\}, a, b) = \varnothing$

Figure 6: *Left: three paths $p_{1/2/3}$ of three exemplary homotopy classes $h_i = [p_i]$ connecting singularities a and b are depicted. The loop arrangement A is still empty. We have $h_{1/2/3} \in H(\varnothing, a, b)$. Middle: All paths of h_1 cross the loop s, thus $h_1 \notin H(\{s\}, a, b)$. Hence, s is $(\varnothing, h_1)$-cutting. Right: All paths of h_2 and all paths of h_3 cross loop t, thus $h_{2/3} \notin H(\{s, t\}, a, b)$. Hence t is $(\{s\}, h_2)$- and $(\{s\}, h_3)$-cutting. Furthermore, now $H(\{s, t\}, a, b)$ is empty, no more paths between a and b not crossing $A = \{s, t\}$ exist.*

Surely, we are not interested in arbitrary cutting loops but minimal ones: let $l_{\min}(A, h)$ denote the c_α-minimal (A, h)-cutting loop. Then, for some arrangement A, $L_{\min}(A) = \cup_{h \in H(A)} l_{\min}(A, h)$ is the set of all minimal loops whose addition to A would contribute to further singularity separation.

Algorithm: We iteratively add such minimal cutting loops to increase the level of separation based on a greedy approach:

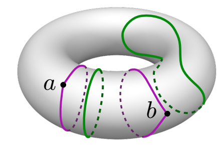
$$\begin{aligned} A^0 &= \varnothing \\ A^{i+1} &= A^i \cup \{ \operatorname*{arg\,max}_{l \in L_{\min}(A^i)} c_\alpha(l) \} \end{aligned}$$

The iteration is repeated until $H(A^i)$ is empty or $l_{\min}(A^i, h)$ is undefined for each $h \in H(A^i)$ because there are no further cutting loops (in cases where not all singularities are separable).

The greedy choice of the maximum instead of a minimum might seem unnatural, but it typically leads to better results. First notice that even the maximum (with respect to c_α) out of the loops of $L_{\min}(A^i)$ is still a minimal cutting loop – for each path homotopy class the set $L_{\min}(A^i)$ contains only the c_α-minimal of the loops that cut it. Hence the choice of the maximum does not lead to unnecessarily long and field-misaligned loops as it might appear at first sight – it still selects from the best loops that are available to achieve separation of certain singularities. See Figure 7 to get an idea of how such loops look like.

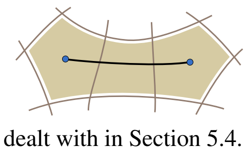
Consider the class $h_{\max} = \arg\max_{h \in H(A^0)} c_\alpha(l_{\min}(A^0, h))$, i.e. the class that, among all homotopy classes, requires the c_α-largest loop to get cut. This loop is $l_{\min}(A^0, h_{\max})$ and $h_{\max}$ cannot be cut by any c_α-smaller loop; in fact, if we first add some loops cutting other classes, then $l_{\min}(A^j, h_{\max})$ for $j > 0$, i.e. the best loop available to cut $h_{\max}$ after j iterations, can easily be c_α-larger due to $L(A^{i+1}) \subset L(A^i)$ by definition. Following the maximum-first strategy, the long loops necessary for separation of some singularities are secured early, in contrast to a minimum-first strategy where other loops already restrict the space of $\mathcal{M}_2$-simple loops available to cut $h_{\max}$ in the end. Hence, as confirmed in our experiments, this maximum-first strategy typically leads to arrangements with a smaller total sum as well as lower variance of the loops' c_α-values.

The objective of achieving regions with disc topology seamlessly integrates into this algorithm: for each singularity a the set of homotopy classes of non-contractible paths connecting a with itself, i.e. $H(A, a, a)$ without the trivial class of contractible loops, is simply included in the set $H(A)$. $H(A)$ will then not become empty before in A all singularities lie in disc topology regions. This is illustrated here based on the example from Figure 6 right: two still remaining non-contractible paths self-connecting a and b are depicted.



4.6 Layout Primalization

The constructed dual layout implies the topological structure of the primal layout. For each irregular region we create a primal node and, in order to provide an initial embedding, position it onto the field singularity lying in that region (if there are multiple, one is chosen arbitrarily). The primal arcs are then constructed from geodesics constrained to the corresponding dual corridors (depicted alongside); regular nodes are created at their intersections. Optimization of the preliminary embedding is dealt with in Section 5.4.



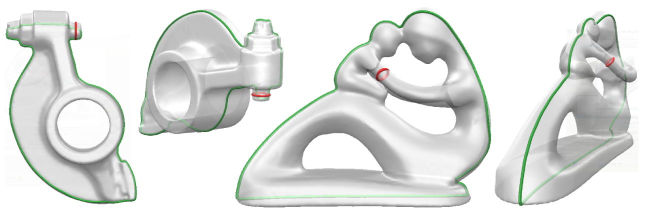


Figure 7: *Visualization of the maximum (green) and minimum (red) of all minimal cutting loops $L_{\min}(A^0)$ in the* ROCKERARM *and* FERTILITY *examples from Section 7. Notice that the longer loops are typically concerned with the global structure, whereas the short ones capture rather local features. The maximum-first greedy approach thus, in some sense, pursues a coarse-to-fine strategy.*

5 Discretization & Implementation

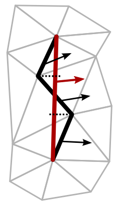
We now present the techniques necessary for a practical implementation of the proposed method on triangulated manifolds $\mathcal{M}$. In this context the terms *vertex*, *edge* and *face* will refer to the triangle mesh $\mathcal{M}$ rather than to dual layout components. The field F_C, with its lifted variants F_L and F_D, is computed and represented discretely per face as described by Bommes et al. [2009]. To simplify parts of the implementation by avoiding special cases, we assume no two singularities are directly adjacent, i.e. incident to a common mesh edge – this can easily be ensured by either merging such pairs or splitting the edge. The branched covering triangle meshes $\mathcal{M}_2$ and $\mathcal{M}_4$ are constructed as described by Kälberer et al. [2007].

