Eurographics Symposium on Geometry Processing 2015
Mirela Ben-Chen and Ligang Liu
(Guest Editors)

Volume 34 (2015), Number 5

Perfect Matching Quad Layouts for Manifold Meshes

Faniry H. Razafindrazaka[†] Ulrich Reitebuch Konrad Polthier

Freie Universität Berlin

Abstract

This paper introduces a new approach to automatically generate pure quadrilateral patch layouts on manifold meshes. The algorithm is based on a careful construction of a singularity graph of a given input frame field or a given periodic global parameterization. A pure quadrilateral patch layout is then derived as a constrained minimum weight perfect matching of that graph. The resulting layout is optimal relative to a balance between coarseness and geometric feature alignment. We formulate the problem of finding pure quadrilateral patch layouts as a global optimization problem related to a well-known concept in graph theory. The main advantage of the new method is its simplicity and its computation speed. Patch layouts generated by the present algorithm are high quality and are very competitive compared to current state of the art.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.5]: Computational Geometry and Object Modelling—;

1. Introduction

Automatic generation of pure quadrilateral patch layouts on manifold meshes has received a lot of interest in recent years. Powerful automatic [CBK12, BCE*13] as well as semi-automatic algorithms [TPSHSH13, CK14a] have been developed, achieving high quality patch layouts comparable to those manually generated by professionals.

Quadrilateral patch layouts or for shortness quad layouts are partitions of 2-manifold surfaces into non-overlapping quadrilateral patches such that any two patches share an edge or vertex or are connected by a chain of adjacent patches. They form a *base domain* of the mesh allowing an easy processing of the underlying surface in a well structured manner.

In mesh processing, the quality of these patches is of high interest. Coarseness and geometric fidelity are practically efficient in applications such as: subdivision surfaces, high-order surface fitting or multiresolution in finite element analysis. Some important quality requirements include: patches close to rectangular shapes, i.e, minimal distortions; patch edges aligning to features of the surface; and finally coarseness, i.e, as few patches as possible. In practice, all these requirements cannot be achieved simultaneously due mainly to metric distortions. A tradeoff between coarseness and geometric fidelity is alas necessary.

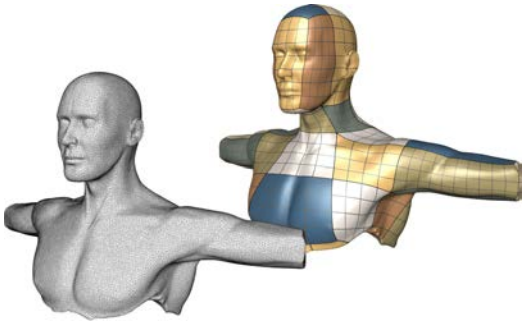
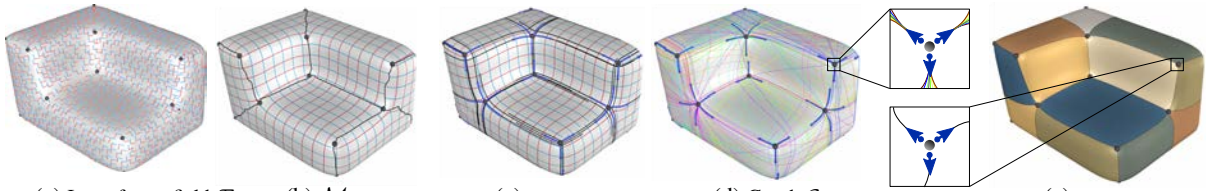


Figure 1: *An example of a perfect matching quad layout on the triangulated* `Torso` *model, produced by the proposed algorithm.*

The present work focuses on the automatic generation of quad layouts consisting of a small number of patches while aligning to geometric features of the surface. Our approach is based on the careful construction of a singularity graph, derived from a given frame field, describing the set of possible layouts of the surface. Our goal is to provide a flexible framework to generate the best possible quad layout in a reasonable amount of time. In Figure 1 is an example of a quad layout generated on the triangulated `Torso` model (75k triangles) formed by 42 patches.

[†] Supported by the Berlin Mathematical School.



(a) Input frame field $\mathcal{F}$ (b) $\mathcal{M}_f$ (c) Growing isolines (d) Graph $\mathcal{G}$ (e) Perfect matching $\mathcal{Q}$

Figure 2: *The fundamental steps of the proposed method, from left to right: (a) Given an input frame field $\mathcal{F}$, for example induced from principal curvature directions, on a surface $\mathcal{M}$ (b) a parameterization $\mathcal{M}_f$ is obtained by removing the curl part of $\mathcal{F}$; (c) an isoline growing approach is used to generate (d) a graph $\mathcal{G}$ whose edges are parametric geodesics connecting singularity ports. Finally, (e) a perfect matching subgraph $\mathcal{Q}$ of $\mathcal{G}$ forming a quad layout of $\mathcal{M}$ is computed.*

1.1. Contributions

We propose a novel algorithm which generates high quality quadrilateral patch layouts on arbitrary 2-manifold meshes. The algorithm uses a novel combinatorial graph derived from the structure of a smooth frame field on the surface. In contrast to iterative greedy approaches, we formulate the quad layout generation problem as a global optimization problem, more precisely, as a constrained minimum weight perfect matching problem. We also propose a possible parameterization of the patches which aligns best to the input frame field.

The proposed framework can be adapted to previous greedy approaches such as [TPP*11] or [CBK12].

1.2. Overview

In classical frame field based parameterization methods [KNP07, BZK09], the generation of a pure quad mesh involves an integer rounding step which is a greedy decision. One choice of integer roundings will alter significantly the global structure of the extracted quad mesh. We avoid this greedy nature by constructing directly the *singularity graph* or *base complex* of the surface using isoline tracing techniques.

The core of the algorithm consists of four fundamental steps for a given surface $\mathcal{M}$ and a frame field $\mathcal{F}$:

1. Generation of a parameterization of $\mathcal{M}$ by removing the curl part of $\mathcal{F}$.
2. Construction of a singularity graph $\mathcal{G}$ in parametric space.
3. Construction of a set of illegal edge crossings I on $\mathcal{G}$.
4. Finding of a minimum weight perfect matching of $\mathcal{G}$ with conflict pair constraints set I, which induces a quad layout of $\mathcal{M}$.

Since we do not require a seamless parameterization of the surface (globally continuous), the first step of the algorithm can be reduced to solving a Poisson equation. This type of parameterization is sometimes called *Poisson parameterization*. The graph $\mathcal{G}$ is constructed by growing isolines emanating from each singularity and analysing the ratio of length of two isolines whenever they intersect. The graph $\mathcal{G}$ is a combinatorial information which will represent our space of quad layout. Layouts of $\mathcal{M}$ are perfect matching subgraphs of $\mathcal{G}$ which, without constraints, are arbitrary polygonal patches. The set I of illegal crossing edges is fundamental in our construction. It will restrict the search of subgraphs, and hence layouts of $\mathcal{M}$, to be only purely quadrilateral. The constrained minimum weight perfect matching proposed in this paper guarantees an optimal quad layout of $\mathcal{M}$ relative to a weight function, balancing coarseness and alignment to geometric features. All these steps are illustrated in Figure 2. Notice that Steps (a) and (b) are not needed if the input is a pure quadrilateral mesh or any periodic global parameterization since the induced frame field is already curl free.

