Optimal Base Complexes for Quadrilateral Meshes

Faniry H. Razafindrazaka*
Konrad Polthier†

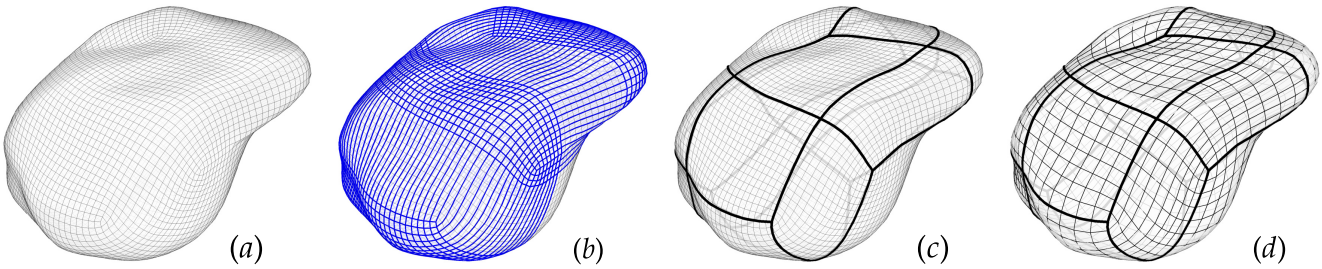


Figure 1: *(a) Semi-regular quad mesh $\mathcal{M}$, (b) base complex $\mathcal{B}$ obtained by removing all parameter loops in $\mathcal{M}$, (c) optimized base complex proposed by this paper and (d) quadrilateral remeshing of $\mathcal{M}$ along the new base complex with 9×9-grid per patch.*

Abstract

In this paper we give an explicit algorithm to optimize the global structure of quadrilateral meshes i.e base complexes, using a graph perfect matching. The approach consists of constructing a special graph over the singularity set of the mesh and finding all quadrilateral based complex subgraphs of that graph. We show by construction that there is always an optimal base complex to a given quadrilateral mesh relative to coarseness versus geometry awareness. Local structures of the mesh induce extra constraints which have been previously ignored but can give a completely different layout. These are diagonal, multiple and close to zero length edges. We give an efficient solution to solve these problems and improve the computation speed. Generally all base complex optimization schemes are bounded by the topology of the singularities, we explore the space of layouts encoded in the graph to identify removable singularities of the mesh while simultaneously optimize the base complex.

This is the author's version. The final version is published at CAGD elsevier (GMP2017).

1 Introduction

Quadrilateral surfaces or quad meshes are fundamental in computer graphics and geometry processing. Their metric property such as low distortion and their global structure such as the number of charts are the perfect domains for many applications such as spline fitting, subdivision surfaces, hierarchy in finite element analysis and computer animation. While several algorithms to convert triangle meshes into highly regular quad meshes have been proposed, few works focus on the global structure of the generated quad meshes. Most of the time these structures are bounded by the combinatorics of the singularities. A small change in the singularity layout induces a global effect on the resulting quad tiling. Changing the connectivity e.g. merging two singularities or flipping one edge of the mesh is very complicated to track on these class of meshes.

This paper concerns the global structure optimization of semi-regular quad meshes generated mainly by frame field based parameterization such as [Kälberer et al. 2007; Bommes et al. 2009; Bommes et al. 2013]. These classes of parameterizations are powerful to achieve very low distortion meshes at the cost of increasing number of quads. They are not very suitable for coarse quad layout generation due to their greedy nature. The generated base complexes have too many patches which are mainly caused by the so called *near misses*, a parameter line misses to connect to a near by singularity. Nevertheless they are very good candidate as a starting layout for our optimization.

So far, there have been handful works focusing on the global structure of a given quadrilateral surface. The work of Bommes et al. [Bommes et al. 2011] analyzes the spirals present in the base complex. They are first detected and then eliminated according to a local editing state machine. This approach produces relatively coarse base complexes but not all spirals could be removed by the local operations. Tarini et al. [Tarini et al. 2011] proposes to disentangle the graph of separatrices in order to find another graph with less separatrix energy. A tree of possible configuration is then built until a local minimum state is found. This approach relies heavily on the mesh combinatorics and the search of valid configuration can be stuck in local minima. Zhang et al. [Zhang et al. 2016] uses a similar approach by identifying possible singularity connection using rectangular area based filtering. Valid quad layout is then obtained by iteratively adding a quad layout constraints in a binary program. Recently, a polycube based simplification [Cherchi et al. 2016] has been introduced. While not directly related to our focus, they succeed to optimize a given polycube base complex to a simpler base complex. Their method unfortunately can only handle meshes with polycube domains.

*e-mail:faniry.razafindrazaka@charite.de
†konrad.polthier@fu-berlin.de

Other approaches [Peng et al. 2011; Verma and Suresh 2015] reduce the number of singularities present in the mesh by local mesh editing or quad templates embedding but ignore the global structure of the resulting quad mesh. Local merging of a pair of singularities is most of the time related to a third singularity. Quad mesh simplification [Tarini et al. 2010] can also be used to generate coarse quad layout. Uncontrollable singularity behaviour and distorted patches make the search of good layout non-optimal. Several quad layout generation algorithms, even though not quadrilateral mesh based, use greedy loop selection [Campen et al. 2012], quadratic optimization [Bommes et al. 2013] or frame field aligned geodesic [Razafindrazaka et al. 2015; Pietroni et al. 2016].