5.1 Anisotropic Front Propagation

Loops minimizing c_α are anisotropic geodesics and, given a start point p, can efficiently be computed by anisotropic front propagation on $\mathcal{M}_4$. In this setting α plays the role of a global anisotropy coefficient that, together with the direction field, forms the tensors defining an anisotropic local surface metric guiding the propagation. This process can be understood as a "fuzzy" curve tracing: the fuzzy curve travels fast in the principal direction and slower the more it deviates (cf. Figure 8). When the front reaches p again, the minimal loop is found.

Sethian and Vladimirsky [2003] presented a method for anisotropic front propagation with nice convergence properties, but on coarse meshes the relative error is rather high for large anisotropy coefficients. As a more practical solution for our purpose we compute directed shortest loops based on a weighted graph distance in a special graph, constructed to better approximate the continuous solution. In detail, we create a directed graph P based on $\mathcal{M}_4$: it contains the same set of vertices and two vertices are connected by two directed edges if their discrete graph distance is $\leq k$ in $\mathcal{M}_4$. In this way the angular resolution of the propagation is increased. We call an edge that connects two vertices of distance i an *i-ring edge*. We found $k = 4$ (cf. Figure 8 left for an illustration) to be sufficient, working very well for our purposes (cf. Figure 8 right). For each edge from vertex v to w in P we define its *support* to be the set of $\mathcal{M}_4$-edges forming a shortest path from v to w.

To determine appropriate edge weights modeling the desired metric we first interpolate the field F_D, defined on faces, to the edges of P. For each vertex we map its k-disk to the plane using the discrete exponential map [Schmidt et al. 2006] to obtain a common domain suitable for interpolation of directions. The interpolation to 1-ring edges can then simply be made by averaging between the two incident faces. A plausible extension to an i-ring edge e (red), $i \in \{2, \ldots, k\}$, is computed in an intuitive manner as a weighted average of the directions assigned to its support edges (black), with weights proportional to the length of these edges' projections onto e.



For each edge e of P we then compute its metric-induced weight $w(e)$ by considering it as a curve and evaluating c_α for it. We obtain

$$w(e) := c_\alpha(e) = \sqrt{(\vec{e}^{\mathsf{T}} F_D(e))^2 + \alpha^2 \left(\vec{e}^{\mathsf{T}}\vec{e} - (\vec{e}^{\mathsf{T}} F_D(e))^2\right)}$$

where $\vec{e}$ is the directed edge vector of e in the plane and $F_D(e)$ is a unit vector representation of the field direction. A standard Dijkstra front propagation can then be performed in P, taking the edge weights into account. A resulting loop lives on P; we transfer it to $\mathcal{M}_4$ by replacing each edge by its support, obtaining a loop embedded in the 1-skeleton of $\mathcal{M}_4$. The support edges are furthermore used to perform the loop crossing tests (detailed in the following) on $\mathcal{M}_4$ resp. $\mathcal{M}_2$ while propagating on P.

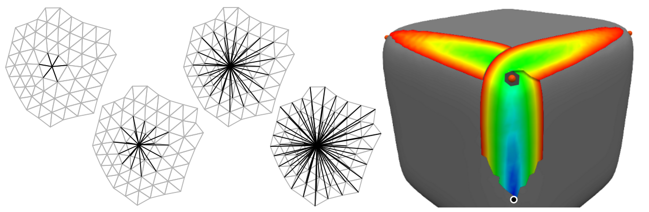
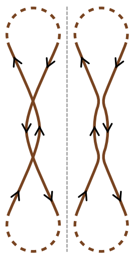


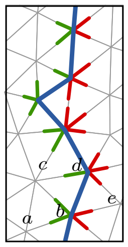
Figure 8: *Left: 1-, 2-, 3-, and 4-disk edges of a vertex. Right: Distance visualization of anisotropic propagation on a rounded cube mesh. Since we propagate on $\mathcal{M}_4$ the front can cross over itself at the branchings and pursue different courses on the different covering sheets, according to the respective field directions.*

5.1.1 Propagation Constraints

The $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4})$ constraint is easily imposed by discarding those edges from the propagation graph P which violate it. $\mathcal{M}_2$-simplicity of the layout is achieved by forbidding the front to cross existing loops on $\mathcal{M}_2$. Additionally, $\mathcal{M}_2$-simplicity of the new loop itself is to be ensured. In the continuous case this is guaranteed by its minimality, but in the discrete setting constructed loops could have small local self-intersections on $\mathcal{M}_2$. These can easily be resolved as depicted: at the intersections, the loop is cut, the parts alternately reversed, and reconnected to a non-intersecting (just grazing) one.