2. Related work

The literature on field aligned global parameterization is vast. A complete survey can be found in [BLP*12]. Our work is based on several previous results using cross fields and line tracing techniques on surfaces.

Field Aligned Global Parameterization One of the first field-aligned global parameterization is the periodic global parameterization of Nicolas Ray et al. [RLL*06], which is later on improved by using a branch covering based approach [KNP07] or a mixed integer formulation [BZK09]. Since these methods rely heavily on good input frame fields, recent works [PPTSH14, ECBK14] focus on generation of good frame fields. This class of parameterization is not in general injective such that more advanced convex constraints are required. Lipman [Lip12] introduces the bounded distortion mapping which explores the maximum deformation of a triangle during the parameterization. The trisector constraint proposed in [BCE*13] uses the Fermat point of a triangle to generate a simple convex constraint per triangle.

Global parameterization methods are particularly useful for quad meshing. Unfortunately, they do not always provide a high-level structure or coarse base mesh of the surface. Other recent works concentrate on the direct generation of coarse quad layouts for a given surface.

Quad Layouts In the past years, there have been several methods to generate high quality quad layouts on surface meshes. These include: direct algorithms, spectral methods, parameterization approaches or field tracing techniques. Direct algorithms work directly on the triangulation of the surface without any high order geometric information. Matthias

Eck [Eck96] uses triangle parings techniques for the purpose of B-spline fitting; a Voronoi partition approach is used in [BMRJ04]; Daniels [DSC09] proposes to quadrangulate the surface by a one step Catmull-Clark subdivision, then uses quad mesh simplification techniques to get a coarse base mesh. Spectral methods use a Morse-Smale complex of the Laplacian eigenfunctions [DBG*06]. Field-aligned parameterization can also generate coarse quad layouts by using large target edge lengths [BCE*13]. Other methods generate initially a fine integer-grid map and build a T-mesh layout [MPKZ10] or optimize the base complex of the induced quad mesh [BLK11, TPP*11]. Recently, several works focus on direct field tracing. Campen et al. [CBK12] trace anisotropic loops to build from scratch the dual of the base complex, followed by a layout optimization [CK14b]; Myles et al. [MPZ14] and Ray et al. [RS14] propose, independently, a robust polyline tracing of rotational symmetric fields. Their techniques produce a patch layout of the surface using the Motorcycle graph algorithm of Eppstein et al. [EGKT08].

Our approach differs from [MPZ14] in two aspects. First, we trace isolines of a parameterization induced from a curl free frame field which avoids the existence of limit cycles. Second, an isoline tracing does not stop when it intersects another one. Instead we evaluate the ratio of the arc lengths of the two isolines from their intersection points to the respective singularities. Accordingly, we construct a graph $\mathcal{G}$ defining the space of possible layouts of the surface.

3. Frame Field Driven Parameterization

In this section, we recall some basic notions of frame field driven parameterization and some fundamental concepts on this class of parameterizations. For all definitions and properties about frame fields, we refer to [RVLL08, PPTSH14]. In this paper, we will restrict to triangle based orthogonal frame fields. We denote by $T_{\mathcal{M}}$ the set of triangles and $E_{\mathcal{M}}$ the set of edges of $\mathcal{M}$.

3.1. Poisson Parameterization

We use the approach proposed in [KNP07, BZK09] which takes a smooth cross-field as input and finds a potential function whose gradients align best to that field. The cross-field can be either given as input or derived, for example, from stable principal curvatures of the surface.

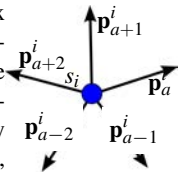
Given a triangle based frame field $\mathcal{F} = \{\mathbf{F}_T\}_{T \in T_{\mathcal{M}}}$, we construct a parameterization map $f = (u,v) : \mathbb{R}^3 \to \mathbb{R}^2$ which minimizes the energy $\sum_{T \in T_{\mathcal{M}}} \|\nabla f - \mathbf{F}_T\|^2$, taking transitions between triangles into account. We do not require f to be an integer-grid map, i.e, a seamless global parameterization of $\mathcal{M}$ with all singularities mapped to integer points in parameter domain. The resulting parameterization is a periodic parameterization also called Poisson parameterization. Figure 2 (b) shows an example of a Poisson parameterization whose parameter lines are continuous everywhere except along a cut graph of the geometry. We denote by $\mathcal{M}_f$ the surface with the metric induced by f. It is governed by transition functions which need to be considered, for example, in computations such as geodesics or isolines tracing.

A point of $\mathcal{M}_f$ is *regular* if its total vertex angle in parameter space is 2π. It is *singular* otherwise. Singularities play an important role in our construction algorithm. If a frame field does not have a singularity, then the present algorithm can be adapted by picking one regular vertex and considering it as a singularity.

A correct total vertex angle at a singularity requires a positive orientation of all triangles adjacent to that singularity in parameter space. It is required by the proposed algorithm. This can be achieved either by using the trisector constraints from [BCE*13] or the bounded distortion mapping from [Lip12]. Notice that we do not need to constrain all triangles of the mesh which makes the computation of the Poisson parameterization relatively fast and close to the global minimum. Fold-overs away from singularities affect only the visual quality of the final results which can be improved afterwards.

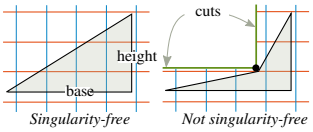
3.2. Ports, Separatrices and Triangles

A *port* is a direction based at a singularity which is tangent to an isoline of $\mathcal{M}_f$. The number of ports at a singularity is derived either from the cross-field index of the singularity or by a direct enumeration approach in parameter domain. We denote by $\mathbf{p}_a^i$ a port based at the singularity s_i, its next counter-clockwise port by $\mathbf{p}_{a+1}^i$ and its next clockwise port by $\mathbf{p}_{a-1}^i$, as shown in the inset Figure. The ordering of the ports is derived in parameter space, using transition functions, similar to [EBCK13].



We call a curve *tangent to a port* if the angle between the curve and the port is less than 45° in the metric of $\mathcal{M}_f$. A *closed separatrix* is a curve on $\mathcal{M}_f$ which connects two singularities (not necessarily distinct). For the case of surfaces with boundary, one of the singularities is allowed to be a boundary vertex. If one of the endpoints is not a singular point, then we call it an *open separatrix*. We denote by $\gamma_{ab}^{ij} : [0,1] \to \mathbb{R}^3$, the closed separatrix which connects the singularities s_i and s_j, tangent to the ports $\mathbf{p}_a^i$ and $\mathbf{p}_b^j$, and by $\gamma_{a\vdash}^i$ the open separatrix starting at the singularity s_i in direction of $\mathbf{p}_a^i$. In our case, the open separatrices are always isolines, closed separatrices not necessarily.