Our approach is similar to [Razafindrazaka et al. 2015; Pietroni et al. 2016] which uses matching theory to find an optimal layout. These previous works focuse mainly on triangle meshes while several problems can occur on quadrilateral meshes that could be problematic in the layout optimality. More a perfect matching always exists on quad meshes such that no T-junctions nor stopping criteria are required. In contrast to Zhang et al. [Zhang et al. 2016], our matching formulation is more versatile, works on meshes with boundary and optimizes for a solution in one go.

In a nutshell we give an algorithm which simplifies a given quad layout and guarantees a pure quad and optimality of the final layout. Our construction is not limited by the topology of the singularity, we propose a simple approach to allow some singularities to be merged while optimizing the global structure of the quadrilateral mesh using a tree of subgraphs. We give a detailed implementation of the construction allowing the generated mesh to be used in several applications scenario.

1.1 Definitions and Notations

On a quad mesh, a *regular* vertex has valence four in the interior and valence two on the boundary. It is called *irregular* or *singular* otherwise. We denote by $\mathcal{S}$ the set of all singularities of $\mathcal{M}$. The *valence* of a vertex is the number of faces adjacent to that vertex. In the dual mesh, a regular (or singular) vertex induces a *regular* quadrilateral patch (or *singular* n-gon with $n \neq 4$) region.

1.1.1 Base Complexes and Quad Layouts

At a valence n singularity there are n *ports* which can be geometrically represented as the edges incident to that vertex. A *parameter line* is a sequence of mesh edges connecting two singularities. A *separatrix* is a curve (not necessarily a parameter line) connecting two singularities. Hence, parameter lines are particular types of separatrices. The *base complex* $\mathcal{B}$ of a quad mesh is the set of all parameter lines of the mesh inducing a quadrilateral patch layout of the surface (figure 1(b)). A *quad layout* is a set of separatrices which after reparameterization gives a base complex of the surface. A quad layout is more general than a base complex since patch edges do not need to be aligned to mesh edges (figure 1(c)). It is this observation which will guide us in our quest of good base complexes.

1.1.2 Halfedge Representation

We use the halfedge datastructure to represent our mesh. In this representation edges are split into two directed and opposite halfedges. A halfedge h will maintain a pointer to its `head` vertex v, opposite halfedge h`.opposite`, next halfedge h`.next` and a face defined on its left. A quadrilateral face can be for example represented as four successive halfedges having the same left face. To avoid exception, boundary halfedges will have an opposite halfedge marked as boundary.

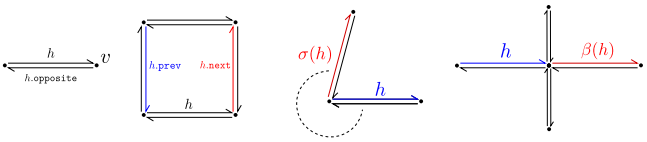


Figure 2: *Halfedge datastructure, operator σ and β.*

1.1.3 Graph and Matching

A *weighted graph* $G = (V_G, E_G)$ is a set of vertices V_G connected by edges in $e \in E_G$ with an associated weight function $w_e : E_G \rightarrow \mathbb{R}_+$. A *subgraph* $H = (V_H, E_H)$ of G is a graph with $V_H \subseteq V_G$ and $E_H \subseteq E_G$. A *matching* of G is a subgraph such that no two edges share a vertex. A *maximum matching* is a matching with a maximal number of edges. If all vertices of G is in the matching subgraph, then the matching is called *perfect*.

1.1.4 Port Splitting

At each singularity s, a halfedge h_s emanating from s defines a *port*. They are related by a permutation σ defined as follow: given an halfedge h_s whose opposite halfedge point to s

$$\sigma(h_s) = h_s\texttt{.prev.opposite}.$$

In other words, σ returns the next halfedges emanating from s in a counter-clockwise order modulo $\texttt{Valence}(s)$. In many quad mesh optimizations, the set of tuples (σ, h_s) form a *rotation system* encoding the combinatorial embedding of a graph layout on the surface.

We define a graph G with these ports as nodes and arc of separatrices as edges,

$$V_G = \{h_s, \text{ s.t } s \in \mathcal{S} \text{ and } s = h_s\texttt{.opposite.head}\}.$$

Our main goal is to explore subgraphs H of G which forms a quad layout of $\mathcal{M}$ such that $E_H \subset E_G$.

1.1.5 Notations

We define an edge of G by $e_{st} = (h_s, \sigma^{\text{sign}_{st}}(h_t))$. We call h_s the *base port* and $\sigma^{\text{sign}_{st}}(h_t)$ the *candidate port* where h_s is connected to in V_G. The operator $\text{sign}_{st} = \pm 1$ decides the orientation of σ in the port association. Each edge is decomposed into two parameter curves $\texttt{Base}(e_{st})$ and $\texttt{Offset}(e_{st})$. $\texttt{Base}(e_{st})$ (resp. $\texttt{Offset}(e_{st})$) is the parameter curve from h_s (resp. h_t) to its intersection point with the parameter curve from h_t (resp. h_s). In other words, they are the two right sides of a parametric right triangle patch on $\mathcal{M}$ as in figure 3.