Preventing the front from crossing already existing loops could easily be accomplished by removing the loops' underlying vertices (and incident edges) from the propagation graph. This, however, directly limits the local maximum density of constructed loops to the resolution of the triangulation. We avoid potential dead-ends in the propagation by allowing multiple loops to run over the same graph edges locally – they just may not cross in the sense of $\mathcal{M}_2$-simplicity: during front propagation each graph vertex at the front memorizes for each already existing loop whether the path to this vertex currently runs along the loop on $\mathcal{M}_2$ and whether it moved onto it from the left or from the right. Propagation may then only proceed further along the loop or back to the side it came from, effectively preventing crossing. This can efficiently be implemented by equipping each constructed loop with so-called *whiskers*: two sets of half-edges, those incident to the loops' vertices from the one and the other side, respectively, as illustrated on the right in green and red. The "memory" per front vertex then consists of two flags per existing loop, that are set or reset whenever propagation proceeds over a green or red whisker, respectively, depending on the half-edge orientation. As an example consider the case of the front propagating from vertex a to b. The green flag is set at b due to moving over a green whisker. This disallows further propagation over a red whisker to e; propagation to d carries the green flag over to d, and when propagating to c the flag is not carried on due to moving over a green whisker reversely.



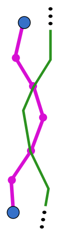
5.2 Singularity Separation

Being able to construct the loops of L in an $\mathcal{M}_2$-simple manner, we now implement the greedy approach presented in Section 4.5. Recall that we need to consider an infinite number of connecting paths in a potentially infinite number of homotopy classes. We employ two efficient heuristics to make this tractable, and add a simple post-validation procedure to be able to guarantee correctness even in case they fail. In practice, they perform so well that post-validation is required to come into action only very rarely. Even then, the worst case is that the result is not greedy-optimal in the described sense, but still structurally correct.

In each step of the greedy construction we need to determine loops which cut certain homotopy classes. To be able to do this efficiently, in the beginning we construct so-called *separation indicators* (SIs) for each pair (a, b) of singularities – paths representing a homotopy class each (discretely embedded in the mesh graph). Based on the intersection constellation of a loop and an SI we are able to estimate whether the loop cuts the whole homotopy class (heuristic I). Figure 9 shows an example of these SIs. While on genus 0 objects (with trivial homotopy group) we have a single SI for each pair, on objects of genus $g > 0$ the homotopy group is infinite; for practicability we restrict ourselves to $2g + 1$ SIs constructed as described by Erickson and Whittlesey [2005] for the case of loop homotopy bases (cf. Figure 6 left for the simplest example of such $2g + 1$ SIs between two singularities on a genus 1 object).

The rationale behind this is the observation that SIs of further homotopy classes (winding around surface handles multiple times, etc.) are very likely to be cut by loops that cut their simpler siblings, anyway (heuristic II). The mentioned method [Erickson and Whittlesey 2005] is adapted from the loop case to our case of paths by not using the distance field of one loop root point but two distance fields, one of each path end point, i.e. the two singularities to be connected, in the maximum spanning tree construction (cf. [Erickson and Whittlesey 2005] for details on this). This yields $2g$ SIs. If $a \neq b$, an additional one, not crossing the cutgraph involved in the construction at all, is created simply by Dijkstra's shortest path algorithm (a step which is not necessary in the case $a = b$ dealt with by the original method due to the triviality of this one class).

Given an SI s and a loop l, we estimate whether l cuts all connections from s's class [s] (heuristic I). If l does not cross s, we can be sure that it does not; if it crosses s an odd number of times, we can be sure it does. In the case of an even number of crossings, we cannot definitely decide locally – whether [s] is cut completely depends on the global interplay of multiple loops. Instead of performing expensive global checks in this regard, a heuristic, exploiting that l is created on $\mathcal{M}_4$, proved to be highly reliable: we check how many of the four pre-images of s on $\mathcal{M}_4$ are crossed by l. If only one is crossed we consider s not cut, if more are crossed we consider s cut. The rationale is that between two crossings, l has to travel around multiple singularities for these two crossings to lie on different sheets – implying intersections of l with further loops in between. If all crossings are on the same sheet, we usually have a situation as depicted on the right, not signifying a true cut.



Now, for each SI s, we aim at finding the minimal loop $l_{\min}$ cutting it. As this loop necessarily crosses s, we can start a front propagation from each graph vertex v on s and compute the minimal loop through each of them. More precisely, we start from two of the four pre-images of v on $\mathcal{M}_4$ – the two other sheets with opposite directions only lead to the same loops in reverse. The c_α-smallest of those that cut s then is $l_{\min}$ of s, called the *candidate* for s.

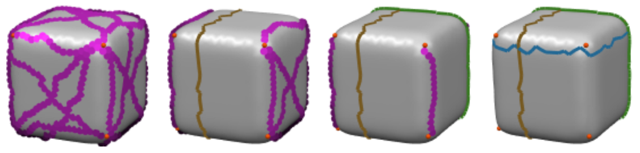


Figure 9: *Visualization of separation indicators (pink) between 8 singularities on a cube: all, and those that remain uncut after the addition of the first, second, and final third loop, respectively.*

Instead of computing the candidate for each SI, i.e. the whole set $L_{\min}$ (cf. Section 4.5), we can heavily increase efficiency in the spirit of lazy evaluation by exploiting the fact that we are always interested in the c_α-largest candidate in each greedy step. If a candidate of s cuts other SIs, their candidate does not (yet) need to be computed as it can only be equal or smaller. To provide an example of the gain in efficiency typically provided by this optimization: for instance on model FERTILITY only 30 candidates are computed in the beginning, they already cut all 10,152 SIs; 11 more are later (re)computed (in repetitions of step 4 of the following algorithm).