A *triangle* is a right triangular patch whose base and height are isolines of $\mathcal{M}_f$. The base is the longer of the right-angled sides. A triangle is *singularity-free* if the parametric geodesic homotopic to the combined base-height curve defines the hypothenuse of the triangle, which is a straight



line on $\mathcal{M}_f$. It is not otherwise and bends exactly at a inner singularity (see inset Figure). In our context, we consider only triangles whose hypotenuses are closed separatrices, i.e, triangles whose non right-angled vertices are singularities.

4. Perfect Matchings

Given a weighted graph $\mathcal{G} = (V_\mathcal{G}, E_\mathcal{G}, w_e)$, a *matching* $\mathcal{Q}$ in $\mathcal{G}$ is a subgraph of $\mathcal{G}$ such that no two edges share a vertex. A matching is *perfect* if all vertices of $\mathcal{G}$ are in $\mathcal{Q}$. In other words, each vertex of $\mathcal{G}$ is pairwise matched. In a quad layout, each port is connected to exactly one other port by a separatrix, which makes the quad layout problem a perfect matching problem in $\mathcal{G}$. A minimum (resp. maximum) weight perfect matching is a perfect matching with a minimal (resp. maximal) total edge weight.

The study of matchings has been a long time discussion in the field of graph theory. A complete overview can be found in [PL86]. In geometry processing, the Blossom-Quad algorithm [RLS*12] uses perfect matching to find a global pairing of triangles on manifold meshes, which then induces a quadrangulation of the surface. Our algorithm uses the minimum weight perfect matching problem with conflict pair constraints (MWPMPC) introduced by [DPSW11] and extended in [OZP13]. In our context, the conflict pair constraints are described in Section 6.2 as illegal crossing constraints which assure the quad topology of the resulting minimum weight perfect matching.

5. Basic Concept

To understand the basic idea behind the proposed algorithm, let us consider four points in the plane, which are placed on the corners of the unit square. Our goal is to use matching theory to find a quadrangulation of these points, which obviously is the unit square, with the knowledge that the points have valence two. First, we duplicate each point and connect each two of the duplicated points that do not share the same position by a straight edge. The resulting graph $\mathcal{G}$ has eight vertices and 24 edges. We define the weight w_e of an edge e in $\mathcal{G}$ to be zero if it is axis aligned and one otherwise. $\mathcal{G}$ has several minimum weight perfect matchings with total weight zero. To avoid the non-quad solutions, pairs of edges which share the same start and end quad corners are not allowed to appear simultaneously in the solution.

We generalize this basic example on surface meshes using: the singularities of the frame field $\mathcal{F}$ as input points; the ports as the vertices of the graph $\mathcal{G}$; closed separatrices as edges (see Figure 2 (d)); and the quad property of $\mathcal{M}_f$ to derive the conflict pair constraints.

6. Method

In this section, we construct the graph $\mathcal{G}$ together with a set of illegal crossing constraints I. We then give a global formulation of the quad layout generation problem on manifold meshes.

6.1. Construction of $\mathcal{G}$

Growing Isolines At each singularity, we grow simultaneously all isolines, at unit parametric length speed, in direction of the ports. These lines are open separatrices. If two of them intersect, we build a closed separatrix from the two end ports using a ratio test. We add a new edge corresponding to that separatrix in $\mathcal{G}$. We define a distance function L^i_a which measures the parametric arc length of an open separatrix $\gamma^i_{a\vdash}$ from a singularity s_i to a point $p = \gamma^i_{a\vdash}(t)$. Now, consider two ports $\mathbf{p}^i_a, \mathbf{p}^j_b$ and the corresponding open separatrices $\gamma^i_{a\vdash}, \gamma^j_{b\vdash}$ intersecting at a point m of $\mathcal{M}_f$ (see Figure 3). A new edge is added to $\mathcal{G}$ according to the following conditions **C1**: if $\Delta s_i m s_j$ is a singularity-free triangle and

1. $L^j_b(m)/L^i_a(m) < 1$, then we add an edge connecting $\mathbf{p}^i_a$ to $\mathbf{p}^j_{b+1}$ in $\mathcal{G}$ if s_j is on the right side of $\gamma^i_{a\vdash}$, or $\mathbf{p}^i_a$ to $\mathbf{p}^j_{b-1}$, if s_j is on the left side.
2. $L^j_b(m)/L^i_a(m) > 1$, then we add an edge connecting $\mathbf{p}^j_b$ to $\mathbf{p}^i_{a+1}$ in $\mathcal{G}$ if s_i is on the right side of $\gamma^i_{a\vdash}$, or $\mathbf{p}^j_b$ to $\mathbf{p}^i_{a-1}$, if s_i is on the left side.
3. $L^j_b(m) = L^i_a(m)$, then we add one edge following Condition 1 and an other edge following Condition 2, in $\mathcal{G}$. Both edges should not appear simultaneously in the final quad layout to avoid degeneration (see Section 6.2).

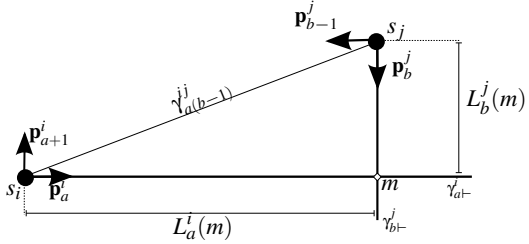


Figure 3: *Creation of an edge by evaluating ratios at an intersection point of two separatrices.*

In Figure 3, the new edge is a closed separatrix $\gamma^{ij}_{a(b-1)}$ defined geometrically as the parametric geodesic connecting s_i at the port $\mathbf{p}^i_a$ and s_j at the port $\mathbf{p}^j_{b-1}$ homotopic to the curve $[\gamma^i_{a\vdash} \circ \gamma^j_{b\vdash}]$. Geometrically, the curve is computed by unfolding the triangle run containing $[\gamma^i_{a\vdash} \circ \gamma^j_{b\vdash}]$ in parametric space and using [LP84]. The transitions between triangles have to be considered in the straightening process.

Singularity-free Triangles In the metric of $\mathcal{M}_f$, singularities behave like small holes. Geodesics can get stuck easily on these special points. In our construction, edges are parametric geodesic curves homotopic to the combined base and height curves of triangles. Allowing non straight edges will produce degenerate quadrilateral patches in the final layout.

Therefore, we need to construct only singularity-free triangles during the isoline growing step.