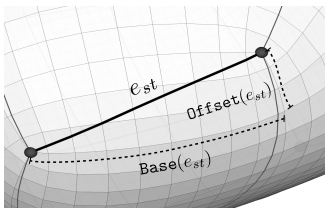


Figure 3: *An edge of G decomposes into two parameter curves $\texttt{Base}(e_{st})$ and $\texttt{Offset}(e_{st})$.*

2 Graph Construction

The construction of G is performed in a motorcycle style graph but a ray only stops when a singularity is reached. The propagation is defined by an operator β acting on a port h_s such that

$$\beta(h_s) = h_s.\texttt{next.opposite.next}$$

It follows immediately that for each $s \in \mathcal{S}$ there exists an integer n and $t \in \mathcal{S}$ such that $\beta^n(h_s) = \overbrace{\beta \circ \cdots \circ \beta}^{n}(h_s) = h_t.\texttt{opposite}$. For separatrices ending at a boundary vertex, we need to consider that vertex as a singularity and define properly the notion of ports with the correct permutation.

To be able to evaluate distances and ratios on the mesh, we assign a unit length to each edge. The length of a parameter curve on $\mathcal{M}$ from a port h_s to an halfedge $\beta^n(h_s)$ is for example

$$|\beta^n(h_s)| = n$$

i.e. the number of parameter edges visited from the port h_s to $\beta^n(h_s)$.

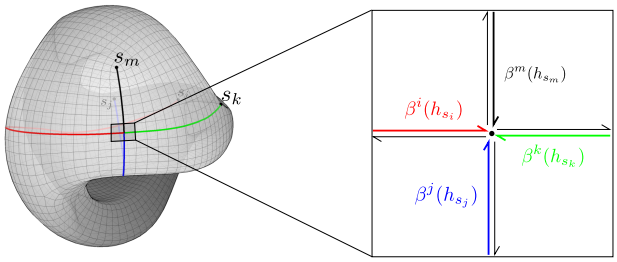


Figure 4: *Halfedge local ordering at a regular vertex of the base complex $\mathcal{B}$.*

We summarize the graph construction in Algorithm 1. Lines (2)–(6) are initialization steps which are necessary for fast evaluation and construction of the graph edges. Line (4) assigns per vertex four integers which are the number of steps used to reach the halfedge pointing to the vertex v in a counter-clockwise order (figure 4). Line (6) is a maximum deviation initialization associated to each port. The importance and use of these values are detailed in Section 3 to reduce the graph and to have a robust quad topology constraints. The main part of the algorithm is executed in Line (7)–(26). The construction starts at a port h_s then follows a parameter line until a singularity is reached. During the propagation, candidate ports are collected according to a ratio test and new edges are added to G. The main idea of the ratio test in Line (15) and (20) is that if two singularities lie on the hypotenuse of a right triangular grid, then they are likely to be connected. Line (26) makes sure that the separatrix part of the base complex is also added to G making sure that as a graph $\mathcal{B} \subseteq G$ and hence a quad layout subgraph always exists.

3 Filtering

The graph G is a combinatorial representation of the port connections. In its geometric realization several filtering have to be introduce to reduce the size of G and to remove arc edges which may induce degenerate patches in the subgraph.

Algorithm 1 Construction of the graph G

1: **procedure** GRAPHCONSTRUCT
2: **for** $v \in \mathcal{B}$ and $v \notin \mathcal{S}$ **do**
3: **if** Valence$(v) == 4$ **then**
4: $I_v = \{i, j, k, m\}$ s.t $v = \beta^p(h_{s_p})$.head for all $p \in I_v$ ▷ Figure 4
5: **for** $h_s \in V_G$ **do**
6: $L_{h_s} = 1, R_{h_s} = 1$
7: **for** $h_s \in V_G$ **do**
8: $n = 1$
9: **while** $v = \beta^n(h_s)$.head $\notin \mathcal{S}$ **do**
10: **if** $v \in \mathcal{B}$ and Valence$(v) == 4$ **then**
11: Assume $n = i \in I_v$ and $s_i = s$
12: $L = m/i$
13: $R = j/i$
14: Base(e_{ss_m}) = Base(e_{ss_j}) = $\{\beta^\alpha(h_s)\}_{\alpha=0...n}$
15: **if** $L \leq L_{h_s}$ **then**
16: Offset$(e_{ss_m}) = \{\beta^\alpha(h_s)\}_{\alpha=0...m}$
17: $\text{sign}_{ss_m} = -1$
18: ADDEDGE(e_{ss_m})
19: $L_{h_s} = L$
20: **if** $R \leq R_{h_s}$ **then**
21: Offset$(e_{ss_j}) = \{\beta^\alpha(h_s)\}_{\alpha=0...j}$
22: $\text{sign}_{ss_j} = +1$
23: ADDEDGE(e_{ss_j})
24: $R_{h_s} = R$
25: $n = n + 1$
26: $E_G = E_G \cup E_{\mathcal{B}}$ ▷ Assure existence of quad layout subgraph

3.1 Geodesic and Singularity

The filtering strategy already included in Algoritm 1 Lines (15),(20) makes sure that all edges in G are a straight line drawn in the grid induced from the quad mesh. This condition is called in [Razafindrazaka et al. 2015] a *singularity-free* condition which enables the predicate ADDEDGE to draw an arc on $\mathcal{M}$ connecting for example the port h_s and $\sigma^{-1}(h_{s_m})$ by using a simple scan-conversion algorithm. Singularity-freeness is not a necessary condition for G but it enables the assignment of a local uv alignment to each edge. Notice that the port h_s is not directly connected to h_{s_m} because naturally the hypotenuse of the triangle rectangle having length $|\beta^i(h_s)|$ and width $|\beta^m(h_{s_m})|$ aligns best to $\sigma^{-1}(h_{s_m})$ at s_m.