Algorithm: With the initial set of candidates C we can then perform an iteration of the greedy algorithm, starting with an empty arrangement A, in the following manner:

1. Add the largest candidate c: $A \leftarrow A \cup \{c\}$, $C \leftarrow C \setminus \{c\}$,
2. remove all SIs that are cut by c,
3. discard candidates from C that are no longer $\mathcal{M}_2$-simple with respect to A,
4. for SIs that are not cut by any curve from C compute new candidates (again in a lazy manner) and add them to C.

These four steps are repeated until no more candidate is available.

To appropriately handle rare cases where in the end some singularities remain unseparated due to the two heuristics, we perform a post-validation: we iteratively check whether paths between two singularities still exist by simple flood-filling within the regions of A (not crossing any loop of A), add these paths as new SIs, and perform further greedy steps (beginning with step 4 to compute new candidates) until no more paths or candidates exist.

5.3 Layout Primalization

The initial embedding of the primal separatrices is obtained as anisotropic field-guided geodesics by performing anisotropic front propagation as described above. The propagation is started from each primal node and restricted to the corridors provided by the dual layout by disallowing crossing of dual loops on $\mathcal{M}_2$.

5.4 Improvement & Parameterization

We have now obtained a topological quad layout Q – some applications can readily exploit this, others rely on its geometry and thus require the initial embedding to be improved. Various methods for the optimization of a layout's geometry and parameterization of its patches are available. Some fix all nodes and optimize the embedding of arcs and patch interiors [Tong et al. 2006; Bommes et al. 2008], other also alter the node positions [Tarini et al. 2011]. As the initial embedding of our regular nodes is a mere byproduct of the separatrix construction described in the last section, we employ a variant of the latter approach, based on iterated local parameterizations, in our implementation. The domain interpolation technique originally employed requires that no triangle spans across non-adjacent patches. As this may not be fulfilled in the initial embedding due to some narrow patches, we deviate as follows:

Instead of optimizing the layout by iteratively performing parameterizations on the fixed triangle mesh $\mathcal{M}$, we construct a quad mesh Q' in the beginning by refining Q to about $\mathcal{M}$'s density (suitably creating a regular grid of quads for each patch, cf. Figure 10). The initial embedding of patch interior vertices is determined using a simple discrete harmonic parameterization of each patch on $\mathcal{M}$. Then the iterated parameterizations are performed on Q' instead of $\mathcal{M}$. By updating Q' after each step of the iterative optimization, quad edges are always aligned with the patch boundaries, guaranteeing simple interpolation domains. The update simply consists in moving the vertices of Q' to the new, improved positions indicated by the local parameterization. To prevent Q' from moving off $\mathcal{M}$, its vertices are *anchored* to faces of $\mathcal{M}$. After each update a vertex is moved onto the closest point on the triangles directly surrounding its anchor face and the concerned triangle becomes its new anchor.

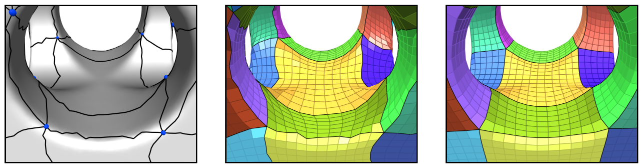


Figure 10: *A layout Q with initial embedding, the refined mesh Q′ with harmonic patch embedding, and its final optimized version.*

6 Extensions

Feature Curves For cases where $\mathcal{M}$ contains feature curves, we can make a simple extension to the algorithm to allow for better alignment of arcs to features. Given a set of curves representing the features (e.g. selected using dihedral angle thresholding for the simplest case of sharp features) we use them as directional constraints in the cross field construction as described by Bommes et al. [2009] such that the cross field is aligned with the features. We then lift each curve to the one sheet of $\mathcal{M}_2$ the line field of which is aligned with the curve tangent, and modify the definition of $\mathcal{M}_2$-simplicity to include that loops may also not intersect these lifted feature curves on $\mathcal{M}_2$. This prevents loops from crossing feature curves in a manner that these curves would not lie within a dual corridor in the final loop arrangement – which would make aligning some separatrix to the feature impossible. After performing the greedy algorithm with this change, separatrices are then not constructed as geodesics but constrained to features where applicable.

It is worth noting that such features constrain the space of admissible loops and in this way already partially prescribe node connectivity. The complexity of the layout is thus predetermined to a degree depending on the comprehensiveness of the feature curve network.

Additional Loops The presented algorithm creates layouts A that are topologically minimal in the sense that no loops are added that do not contribute to singularity separation. This is a desirable property, allowing us to obtain simple layouts. But depending on the geometry of $\mathcal{M}$, the addition of some further loops can potentially have a much larger impact on the improvement of geometric fidelity than on the concomitant decrease of simplicity as it can alter the node connectivity in the primal layout – which might allow for better curvature and feature alignment. Loops whose addition to A can cause such changes can be identified in the following way: each SI lifts to two curves on $\mathcal{M}_2$; we say an SI is *doubly-cut* if both of these are cut by the loops of A on $\mathcal{M}_2$. A further loop causes a change to node connectivity if its addition to A doubly-cuts an SI that, so far, was only singly-cut. This is based on the observation that singly-cut SIs might appear in the primal layout as separatrices (in the form of curves homotopic to the SI), while doubly-cut SIs do not. Figure 11 shows a prototypical example of the situation.