For each port, we associate two propagation windows: left window and right window which initially have an angle of 45° with the port. Consider an open separatrix $\gamma^0_{a_0\vdash}$, starting at a singularity s_0 in direction of $\mathbf{p}^0_{a_0}$, as illustrated in Figure 4. Suppose that it intersects n other open separatrices $\gamma^1_{a_1\vdash}, \gamma^2_{a_2\vdash}, \ldots, \gamma^n_{a_n\vdash}$ at the points $m_1, m_2, \ldots, m_n$. Additionally, suppose that all the corresponding base singularities satisfy one of the ratio test in condition **C1** but not necessarily the singularity-free triangle condition. A triangle is singularity-free if the angle of its hypotenuse with $\gamma^0_{a_0\vdash}$ is less than the latest updated window size. Initially, the window size ψ is 45°. It is then updated according to the first singularity satisfying condition **C1**. $\Delta s_0 m_1 s_1$ is always singularity-free since there is no other singularity preceding it within the starting window size. Depending on the position of s_1 (left or right of $\gamma^0_{a\vdash}$), the corresponding new window size becomes $\psi = \arctan\left(L^1_{a_1}(m_1)/L^0_{a_0}(m_1)\right)$. Along $\gamma^0_{a_0\vdash}$, for $k \geq 2$, we collect singularities s_k satisfying

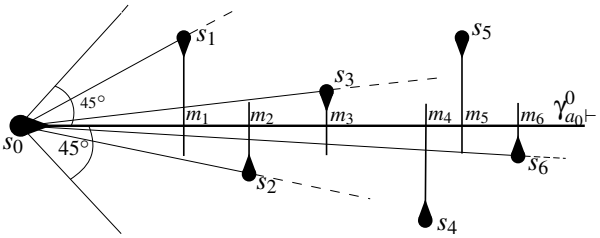


Figure 4: *Singularities satisfying one of the ratio test along the open separatrix $\gamma^0_{a_0\vdash}$. In this figure, only s_1, s_2, s_3 and s_6 agree with the singularity-free triangle constraint*

$$\arctan\left(L^k_{a_k}(m_k)/L^0_{a_0}(m_k)\right) < \psi,$$

and assigning the value of ψ to be the newly computed angle. This window reduction approach makes sure that no triangle contains a singularity and hence produces always straight geodesics on $\mathcal{M}_f$.

Edge weights Recall that an edge $e \in \mathcal{G}$ connects two singularities s_i and s_j at the ports. It is the hypotenuse of the triangle having $s_i m$ as base and $m s_j$ as height where m is the intersection point of the two separatrices $\gamma^i_{a\vdash}$ and $\gamma^j_{b\vdash}$. We define the weight of e as the linear combination of the base length and height length of the corresponding triangle. Mainly,

$$w_e = L^i_a(m) + \alpha L^j_b(m). \quad (1)$$

Here, α is a parameter used to balance between shortness and alignment to the input frame field (also used in [TPP*11]).

A Stopping Criterion In general, the isolines of $\mathcal{M}_f$ have infinite lengths. The line growing approach can never end unless either all open separatrices end at the boundary, or in the very rare case where they all hit singularities. This situation happens, for example, when the surface is very symmetric. We need a correct way to stop the growing procedure and make sure that enough edges are in $\mathcal{G}$ in order to get a large space of possible solutions.

One approach is to fix a maximum growing length $L_{\max}$ relative to the diameter of the geometry and to terminate when all isolines reach the prescribed length. This approach will produce enough edges in $\mathcal{G}$ but can be very slow in practice, especially if $\mathcal{M}_f$ contains too many singularities. A more elegant approach is an adaptive stopping criterion. Mainly, a line stops growing while others continue in a motorcycle graph like approach. Our termination condition is to stop an isoline when it starts to spiral. However, this procedure involves the careful understanding of the structure of spirals (a priori unknown) on $\mathcal{M}_f$ and a robust algorithm to detect them as done in [BLK11] on quadrilateral meshes.

If a growing line intersects another one the second time, then one of them is susceptible to be a spiral. It is then convenient to stop it. The decision depends on a fixed threshold ρ controlling the pitch of helices of $\mathcal{M}_f$. Consider two open separatrices $\gamma^i_{a_0\vdash}$ and $\gamma^j_{a_1\vdash}$ which intersect each other the first time at a point m_0 and the second time at a point m_1 of $\mathcal{M}_f$. If

$$|L^i_{a_0}(m_1) - L^i_{a_0}(m_0)| > \rho |L^j_{a_1}(m_1) - L^j_{a_1}(m_0)| \quad (2)$$

then $\gamma^i_{a_0\vdash}$ stops growing. Else if

$$|L^j_{a_1}(m_1) - L^j_{a_1}(m_0)| > \rho |L^i_{a_0}(m_1) - L^i_{a_0}(m_0)| \quad (3)$$

then $\gamma^j_{a_1\vdash}$ stops growing. If $i = j$ and $a_1 = a_0 \pm 1$, then the inequality is checked at the first intersection point of the two separatrices (see Figure 5). In that case, $m_0 = s_i = s_j$. If ρ is large, then the growing step might collect too many unwanted large pitch spirals affecting considerably the size of $\mathcal{G}$ and hence the construction's running time. If ρ is too small, then some isolines might stop too early even if they do not start to spiral, reducing the size of $\mathcal{G}$ and hence the space of possible solutions. In our implementation, choosing the fix value $\rho = 10$ works very well. In Figure 6 are

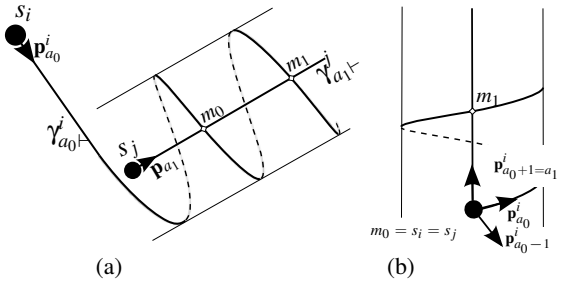


Figure 5: *Spiral detection during the growing line: (a) a separatrix starting from a far way singularity before spiralling in a cylindrical region; (b) separatrices starting at a cylindrical region based at the same singularity.*

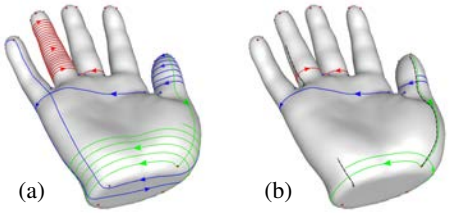
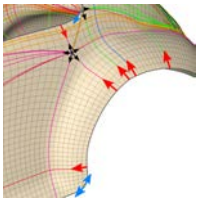


Figure 6: *(a) Example of spirals for a fixed maximum tracing length L_{max}; (b) isolines are stopped before spiraling using $\rho = 10$ in parameteric length unit.*

two examples of stopping strategies. In the first approach (a), we use a fix maximum tracing length $L_{\max}$ for all isolines which produces many unwanted spirals while in the second approach (b), we trigger the spiral detection which removes all unwanted spirals appearing in the first approach.