3.2 Helix Filtering

The quality of the base complex is closely related to the number of helical configurations present in the mesh. They are necessary to achieve very uniform quadrilateral meshes but are problematic in our search of coarse layouts. Helices introduce multiple edges in the graph and hence increase the size of G with unnecessary edges. Since we aim for coarse base complexes, the arc-length of edges in G should be preferably short. Hence we assign to ADDEDGE a check if an edge is already in G, determine its base length and replace it if the new constructed edge has less base length. We summarize the procedure ADDEDGE in Algorithm 2. The input of the algorithm is an edge e_{st} which is going to be added to G. Line 2 checks if the edge is not yet in E_G then add it accordingly. Else it is already in E_G denoted by e_{st}^{prev} which shares the same base port and candidate port as e_{st} but with different lengths. Line 6 checks if the base length of the new edge is smaller than the previous edge. If it is the case, then the old edge is replaced by the new one. This check makes sure that only helices with the largest pitch are stored. Hence two vertices of G are at most connected by two edges of different lengths: the shortest (in helical configurations) and the longest (separtrix of the base complex) among all possible connections.

In [Razafindrazaka et al. 2015] the helices are used as a stopping criterion. A user defined threshold is then necessary to filtered the type of helices. Notice that in the case of quadrilateral mesh such

Algorithm 2 Adding an edge to G with helix filtering

```
1: procedure AddEdge(e_st)
2:     if e_st ∉ E_G then
3:         E_G = E_G ∪ {e_st}
4:     else
5:         Let e_st^prev ∈ G          ▷ Same endports as e_st but with different lengths
6:         if |Base(e_st)| < |Base(e_st^prev)| then
7:             e_st^prev ← e_st       ▷ Use edge with less base length
```

a threshold is not needed and hence, the implementation is fairly simple.

4 Constrained Minimum Weight Perfect Matching

4.1 Minimum Weight Perfect Matching with Disjunctive Constraints

The problem of finding a layout subgraph of $\mathcal{M}$ is equivalent to finding a minimum weight perfect matching under disjunctive constraints in G [Öncan et al. 2013; Razafindrazaka et al. 2015]. The perfect matching assures that per port there is exactly one adjacent separatrices. The disjunctive constraints are topological constraints which guarantee that the resulting patches are quads without degeneration. The binary program $\mathbf{P}_1$ solving the matching problem is formulated as follow

$$\text{minimize} \quad \sum_{e \in E_G} w_e x_e \tag{1}$$

$$\text{s.t.} \quad \sum_{e \in \text{Adj}(v)} x_e = 1, \quad \text{for all not virtual } \; v \in V_G \tag{2}$$

$$x_{e_1} + x_{e_2} \leq 1, \quad \text{for all } \; (e_1, e_2) \in C_{\text{diag}} \cup C_{\text{disj}} \tag{3}$$

$$x_e \in \{0, 1\}, \quad \text{for all } \; e \in E_G \tag{4}$$

where Adj(v) is the set of edges sharing a vertex $v \in V_G$. The objective functional minimizing the total edge weight is given in (1). The weight of an edge e_{st} is simply the linear combination of its base length and height length i.e, $w_e = |\texttt{Base}(e_{st})| + \alpha|\texttt{Length}(e_{st})|$. The parameter α controls generally the coarseness against geometric feature alignment. Virtual vertices are boundary vertices where a tracing line is stopping as defined in [Razafindrazaka et al. 2015]. The inequality in (3) is a disjunctive constraints disallowing two particular edges in G to appear simultaneously in the solution.

4.2 Disjunctive Edges C_{disj}

Two edges are in conflict or disjunctive if their presence in the perfect matching subgraph of G induces a non quad patch in $\mathcal{M}$. These are exactly two edges which are locally aligned to the same parameter line on $\mathcal{M}$. Instead of checking them one by one and iteratively solve a matching problem, we build a set of conflicting edges C_{disj} similar to [Razafindrazaka et al. 2015] but with a simpler construction. In this case, we do not need to store curves passing through triangles, the discrete metric of the quad enables to store them efficiently per vertex independent of the mesh resolution.