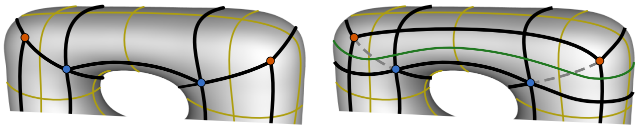


Figure 11: *Left: Primal quad layout (black) induced by a loop arrangement (yellow). Right: Adding a further loop (green) alters the irregular node connectivity: the dashed connections vanish, caused by the cuts (by yellow and green loops) on both sheets of $\mathcal{M}_2$.*

In order to exploit this possibility, after the greedy algorithm has been performed, we can, where possible, construct candidate loops doubly-cutting the SIs that, so far, are not. Then the quality improvement can be assessed for each of these loops by rating the separatrices in the corresponding primal layouts based on their cross field deviation. Starting with the one providing the highest improvement, loops can then iteratively be added until no further loop is admissible or no further gain possible. The assessment can quickly be made based on the initial embedding, or, more accurately, based on the final improved embedding. Notice that it is also possible to remove a loop from A if, by the addition of others, it becomes redundant (= all SIs cut by the loop are also cut by other loops).

7 Results

In this section we show some exemplary results obtained by our implementation of the presented method. Figure 12 depicts the result layouts, the statistical facts are summarized in the following table:

Model	Singularities	Loops	Nodes	Patches
TETRATHING	24	18	80	84
FERTILITY	48	24	92	98
BLOCK	48	20	72	76
3HOLES	16	12	16	20
ELK	52	22	86	86
JOINT	24	16	77	79
GUY	40	18	170	168
ROCKERARM	30	15	115	115
BOTIJO	72	36	213	221
LEVER	83	36	269	275

With our current implementation the processing from the initial construction of the coverings with the fields, over creation of SIs and execution of the greedy algorithm, to the primalization of the layout takes from a few seconds to a few minutes on a commodity PC (model complexities between 20K and 122K faces). The processing is clearly dominated by the front propagations performed to construct candidate loops – which can easily be parallelized.

As feature constraints (cf. Section 6) only the trivially detectable sharp edges on JOINT and LEVER have been used – advanced detection techniques for smooth feature edges were not employed.

Comparison We applied our method to several datasets also taken as input by Tarini et al. [2011] and Bommes et al. [2011] for their simplification methods. While on model ROCKERARM the patch count of our result is higher by a factor of 1.17 compared to the result of Tarini et al., on model 3HOLE the same obvious layout was achieved, and our layouts of models JOINT, BOTIJO, FERTILITY, and LEVER are simpler by factors of 1.4, 2.1, 2.6, and 2.8, respectively (the BOTIJO and LEVER results of Tarini et al. can be found in their supplemental material). In comparison to Bommes et al. the layout complexities in terms of patch count achieved by our Dual Loops Meshing approach are even simpler by factors of 1.5, 3.3, 4.7, and 5.4 on models ROCKERARM, LEVER, BOTIJO, and FERTILITY, respectively. This generally large difference is explained by the fact that their method is based on removing only certain helical configurations from the base complex.

Surely, the possibility cannot be ruled out that the greater layout simplicity might come at the expense of geometric quality to some extent, but visual inspection and comparison of the results does not reveal distortions which would generally be considered intolerable or out of proportion.

The general advantage of the simplification-based methods is their explicit control of the geometric quality: the deviation from the

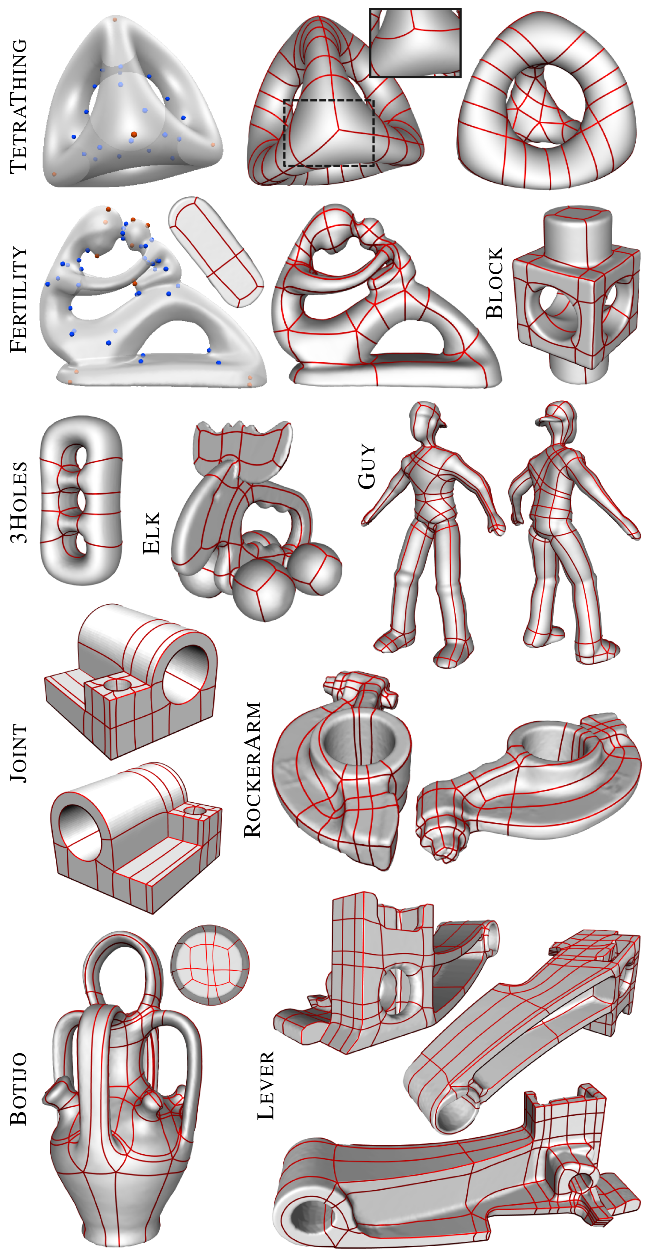
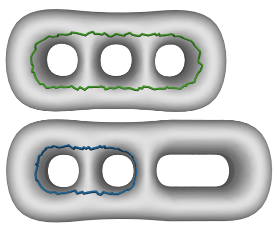