Boundary If an isoline ends at a boundary then we introduce a *virtual port* tangent to that isoline pointing inside the surface and based at the boundary vertex. We then add a new vertex corresponding to the virtual port in $\mathcal{G}$ together with an edge representing the separatrix, as shown in the inset Figure (red). We do not collect singularity candidates from virtual ports, we only use them to have an uniform framework for surfaces with and without boundary. For singularities lying on boundary curves, we collect only candidates for the ports pointing inside the surface. The ports tangent to the boundary only grow but can also be used to collect candidates for other isolines. The concept of virtual port together with the previous consideration always maintain the quad topology of the boundary induced from $\mathcal{M}_f$. However, it requires that the boundary curves are isolines of $\mathcal{M}_f$. If not, it only assures pure quads for interior patches and arbitrary polygons for boundary patches. Boundary constraints can be added during the generation of Poisson parameterizations as done in [BZK09].



Matching Now, we would like to find a subgraph $\mathcal{Q}$ of $\mathcal{G}$ which forms a quad layout of our surface $\mathcal{M}$. The graph $\mathcal{G}$ is huge and finding a proper quad layout subgraph needs a careful understanding of the edge crossings of $\mathcal{G}$ on $\mathcal{M}_f$. By choosing exactly one edge per port (one or zero edges per virtual port), we obtain a perfect matching subgraph forming an arbitrary layout of $\mathcal{M}$ which is not necessarily a pure quad layout. In Figure 7 (middle) is a perfect matching without constraints where the crossing red edges are for example in conflict. This is resolved in Figure 7 (right) by allowing at most one edge of each conflict pair to appear in the perfect matching.

6.2. Illegal Crossings

The non-quad patches appearing in the unconstrained matching are caused by crossing edges which are locally aligned

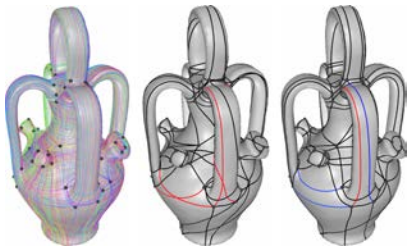
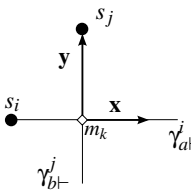


Figure 7: *Examples of a perfect matching subgraph of $\mathcal{G}$ on the `Botijo` model where the middle one is unconstrained and the right one constrained for $\alpha = 0$.*

to the same parameter lines. We detect these edges and define a conflict set I which assures the quad topology of the final layout. Two crossing edges in $\mathcal{G}$ are aligned locally to the same parameter line on $\mathcal{M}_f$ if the two triangles associated to the edges intersect and have parallel bases. We then say that the crossing is *illegal*, and it is *legal* otherwise. We denote by I the set of all illegal crossings of $\mathcal{G}$,

$$I = \{(e_0, e_1) \in E^2_{\mathcal{G}}, e_0 \,\text{crosses}\, e_1 \,\text{illegally}\}. \qquad (4)$$



We construct I as follows. Consider two open separatrices $\gamma^j_{a\vdash}$ and $\gamma^j_{b\vdash}$ intersecting at a point m_k (see inset diagram). We define an orthogonal frame $\{\mathbf{x}, \mathbf{y}\} = \{\mathbf{s_i m_k}, \mathbf{m_k s_j}\}$ at m_k in parametric space. Now, consider all edges whose triangles have a side on $\gamma^j_{a\vdash}$ or $\gamma^j_{b\vdash}$. In this local coordinate system, we can do 2D triangle intersection checks without ambiguity.

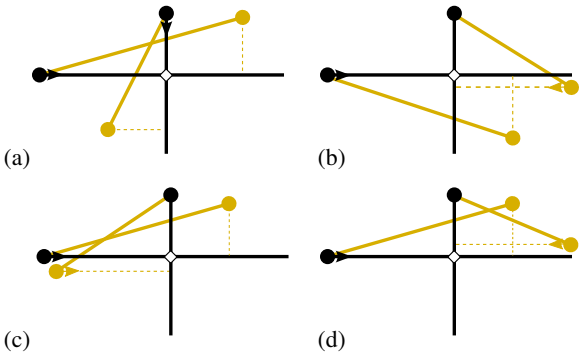


Figure 8: *Example of edge crossings (yellow) represented in a common local coordinate system. In (a) and (b) are examples of legal crossings where in (c) and (d) are illegal crossings.*

We say that an edge is **x**-aligned (resp. **y**- aligned) if the base of its triangle is parallel to **x** (resp. **y**). Two edges which cross and have different alignment have a legal crossing. Two edges which cross and have the same alignment have an illegal crossing. It is important that they cross locally because it can happened that two edges with the same alignment do not cross although their triangles intersect, as illustrated in Figure 8 (b). We also add the pairs of edges gener-

ated by the case $L_a^i(m) = L_b^j(m)$ in Section 6.1 to I to avoid degenerate patches in the final quad layout.

We summarize all these steps in Algorithm 1. The set K contains all intersection points of the open separatrices satisfying condition **C1**. The set I can be empty if $\mathcal{G}$ is already a quad layout matching of $\mathcal{M}$. As we can see in Table 2, the size of I is far larger than $E_{\mathcal{G}}$.

6.3. Global Optimization Problem

So far, we have constructed a graph of closed separatrices $\mathcal{G}$ together with a set of edge pairs having mutual illegal crossings. We would like to construct a quad layout as a perfect matching subgraph of $\mathcal{G}$ which minimizes the total edge weights $\sum w_e$ and is guaranteed to be pure quad. The problem can be formulated as a MWPMPC. Mainly, finding a quad layout of $\mathcal{M}$ is equivalent to solving the following binary program $\mathbf{P}_1$,

$$\text{minimize} \quad \sum_{e \in E_{\mathcal{G}}} w_e x_e \tag{5}$$

$$\text{s.t.} \quad \sum_{e \in \text{Adj}(v)} x_e = 1, \quad \text{for all not virtual } v \in V_{\mathcal{G}} \tag{6}$$

$$x_{e_1} + x_{e_2} \leq 1, \quad \text{if } (e_1, e_2) \in I \tag{7}$$

$$x_e \in \{0,1\}, \quad \text{for all } e \in E_{\mathcal{G}} \tag{8}$$

where $\text{Adj}(v)$ is the set of edges sharing a vertex $v \in V_{\mathcal{G}}$. The objective functional minimizing the total edge weights is given by (5). Constraint (6) assures that per vertex of $\mathcal{G}$, exactly one incident edge is taken. In $\mathcal{M}_f$, this is equivalent to assigning exactly one closed separatrix to each port. Condition (7) is a translation of the illegal constraints into inequality constraints. It means that if a pair of edges is in I, then only one or none of the two edges is in the solution. It assures the quad topology of the final layout. The binary condition (8) means that the edge is either in the solution or not. The integer program (5), (6) and (8) gives the classical formulation of a minimum weight perfect matching problem.