Given an edge $e_{st} \in E_G$, we would like to find all edges which are locally aligned to e_{st}. These are all edges whose base curves are locally parallel. More precisely let $v \in \texttt{Offset}(e_{st}) \cap \mathcal{B}$ then there exists two edges e_{px} and e_{ry} such that $v \in \texttt{Base}(e_{px}) \cap \texttt{Base}(e_{ry})$ (see figure 5 (a,b)). These can be eventually edges of

Algorithm 3 Disjunctive Edges

```
1: procedure Disjunctive(e_st)
2:     for v ∈ Offset(e_st) ∩ B do                    ▷ Interior vertices
3:         Take {e_px}_x, {e_ry}_y s.t v ∈ Base(e_px) ∩ Base(e_py)   ▷ Figure 5
4:         for all x do
5:             C_disj = C_disj ∪ {(e_st, e_px)}
6:         for all y do
7:             if sign_st ≠ sign_ry or |Offset(e_ry)| < |Offset(e_st)| − d(t, v) then
8:                 C_disj = C_disj ∪ {(e_st, e_ry)}
```

$\mathcal{B}$. All edges which are connected to h_p, have base length greater than $|\texttt{Base}(e_{px})|$, are locally aligned to e_{st}. The same as all edges which are connected to h_r, have base length greater than $|\texttt{Base}(e_{ry})|$, are locally aligned to e_{st} except those which do not intersect with e_{st} like the red edge drawn in figure 5(b). The disjunctive edge set is defined as follow

$$C_{\text{disj}} = \{x_e + x_f \leq 1 \text{ if } e \cap f \neq \varnothing \text{ and are locally aligned on } M\}$$

In algorithm 3 is a summary of the process. The metric d in line (7) is the integer metric on $\mathcal{M}$. Notice that in contrast to the algorithm 1 of [Razafindrazaka et al. 2015], no explicit intersections are made during the construction.

4.3 Diagonal Edges C_{diag}

In the graph construction, the ratios defined in algorithm 1 line (12) and (13) can be exactly one. Notice that, this case will generate two geometrically equal edges but different alignment on $\mathcal{M}$ which induces a degenerate patch (area equal to zero) in the final layout, figure 6(c). The degeneration is avoided by adding extra disjunctive constraints to the matching problem to prevent the appearance of both edges in the subgraph matching. The generated diagonal edges will always have less separatrix energy independent of the values of the parameter α. It is then beneficial to have them in the matching solution. However they are minimum in both base lengths and offset deviations, and reduce the number of patches of the initial base complex at the cost of high angle deviations which cannot be resolved even for large values of α.

A solution to this problem is for example to allow only far apart singularities to be diagonally connected. This is a parameter dependent since closeness is not well defined. A post validation could also be applied by identifying these configurations, removing the corresponding edges from the graph and recomputing a solution. In our implementation we did not do any of these optimizations. We keep diagonal edges to avoid missing some interesting layouts especially when the number of singularities increase as in figure 6. The additional conflict set C_{diag} is defined as follow,

$$C_{\text{diag}} = \{x_e + x_f \leq 1 \text{ if } e, f \in \text{Diag(Square) on } \mathcal{M}\}$$

where Diag(Square) is the diagonal of a square grid on M.

4.4 Graph Reduction

As proven empirically in [Darmann et al. 2011], the complexity of the binary program $\mathbf{P}_1$ depends weakly on the size of G but strongly on the size of the disjunctive constraints $C_{\text{disj}} \cup C_{\text{diag}}$. Our preferred solver ILOG-CPLEX does not succeed for example to find a feasible solution for disjunctive constraints exceeding three hundred thousand. This can happen even on low sized graph. In figure 9 are two examples (ARMADILLO,LUCY) of quad meshes where this issue appears. We suspect that this is caused by spirals present in the base complex where no other possible candidates can