Figure 12: *Patch layouts created by our method (top left: visualization of cross field singularities; index $\frac{1}{4}$ in orange, $-\frac{1}{4}$ in blue). The inset at example* TETRATHING *shows the result before an additional loop which increased separatrix/field alignment was added after the greedy algorithm (as described in Section 6). The* FERTILITY *and* BOTIJO *insets show bottom views of the models.*

start configuration can be kept track of and be limited as desired. Furthermore, any previous state can always be used as a fallback solution. In this sense such methods can be considered safer than from-scratch-construction approaches like ours – especially when geometric fidelity is of higher importance than simplicity.

On the downside, it is not easy to drive such simplification processes in a well-targeted manner; roll-back mechanisms can be necessary to undo series of previous steps in order to escape from dead-ends. The robust automatic generation of the quad meshes taken as input is a further non-trivial subtask. These input meshes' edge directions are then implicitly taken as geometrically optimal reference, but, assuming the input mesh is generated in a common field-guided manner, this reference will often already deviate from the original field (in the sense that quad mesh edges are not well-aligned with the field everywhere) as such fields rarely are integrable and can only be aligned to approximately during meshing. The finite density of the mesh further contributes to this deviation.

The Parameter α While variation of α could be expected to be useful for controlling the layout complexity to some extent, it should be mentioned that this relation is not linear (nor fully monotonic) due to the discrete decisions implicitly made by the routing of the loops. Especially for small α (<10) loops tend to become too unaligned with the field, this incompatibility sometimes leading to less plausible, complex loops being necessary towards the end of the algorithm. On the other hand, increasing α beyond ~ 70 usually does not lead to any more changes due to accuracy limitations of the discrete propagation process. The setting $\alpha = 30$ proved to work very well quite generally and all the presented layouts have been obtained with this fix setting. A prototypical case where a larger α would be advantageous is depicted here exemplarily: while on the upper original THREEHOLES model $\alpha = 30$ leads to the desired green loop (dual to the outer quad cycle which can be seen in Figure 12), it yields the blue loop after some artificial elongation because at some point the deviation from the principal directions necessary for this shortcut weighs lower than the additional length needed for the full round. Nevertheless, at the end of the greedy algorithm we still obtain a valid, though less qualitative, layout.



7.1 Limitations & Future Work

The proposed approach employs a greedy strategy for practicability. Employing a non-greedy, globally optimizing loop selection would be very attractive – but is not straightforward due to the interdependencies caused by the necessary $\mathcal{M}_2$-simplicity. Further, while control over the simplicity of the primal layout is directly given in the dual setting, control over the geometric fidelity is, to some extent, indirect due to the loose relation between dual and primal geometry. Integrating a more direct control could be advantageous.

While our results already show good simplicity, it can be expected that even simpler layouts could be achieved when determination of singularities and their connectivity would not be performed in two separate stages but in an integrated approach. The most important aspect to be exploited in this regard seems to be that singularities can be varying in "confidence": some lie stably at distinct curvature extrema while others are rather arbitrary in their position or could even be merged with others without problems. To ultimately be able to automatically generate layouts comparable to those constructed manually by professionals another singularity-concerned aspect will have to be considered: in manually designed layouts one can often find some irregular nodes that are rather placed by tactical than by purely geometrical considerations. These nodes might be suboptimal in terms of the distortion they induce locally, but are used anyway to keep separatrices local that would otherwise wind around the whole model, creating numerous additional patches.

Our main focus in this work was on obtaining the layout structure – for the geometry optimization we currently employ a plain relaxation procedure (cf. Section 5.4) oblivious to the field of principal directions. Additionally exploiting this information should allow us to achieve improved alignment of layout arcs and nodes also to non-sharp features (as prominent, e.g., in BLOCK or ROCKERARM).

We, so far, employ our method for the construction of general quad layouts. An interesting avenue for future work is the investigation of possibilities to adapt our framework to restricted classes that are of interest in certain applications, e.g. quad layouts without self-crossing dual cycles for hexahedral meshing [Müller-Hannemann 1998] or polycube meshes [Tarini et al. 2004], which also still lack a high-quality automatic and intrinsic construction method.