Proposition 1 The solution of the linear program (5) subject to (6), (7), (8) is a pure quadrilateral patch layout of $\mathcal{M}$.

Suppose that there exists a patch P in the solution which is a n-gon, $n \neq 4$ bounded by legal edges. Consider the parameterized surface element S which contains P. In S, we can characterize locally to which parameter lines the edges of P are aligned. If n is odd, then a cyclic uv-assignment, according to the tangential property, of the edges will always produce a uu or a vv intersecting edges contradicting legality. If n is even, then a consistent uv-assignment is impossible since P does not contain a singularity. In fact, it is only possible for $n = 4$ or if P contains a singularity which is not the case here. A similar proof can also be found in [CBK12] using the winding number of the boundary edges.

Adding Constraints In practice, especially for CAD models, the input geometries are dominated by sharp edges. It is desirable to have some separatrices following these geometric features. If these edges lie on the same isoline in $\mathcal{M}_f$, then the corresponding edge e in $\mathcal{G}$ can be simply hard constrained by taking $x_e = 1$. Sharp edge constraints are added during the Poisson parameterization as done in [BZK09].

Algorithm 1 Construction of the set of illegal crossings I

Require: Set of open separatrices intersection points K

```
Put I = ∅
for m_k ∈ K do
    Take the two separatrices γ^i_{a⊢} and γ^j_{b⊢} intersecting at m_k.
    {x, y} = {s_i m_k, m_k s_j}
    {e_l}_{l=0,...,n} the set of edges whose triangles have a side on γ^i_{a⊢} or γ^j_{b⊢} .
    for l ← 0 to n do
        for r ← l + 1 to n do
            if e_l and e_r are both x- or y-aligned then
                if e_l and e_r intersect then
                    I = I ∪ {(e_l, e_r)}
                end if
            end if
        end for
    end for
end for
```

7. Results and Analysis

We apply our algorithm on several datasets which are mainly classical benchmark models. The input of the algorithm is either a frame field or directly a Poisson parameterization $\mathcal{M}_f$. We use default parameters for all quad layout generated in this paper by choosing a balance $\alpha = 5$, a maximum separatrix length $L_{\max} = 3/h \times \text{Diam}(\mathcal{M})$, a sizing field $h = 10^{-2} \times \text{Diam}(\mathcal{M})$, and a spiral detection constant $\rho = 10$. The choice of α is motivated by the experiment illustrated in Figure 9. For the benchmarking, we use the frame fields also taken as input from the corresponding previous works. We also use an independent frame field generator based on the continuous reliability weights and gradient relaxation of Nieser [Nie12], and the trisector constraints [BCE*13] for the Poisson parameterization generation.

Numerical Solver We use IBM's Ilog CPLEX (`www.ibm.com`) to solve the integer program. The algorithm was ran on a Core i7-2720 2.2Ghz 8Gb RAM using Java as programming language.

Robustness The algorithm runs very well on badly triangulated meshes. In Figure 10 (a) is a bad triangulation of the `Joint` model which contains many skinny triangles. Our method still produces correct quad layout on that model. We

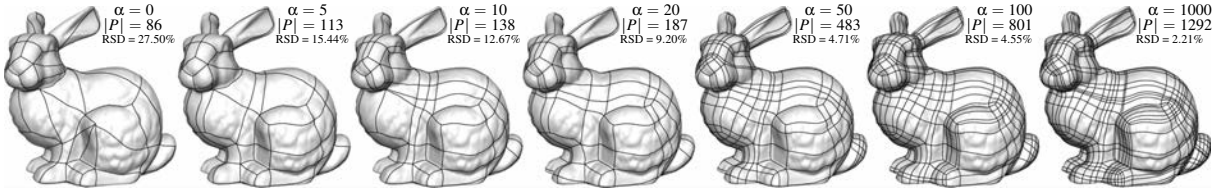


Figure 9: *Several quad patches on the* `Bunny` *model for different values of* α. $|P|$ *denotes the number of patches and RSD is the Relative Standard Deviation of the patch angles measured in parametric space. Big values of* α *give a better alignment to the input field at the cost of increasing number of patches. We find* $\alpha = 5$ *to be a good balance.*

also generate random noises on some region of the `Bumpy Sphere` model with bad shaped triangles, over-foldings and self-intersections. The algorithm still produces valid quad patches, as illustrated in Figure 10 (b).

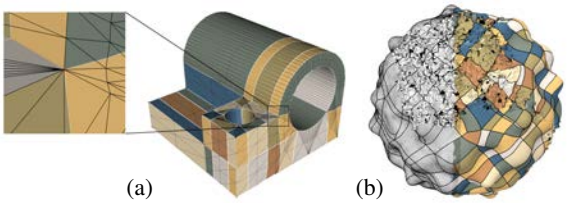
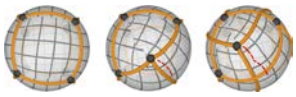


Figure 10: *(a) A bad triangulation of the* `Joint` *model. (b) Exaggerated noise on the* `Bumpy sphere` *model.*

Singularity placement In the quad layout, the position of singularities is very important to assure geometrical meaningful patches. Our algorithm is not very sensitive to local changes of singularity positions but can generate a completely different layout (but still optimal with respect to the new singularity positions) if the change induces a global effect on the field as illustrated in the inset Figure.



Comparison We compare against the Dual loops meshing (DLM) [CBK12] and the Integer-grid map (IGM) [BCE*13] for the same models and cross-fields. The results are shown in Figure 11 together with a comparison table. Our algorithm produces results as good as both methods. The main advantage of our approach is its speed and control over the final layout. Changing the value of α or adding user constraints can be done in a split second. The Dual loop meshing method is greedy and works on the dual complex. It is not very much affected by singularity placement but unfortunately, it is not clear how user constraints could be incorporated once the layout is computed. The Integer-grid map approach finds directly an optimal solution without any balancing parameters. This method, unfortunately, requires a fold-over free Poisson parameterization where in our case it is only needed for a very small set of triangles. our method also gives a balancing parameter which makes the layout computation more flexible. In contrast to IGM, our method is guaranteed to be spiral-free which on one hand allows less patch numbers but on the other hand can produce non-uniform dense layout (see Appendix). In Table 2 is an overall statistics of our algorithm. The timing does not include the frame field generation nor the Poisson parameterization. We consider these two steps as preprocessing and can be taken as input by the algorithm. In our implementation, solving $\mathbf{P}_1$ is in average one second making the computation of a new solution extremely fast.

Model	Singularities	Patches		
		Ours	DLM	IGM
Botijo	72	163	221	175
Fertility	48	137	98	117
RockerArm	30	64	115	74
Elk	52	85	86	62

Figure 11: *Our algorithm applied to benchmark models also taken as input by DLM and IGM.*

In Figure 12 are other models, where the frame fields are generated by our own implementation. The models are especially chosen to show reliability of the proposed approach even for large sized models. The `Genus-91` model contains 720 singularities and was artificially constructed to test the algorithm under high number of singularities. We did not add any sharp edge constraints for the `Casting` model. The resulting layouts are naturally produced by the fix value of $\alpha = 5$.