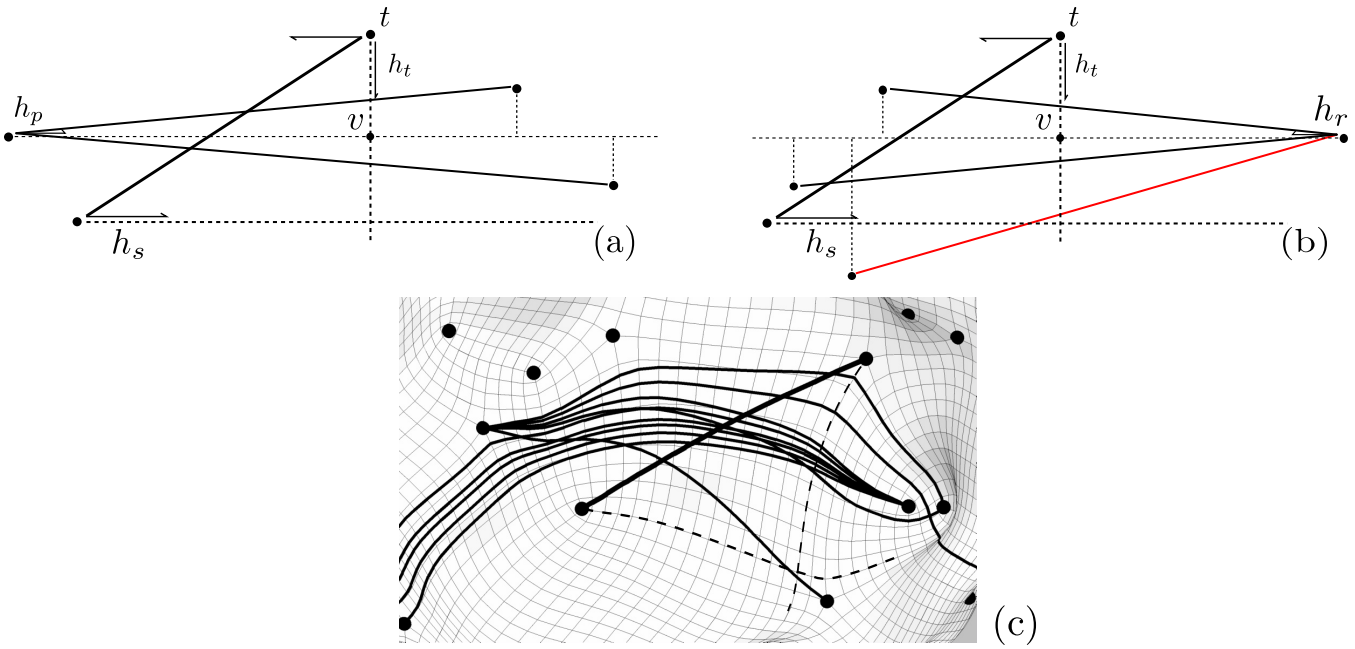


Figure 5: *Examples of disjunctive edges which are not allowed to appear simultaneously in the final layout.*

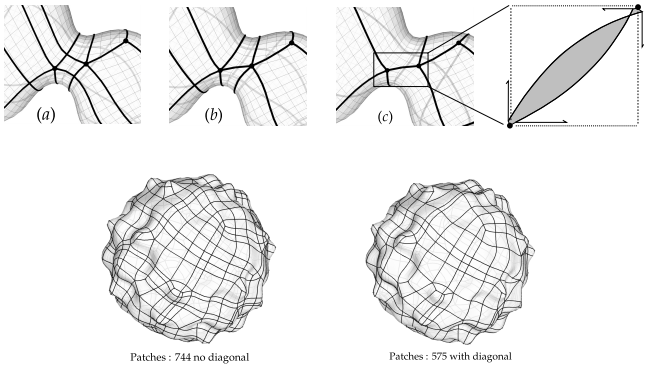


Figure 6: *(a) Without diagonal edges, (b) with constrained diagonal edges and (c) with non-constrained diagonal edges.*

be used and hence producing an exponential increase of conflicting edges.

Since we are aiming at a much coarser and low distortion quad layout, we can exploit the geometric property of the edges to reduce G. Our reduction strategy is to remove an edge of G if it is too close to another edge sharing a common vertex in terms of ratio. More precisely, consider two edges e_{st} and e_{sw} such that $|\texttt{Base}(e_{st})| < |\texttt{Base}(e_{sw})|$. Consider $\text{Ratio}_{st} = |\texttt{Offset}(e_{st})|/|\texttt{Base}(e_{st})|$ and $\texttt{Offset}(e_{sw})|/|\texttt{Base}(e_{sw})|$ such that

$$d = |\text{Ratio}_{st} - \text{Ratio}_{sw}| < \varepsilon$$

for a user specified ε. We then remove e_{sw} from the graph if it is not in $\mathcal{B}$ (figure 8 (b)). The motivation behind the reduction is that if e_{sw} is part of the final layout, then it will induce a very small sided patches since the singularity t is close by as illustrated in figure 8(a). Notice that the reduction does not affect very much the final layout geometry for coarse layout (see for example the PEGASO model in figure 9).

Theorem 1 *(Optimal Base Complex) Given a quadrilateral mesh M, an optimal base complex of M relative to the parameter α is a solution of* $\mathbf{P}_1$.

The existence of a solution is guaranteed by $\mathcal{B}$. Since no patch contains a singularity, C_{disj} guarantees a pure quadrilateral base complex. Degenerate patches i.e area equal to zero are removed by C_{diag} and patches with side lengths close to zero are removed by ε. In other words, the theorem fills the gap in all previous proposed solutions: no existence guarantee [Razafindrazaka et al. 2015] and the presence of T-junctions [Pietroni et al. 2016].

5 Results and Analysis

In our analysis, we took models already optimized by existing methods and run our algorithm to check if these can still be improved or not. We also check our method for robustness on models with several singularities. In all experimentation, we took the balance parameter $\alpha = 10$ as motivated by the sequences in figure 7. In general, small values of α are always suitable for coarse base domains. We choose $\varepsilon = 10^{-5} \times \text{Diam}(\mathcal{M})$ which we found reasonably good, mesh resolution independent and works well for a large variety of quad meshes. To solve the binary problem, we use ILOG-CPLEX (www.ibm.com).

5.1 Compared to Matching Based Approach

We compare with existing methods [Razafindrazaka et al. 2015; Pietroni et al. 2016] using matching formulation. The perfect matching quad layout proposed in [Razafindrazaka et al. 2015] works on seamless parameterization but unfortunately a theoretical guarantee which assures the existence of subgraph layout for their proposed graph construction is still missing. This is resolved in [Pietroni et al. 2016] by only looking for a maximum matching at the cost of T-junctions and eventually complicated zero-chains. Our approach is an optimization approach, has always a solution and contains no T-junctions. In figure 10 we took models generated by [Pietroni et al. 2016] as input and analyze the optimized layout