8 Conclusion

We presented a novel method for the creation of simple all-quadrilateral patch layouts by means of the construction of loop arrangements as duals of these layouts. This dual perspective beneficially exhibits the global structural implications involved in quad layouts more directly. Our method is built on a theoretical framework based on suitably restricted loop spaces on branched coverings of the underlying surface induced by principal curvature cross fields. It allows us to guarantee valid dual layouts which furthermore show good geometric fidelity. We described an implementation exploiting these findings that efficiently creates such layouts based on a greedy approach. There are numerous applications that can benefit from the results and we see interesting directions for future work based on the presented ideas.

Acknowledgements

We would like to thank Henrik Zimmer for his technical assistance. Models have been obtained from the AIM@SHAPE repository and the Image-based 3D Models Archive, Télécom Paris. The GUY model was initially created using Cosmic Blobs® by Dassault Systèmes Solidworks Corp. Funding for this work was provided by the DFG Cluster of Excellence *UMIC*, grant DFG EXC 89.

References

BOIER-MARTIN, I. M., RUSHMEIER, H. E., AND JIN, J. 2004. Parameterization of triangle meshes over quadrilateral domains. In *Proc. SGP '04*, 197–208.

BOMMES, D., VOSSEMER, T., AND KOBBELT, L. 2008. Quadrangular parameterization for reverse engineering. *Mathematical Methods for Curves and Surfaces*, 55–69.

BOMMES, D., ZIMMER, H., AND KOBBELT, L. 2009. Mixed-integer quadrangulation. In *Proc. SIGGRAPH 2009*, 1–10.

BOMMES, D., LEMPFER, T., AND KOBBELT, L. 2011. Global structure optimization of quadrilateral meshes. *Computer Graphics Forum 30*, 2, 375–384.

COHEN-STEINER, D., ALLIEZ, P., AND DESBRUN, M. 2004. Variational shape approximation. In *Proc. SIGGRAPH 2004*, 905–914.

DANIELS, J., SILVA, C. T., SHEPHERD, J., AND COHEN, E. 2008. Quadrilateral mesh simplification. *ACM Trans. Graph. 27*, 5, 148.

DANIELS, J., SILVA, C. T., AND COHEN, E. 2009. Semi-regular quadrilateral-only remeshing from simplified base domains. *Comput. Graph. Forum 28*, 5, 1427–1435.

DONG, S., BREMER, P.-T., GARLAND, M., PASCUCCI, V., AND HART, J. C. 2006. Spectral surface quadrangulation. In *Proc. SIGGRAPH 2006*, 1057–1066.

ECK, M., AND HOPPE, H. 1996. Automatic reconstruction of b-spline surfaces of arbitrary topological type. In *Proc. SIGGRAPH 96*, 325–334.

Eppstein, D., Goodrich, M. T., Kim, E., and Tamstorf, R. 2008. Motorcycle Graphs: Canonical Quad Mesh Partitioning. *Computer Graphics Forum 27*, 5, 1477–1486.

Erickson, J., and Whittlesey, K. 2005. Greedy optimal homotopy and homology generators. In *Proc. 16th Ann. ACM-SIAM Symp. Discrete Algorithms*, 1038–1046.

Huang, J., Zhang, M., Ma, J., Liu, X., Kobbelt, L., and Bao, H. 2008. Spectral quadrangulation with orientation and alignment control. *ACM Trans. Graph. 27*, 5, 147.

Ji, Z., Liu, L., and Wang, Y. 2010. B-mesh: A modeling system for base meshes of 3d articulated shapes. In *Proc. Pacific Graphics '10*, 2169–2178.

Kälberer, F., Nieser, M., and Polthier, K. 2007. Quadcover - surface parameterization using branched coverings. *Computer Graphics Forum 26*, 3, 375–384.

Kovacs, D., Myles, A., and Zorin, D. 2011. Anisotropic quadrangulation. *Comp. Aided Geom. Design 28*, 8, 449–462.

Krishnamurthy, V., and Levoy, M. 1996. Fitting smooth surfaces to dense polygon meshes. In *Proc. SIGGRAPH 96*, 313–324.

Lai, Y.-K., Jin, M., Xie, X., He, Y., Palacios, J., Zhang, E., Hu, S.-M., and Gu, X. 2010. Metric-driven rosy field design and remeshing. *IEEE Trans. Vis. Comput. Graph. 16*, 1, 95–108.

Marinov, M., and Kobbelt, L. 2004. Direct anisotropic quad-dominant remeshing. In *Proc. Pacific Graphics '04*, 207–216.

Müller-Hannemann, M. 1998. Hexahedral mesh generation by successive dual cycle elimination. In *Int. Meshing Roundtable 98*, 379–393.

Murdoch, P., Benzley, S., Blacker, T., and Mitchell, S. A. 1997. The spatial twist continuum: a connectivity based method for representing all-hexahedral finite element meshes. *Finite Elem. Anal. Des. 28*, 137–149.

Myles, A., Pietroni, N., Kovacs, D., and Zorin, D. 2010. Feature-aligned T-meshes. In *Proc. SIGGRAPH 2010*, 117:1–117:11.

Palacios, J., and Zhang, E. 2007. Rotational symmetry field design on surfaces. In *Proc. SIGGRAPH 2007*, 55:1–55:10.