Per-Patch Parameterization There are several post processing algorithms which can be applied to the final layout. One attractive approach is the layout relaxation of Campen et al. [CK14b] consisting of realigning the frame fields to the arcs direction and relaxing the singularities to reduce distortions. Another approach is the patch parameterization of Tarini et al. [TPP*11] together with an iterative smoothing technique. Since the patches are already clipped against the geometry, a direct parameterization, using [Lip12] as done in [MPZ14], can also be applied if a seamless global parameterization is not of interest.

Figure 12: *Perfect matching quad layouts generated on the `Casting`, `Dragon`, `Bunny`, `Boudha`, `Hand` and `Genus-91` models.*

In our implementation, we assure each patch to be connected in parameter space and we map each of them to a reference rectangle using a simple bilinear interpolation. The side-length of each rectangle is determined by the patch stripes. This approach produces a seamless global parameterization with relatively small distortion. In the inset Figure is an example of a per-patch parameterization of the `Bimba` model.

Quad Mesh Optimization If a quadrilateral meshing of the surface is available, then the proposed method can be adapted using a vertex base data structure. Quad meshes are simpler to process since all separatrices are closed (no need for a stopping criterion) and they can be handled at a purely combinatorial level. In contrast to triangle meshes, the surface has a quad layout derived from the base complex of the quad mesh. Hence, $\mathbf{P}_1$ does not depends on the size of $\mathcal{G}$ and the computation is extremely fast. We compare against two

Model	|S|	|F| of $\mathcal{B}$	**Optimized**			RSD	Total
			Ours	SQD	GSO	angles (%)	time (s)
Botijo	74	4957	**164**	460	1034	27.27	1.14
Fertility	48	2271	**100**	253	526	28.37	0.78
Rockerarm	36	4524	**64**	98	178	28.36	0.26
Drill hole	26	1368	**61**	82	216	22.83	0.88
Fandisk	30	408	**66**	88	144	16.26	0.18
Beetle	56	259	**156**	-	-	15.20	0.07

Table 1: *Comparison to other quad mesh optimization algorithm with the same input models also taken by [TPP*11] and [BLK11]. $\mathcal{B}$ denotes the base complex of the quad mesh.*

state of the art quad mesh optimization algorithms, the Simple quad domain (SQD) [TPP*11] and the Global structure optimization (GSO) [BLK11] using the same input models. The quad layouts generated by our approach are illustrated in Figure 13. The statistics are given in Table 1.

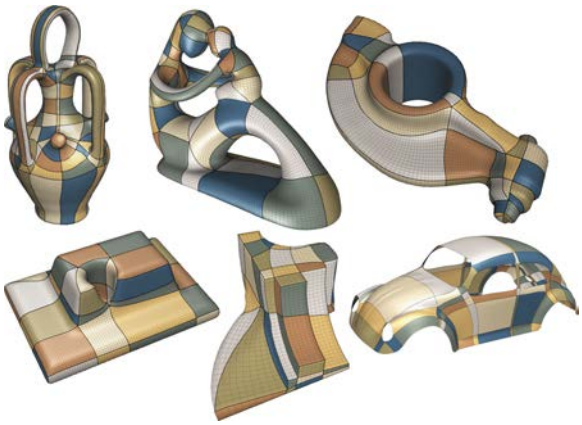


Figure 13: *Optimized quad layout on quad meshes produced by our algorithm.*

Limitations and Future Work A main limitation of our approach is the lack of a theoretical guarantee on the existence of a solution for the triangle mesh case. We construct $\mathcal{G}$ by stopping at spirals or at a prescribed maximum length but we did not prove that this stopping criterion guarantees the existence of a perfect matching in $\mathcal{G}$. In practice, we never met cases where the graph is not large enough to produce a solution with the proposed parameters.

Adapting our algorithm to work directly on streamline based layouting such as [MPZ14, RS14] is an attractive direction for future works. The main problem eventually lies on the caracterization of triangles and spirals since there is no metric on which we can reliably construct the graph.

Solving a MWPMPC is in general *NP*-hard as proven in [DPSW11]. Nevertheless, as we can see from Table 2, the running time is dominated by the graph construction (not the solving of $\mathbf{P}_1$) which can be intelligently parallelized.

Models	$\lvert T_{\mathcal{M}}\rvert$	$\lvert S\rvert$	**Graph**		Av. length	$\lvert P\rvert$	RSD	**Time in s**	
			$\lvert E_{\mathcal{G}}\rvert$	$\lvert I\rvert$	$1/N\sum \lVert l_e \rVert_f$		angles (%)	$\mathbf{P}_1$	Layout
Torso	74628	28	528	7892	162.10	42	17.71	0.12	4.61
Sofa	29020	14	229	2162	68.23	24	4.90	0.06	1.73
Fertility	27954	48	1118	18831	80.33	137	15.84	1.09	2.54
Elk	18308	52	1023	14150	95.04	85	15.21	0.46	1.91
Botijo	29994	74	1736	32192	204.04	163	17.76	0.64	5.77
RockerArm	70318	30	609	9441	127.49	64	18.65	0.15	5.13
Joint	1784	24	255	6085	55.96	79	7.32	0.03	0.61
Bumpy sphere	102996	150	4194	122991	184.82	480	17.78	2.71	63.74
Boudha	198736	70	1804	48888	161.72	255	17.45	1.93	55.51
Genus-91	28800	720	15903	151430	86.46	1500	6.29	3.85	31.22
Casting	66914	100	2683	83489	87.21	235	18.92	1.89	14.59
Hand	99999	38	763	20297	84.36	117	18.26	1.03	14.64
Bunny	69666	60	1347	23939	119.52	113	14.95	0.43	8.48
Dragon	102400	44	1006	22412	283.81	105	14.15	0.37	23.30
Bimba	149524	76	1944	64551	141.27	231	20.37	1.95	59.08

Table 2: *Statistics of the proposed method applied to several benchmark models. $\lvert S\rvert$ is the number of singularities, $\lvert P\rvert$ the number of patches and RSD the patch angles relative standard deviation measured in parameter space. $\mathbf{P}_1$ is the time taken to solve the constrained minimum weight perfect matching. Layout denotes the time needed to compute a solution from a given Poisson parameterization (without per patch colorings).*

8. Conclusion

We have introduced a novel algorithm to generate pure quadrilateral patch layouts on manifold surfaces. The new

approach reduces the quad layout problem to a constrained minimum weight perfect matching problem, which is classical in the field of graph theory. The resulting layouts are coarse and still align with geometric features. Our algorithm takes advantage of a global parameterization of the surface which makes the computation fast and the implementation robust. We expect that the presented concept will give rise to further research for patch layout generation and parameterization.