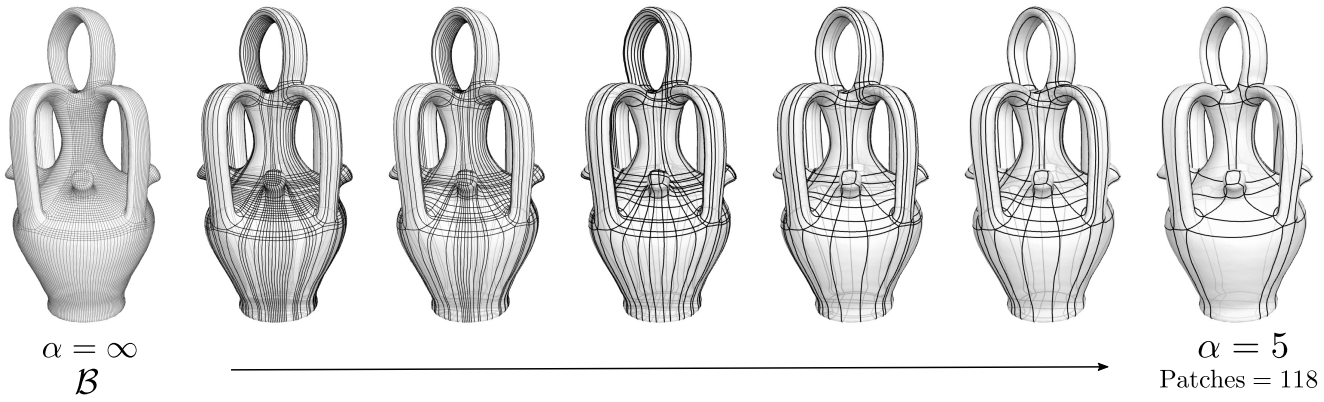


Figure 7: *Several values of α which correspond to a simplification like behaviour of the base complex $\mathcal{B}$.*

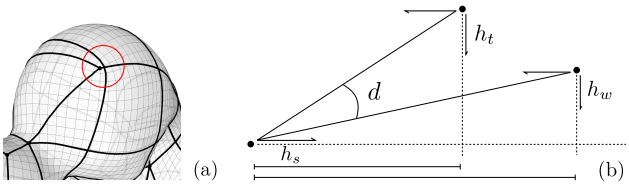
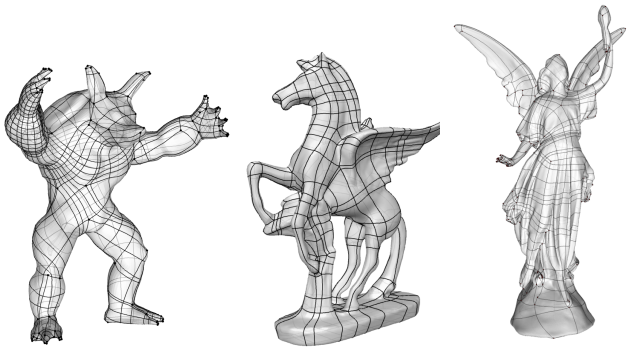


Figure 8: *Graph reduction by removing edges which might induce a thin strip on the final layout, using a control parameter ε.*



		$\lvert E_G \rvert$	$\lvert C_{disj} \rvert \cup \lvert C_{diag} \rvert$	CPLEX (in s)	Num. Patches
Armadillo	With Reduction	1959	27606	0.66	977
	Without Reduction	3059	736023	Fail	–
Pegaso	With Reduction	4165	187810	3.32	574
	Without Reduction	4245	196148	3.38	580
Lucy	With Reduction	2528	40961	1.03	940
	Without Reduction	9863	5772675	Fail	–

Figure 9: ARMADILLO *[Ebke et al. 2014],* PEGASO *and* LUCY *[Pietroni et al. 2016] the size of the conflict set is huge and the solver may fail to find a solution.*

generated by our method. We noticed that the number of patches is not so different although the T-junctions are gone. In this case, optimality is relative since their method aims for exact field alignment where we aim at coarse layout but still low distortion.

5.2 Compared to Zhang et al.

The approach of Zhang et al. [Zhang et al. 2016] is a particular case of the perfect matching approach. Their graph construction is based on using rectangular area filtering where we are using singularity-

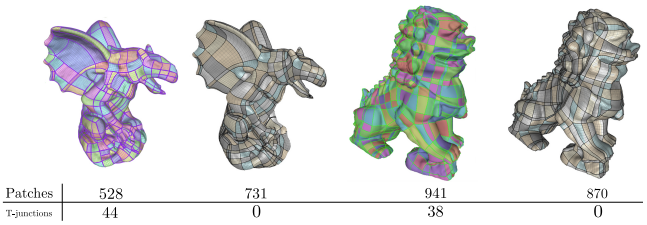


Figure 10: *Layout generated by [Pietroni et al. 2016] containing T-junctions (left), optimized using the proposed method (right) with no T-junctions.*

free triangular area, our space of layout is then larger. The use of area detection instead of separatrix tracing construction makes the construction of the conflict set difficult and hence the use a suboptimal iterative approach is not appropriate. In figure 11 is a comparison example on the ROCKERARM model. We can clearly see the effect of using rectangular patch detection where all separatrices tend to be short. In our triangular patch detection (on the right), some candidates port which are not found by the rectangles are identified and the layout , at least visually, exhibit a good balance between coarseness and geometric feature alignment.