Panozzo, D., Puppo, E., Tarini, M., Pietroni, N., and Cignoni, P. 2011. Automatic construction of quad-based subdivision surfaces using fitmaps. *IEEE Trans. Vis. Comput. Graph. 17*, 10, 1510–1520.

Ray, N., Li, W. C., Lévy, B., Sheffer, A., and Alliez, P. 2006. Periodic global parameterization. *ACM Trans. Graph. 25*, 1460–1485.

Ray, N., Vallet, B., Li, W. C., and Lévy, B. 2008. N-symmetry direction field design. *ACM Trans. Graph. 27*, 10:1–10:13.

Ray, N., Vallet, B., Alonso, L., and Levy, B. 2009. Geometry-aware direction field processing. *ACM Trans. Graph. 29*, 1, 1:1–1:11.

Schmidt, R., Grimm, C., and Wyvill, B. 2006. Interactive decal compositing with discrete exponential maps. In *Proc. SIGGRAPH 2006*, 605–613.

Sethian, J. A., and Vladimirsky, A. 2003. Ordered upwind methods for static hamilton-jacobi equations: Theory and algorithms. *SIAM J. Numerical Analysis 41*, 1, 325–363.

Tarini, M., Hormann, K., Cignoni, P., and Montani, C. 2004. Polycube-maps. In *Proc. SIGGRAPH 2004*, 853–860.

Tarini, M., Puppo, E., Panozzo, D., Pietroni, N., and Cignoni, P. 2011. Simple quad domains for field aligned mesh parametrization. *Proc. SIGGRAPH Asia 2011 30*, 6.

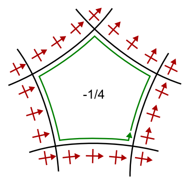
Tierny, J., Daniels, J., Nonato, L. G., Pascucci, V., and Silva, C. 2011. Interactive quadrangulation with reeb atlases and connectivity textures. *IEEE TVCG 99*.

Tong, Y., Alliez, P., Cohen-Steiner, D., and Desbrun, M. 2006. Designing quadrangulations with discrete harmonic forms. In *Proc. SGP '06*, 201–210.

Zhang, M., Huang, J., Liu, X., and Bao, H. 2010. A wave-based anisotropic quadrangulation method. In *Proc. SIGGRAPH 2010.*, 118:1–118:8.

Appendix

Proof of Theorem 4.1 $\mathcal{M}_2$-simplicity and $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4})$ guarantee transversality of all loop intersections on $\mathcal{M}$: a tangential intersection would imply equal tangent lines of the two loops at the intersection; as admissible tangent lines are disjunct on the two sheets of $\mathcal{M}_2$ above each point due to $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4})$, this would imply the intersection happens on one sheet of $\mathcal{M}_2$, contradicting $\mathcal{M}_2$-simplicity. Then, considering R is bounded by a cycle formed by loop segments between k intersections (as depicted for the case $k = 5$), we determine the *turning number* [Ray et al. 2008] of the cycle to show the relation to $I(R)$. Starting at an arbitrary intersection, we travel along the first segment to the next intersection. At that point the intermediate turning number t fulfills $-\frac{1}{4} < t < \frac{1}{4}$ since, due to the $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4})$ constraint, the segment cannot make more than $< \frac{1}{4}th$ of a full turn w.r.t. the field. Now turning to the next segment involves a rotation of the "sector" of allowed tangents by $\frac{\pi}{2}$, leading to $-\frac{1}{4} - \frac{1}{4} < t < \frac{1}{4} - \frac{1}{4}$ along this segment. Repeating this for all k segment switches to get back onto the start segment, we obtain $-\frac{1}{4} - \frac{1}{4}k < t < \frac{1}{4} - \frac{1}{4}k$ for the turning number of the whole cycle. As the turning number of a cycle necessarily is an integer multiple of $\frac{1}{4}$, we have $t = -\frac{1}{4}k$. It is related to $I(R)$ by $t = I(R) - 1$ [Ray et al. 2008], hence $I(R) = 1 - \frac{1}{4}k$.



Proof sketch for Theorem 4.2 Note that we can always add loops that (with infinitesimal distance) run along existing loops without violating $\mathcal{M}_2$-simplicity and without introducing any new irregular regions. The proof is based on the construction of one or more such loops that are then cut and reconnected within R. Figure 13 illustrates these constructions for the concerned cases. Note that reconnections inside the regions are possible in the shown ways because each subregion fulfills $I(R) = 1 - \frac{1}{4}k$.

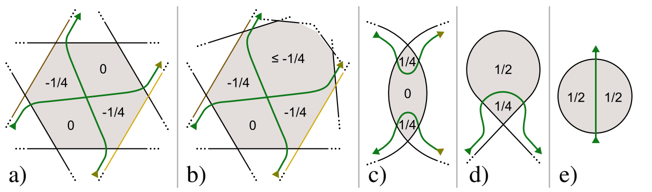


Figure 13: *Schematic illustration of the cases of Theorem 4.2: region with $I(R) = -\frac{1}{2}$ (a), $<-\frac{1}{2}$ (b), $\frac{1}{2}$ (c), $\frac{3}{4}$ (d), 1 (e). Numbers indicate the I of the subregions. The green curve is a single loop that splits the region while running solely along other loops outside the region (a)-(d). In (e) the loop can be connected on the outside due to symmetry (the outside region has this same boundary loop).*