Acknowledgments This work has been supported by the Berlin Mathematical School. We thank David Bommes, AIM@SHAPE, Stanford 3D scanning repository for providing several datasets and Marco Tarini for providing his graph simplification program. Finally, we are very grateful for the helpful comments of the anonymous reviewers.

References

[BCE*13] BOMMES D., CAMPEN M., EBKE H.-C., ALLIEZ P., KOBBELT L.: Integer-grid maps for reliable quad meshing. *ACM Trans. Graph. 32*, 4 (July 2013), 98:1–98:12. 1, 2, 3, 7, 8

[BLK11] BOMMES D., LEMPFER T., KOBBELT L.: Global structure optimization of quadrilateral meshes. *Computer Graphics Forum 30*, 2 (2011), 375–384. 3, 5, 9

[BLP*12] BOMMES D., LÉVY B., PIETRONI N., PUPPO E., A C. S., TARINI M., ZORIN D.: State of the art in quad meshing. *Eurographics STARS* (2012). 2

[BMRJ04] BOIER-MARTIN I., RUSHMEIER H., JIN J.: Parameterization of triangle meshes over quadrilateral domains. *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing* (2004), 193–203. 3

[BZK09] BOMMES D., ZIMMER H., KOBBELT L.: Mixed-integer quadrangulation. *ACM Trans. Graph. 28*, 3 (July 2009), 77:1–77:10. 2, 3, 6, 7

[CBK12] CAMPEN M., BOMMES D., KOBBELT L.: Dual loops meshing: Quality quad layouts on manifolds. *ACM Trans. Graph. 31*, 4 (July 2012), 110:1–110:11. 1, 2, 3, 7, 8

[CK14a] CAMPEN M., KOBBELT L.: Dual strip weaving: Interactive design of quad layouts using elastica strips. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 183:1–183:10. 1

[CK14b] CAMPEN M., KOBBELT L.: Quad layout embedding via aligned parameterization. *Computer Graphics Forum 33* (2014), 69–81. 3, 8

[DBG*06] DONG S., BREMER P.-T., GARLAND M., PASCUCCI V., HART J. C.: Spectral surface quadrangulation. *ACM Trans. Graph. 25*, 3 (July 2006), 1057–1066. 3

[DPSW11] DARMANN A., PFERSCHY U., SCHAUER J., WOEGINGER G. J.: Paths, trees and matchings under disjunctive constraints. *Discrete Applied Mathematics 159*, 16 (2011), 1726 – 1735. 8th Cologne/Twente Workshop on Graphs and Combinatorial Optimization (CTW 2009). 4, 9

[DSC09] DANIELS J., SILVA C. T., COHEN E.: Semi-regular quadrilateral-only remeshing from simplified base domains. *Comput. Graph. Forum 28*, 5 (2009), 1427–1435. 3

[EBCK13] EBKE H.-C., BOMMES D., CAMPEN M., KOBBELT L.: Qex: Robust quad mesh extraction. *ACM Trans. Graph. 32*, 6 (Nov. 2013), 168:1–168:10. 3

[ECBK14] EBKE H.-C., CAMPEN M., BOMMES D., KOBBELT L.: Level-of-detail quad meshing. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 184:1–184:11. 2

[Eck96] ECK M.: Automatic reconstruction of b-spline surfaces of arbitrary topological type. *In Proc. SIGGRAPH'96* (1996), 325–334. 2

[EGKT08] EPPSTEIN D., GOODRICH M. T., KIM E., TAMSTORF R.: Motorcycle graphs: Canonical quad mesh partitioning. *Proceedings of the Symposium on Geometry Processing* (2008), 1477–1486. 3

[KNP07] KÄLBERER F., NIESER M., POLTHIER K.: Quadcover - surface parameterization using branched coverings. *Comput. Graph. Forum 26*, 3 (2007), 375–384. 2, 3

[Lip12] LIPMAN Y.: Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph. 31*, 4 (July 2012), 108:1–108:13. 2, 3, 8

[LP84] LEE D. T., PREPARATA F. P.: Euclidean shortest paths in the presence of rectilinear barriers. *Networks 14* (1984), 393–410. 4

[MPKZ10] MYLES A., PIETRONI N., KOVACS D., ZORIN D.: Feature-aligned t-meshes. *ACM Trans. Graph. 29*, 4 (2010), 1–11. 3

[MPZ14] MYLES A., PIETRONI N., ZORIN D.: Robust field-aligned global parametrization. *ACM Trans. Graph. 33*, 4 (July 2014), XX:1–XX:14. 3, 8, 9

[Nie12] NIESER M.: *Parameterization and Tiling of Polyhedral Surfaces*. PhD thesis, Freie Universität Berlin, 2012. 7

[OZP13] ÖNCAN T., ZHANG R., PUNNEN A. P.: The minimum cost perfect matching problem with conflict pair constraints. *Computers and Operations Research 40*, 4 (2013), 920 – 930. 4

[PL86] PLUMMER D., LOVÁSZ L.: *Matching Theory*. North-Holland Mathematics Studies. Elsevier Science, 1986. 4

[PPTSH14] PANOZZO D., PUPPO E., TARINI M., SORKINE-HORNUNG O.: Frame fields: Anisotropic and non-orthogonal cross fields. *ACM Trans. Graph. 33*, 4 (July 2014), 134:1–134:11. 2, 3

[RLL*06] RAY N., LI W. C., LÉVY B., SHEFFER A., ALLIEZ P.: Periodic global parameterization. *ACM Trans. Graph. 25*, 4 (Oct. 2006), 1460–1485. 2

[RLS*12] REMACLE J.-F., LAMBRECHTS J., SENY B., MARCHANDISE E., JOHNEN A., GEUZAINET C.: Blossom-quad: A non-uniform quadrilateral mesh generator using a minimum-cost perfect-matching algorithm. *Int. J. Numer. Meth. Engng. 89*, 9 (Mar. 2012), 1102–1119. 4

[RS14] RAY N., SOKOLOV D.: Robust polylines tracing for n-symmetry direction field on triangulated surfaces. *ACM Trans. Graph. 33*, 3 (June 2014), 30:1–30:11. 3, 9

[RVLL08] RAY N., VALLET B., LI W. C., LÉVY B.: N-symmetry direction field design. *ACM Trans. Graph. 27*, 2 (May 2008), 10:1–10:13. 3

[TPP*11] TARINI M., PUPPO E., PANOZZO D., PIETRONI N., CIGNONI P.: Simple quad domains for field aligned mesh parametrization. *ACM Transactions on Graphics, Proceedings of SIGGRAPH Asia 2011 30*, 6 (2011). 2, 3, 5, 8, 9

[TPSHSH13] TAKAYAMA K., PANOZZO D., SORKINE-HORNUNG A., SORKINE-HORNUNG O.: Sketch-based generation and editing of quad meshes. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH) 32*, 4 (2013), 97:1–97:8. 1