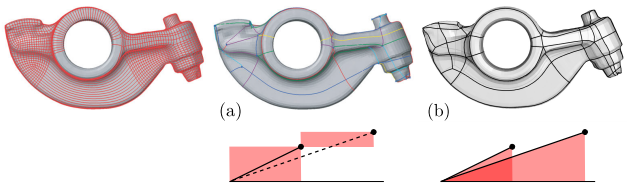


Figure 11: *(a) Approach proposed by Zhang et al. using rectangular area detection (b) our approach using triangle free singularity approach.*

5.3 Compared to Tarini et al.

Tarini et al. [Tarini et al. 2011] proposes a greedy approach to optimize the base complex of a given quadrilateral mesh. The key idea behind their algorithm is to modify $\mathcal{B}$ by finding possible connections inside the corridor of a given port (two parallel separatrices) and making sure that the resulting layout is all quads. This is in some aspect a possible way to construct G using corridor as a filtering condition for port connections similar to the rectangular area detection of [Zhang et al. 2016] with the drawback that the behavior

Name	$\|S\|$	$\|E_G\|$	$\|C_{disj}\|$	$\|C_{diag}\|$	Num. Patches
LION	229	2725	31851	50	870
GARGOYLE	178	2340	28489	84	731
FERTILITY	42	904	12856	20	132
BUMPY TORUS	108	2514	61216	40	285
ELEPHANT	110	2182	43773	52	306
HAND	40	693	16943	11	124
FANDISK	30	334	1798	10	103

Table 1: *Several statistics on the optimized patches*

of the corridors cannot be control during the optimization. Nevertheless they propose a greedy solver based on a tree of valid configurations to solve $\mathbf{P}_1$ without disjunctive constraints. The search of valid configurations may take several minutes due to local minima and optimal valid configuration cannot be reached.

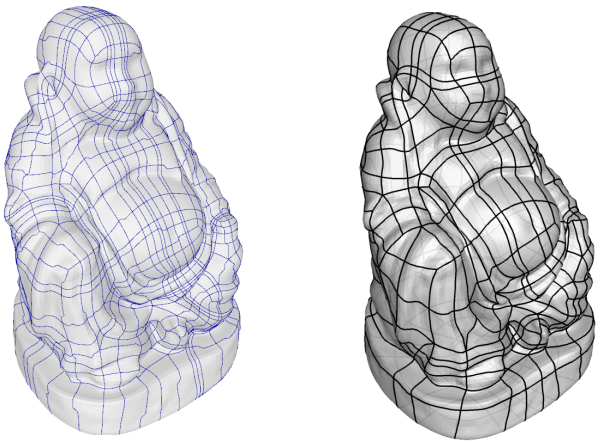


Figure 12: *Left: layout optimized by Tarini et al. with 1308 patches after 351 moves ($\sim$ 15 s). Right: our optimization with 455 patches ($\sim$ 0.6 s). The* BUDDHA *model was produced by [Bommes et al. 2013].*

On low resolution model i.e low number of singularities, the method of Tarini et al. succeeds to produce similar layout as ours. On high resolution model (figure 12) on the other hand their method struggle to get out of local minima even after several moves increasing significantly the number of patches.

In figure 13 are some more layouts optimized by our method where the quad meshes where generated by [Kälberer et al. 2007]. The statistics of the patches and the graphs are given in Table 1. The running time is relatively fast. The low resolution models are in average one seconds on a single core 3,5 Ghz CPU.

6 Extensions and Future Works

6.1 Singularity Reduction

In this section, we study subgraph matching H^k of G such that $|V_{H^0}| = |V_G|$ and $|V_{H^k}| < |V_G|$, for $k > 0$. The subgroups H^k's form quad layouts of $\mathcal{M}$ with some singularities removed. For the practical implementation and evaluation, we leave them as future works.

Merging singularities are very difficult on quad meshes which have been the subject of many research papers. They are global operation and uncontrollable behaviour can easily occur during the modifications. In [Peng et al. 2011] the merging of two singularities of valence 3 and 5 requires for example a third singularity and a smoothing is afterwards applied to reduce the distortion in the mesh. We call a pair of singularities *removable* if a subgraph matching quad layout of G exists without the pairs. We cannot remove one singularity since it leaves the graph with odd number of vertices where a perfect matching cannot exist. Without loss of generality we only focus on removable pairs of singularity $3-5$, other valences can be handled similarly.

We are not solving the problem of the choice of singularities to be removed. This is still an open problem.Given the singularities connectivity the quad mesh, we provide conditions to remove a pair of singularities and still provide a quad layout of the surface.

6.2 Necessary Condition

In order to have a perfect matching after removing a pair of singularities, all ports adjacent to both singularities should have at least two edges adjacent to them in G unless they are connected to each other. In figure 15 (a) is for example two singularities which cannot be removed since two endports (on the left of the valence 3 and on the right of the valence 5) have only one edge adjacent to them. This could be resolve if we modify the quad mesh and connect directly the loose ports to each other but this does not follow our general construction and conflicting edges are not anymore well defined. We leave that as part of independent research problem.

6.3 Sufficient Condition

Finding a sufficient condition for removable pairs of singularities cannot be done without checking if a perfect matching exists or not on the remaining set of port-singularities. A naive approach checks all pairs of $3-5$ singularities, removes them from the graph and solves $\mathbf{P}_1$. This could work but the resulting layout might not be pure quads. In the dual setting, a singularity of valence k induces a $k-$gonal patch. The main idea is then to make sure that the pairs $3-5$ are very close to each other which induces a triangular patch and a $5-$gon adjacent to each other in the dual (e.g figure 15 (b)). Removing the common edge of both patches generates a quadrilateral patch. We then introduce two new constraints as follow:

1. a candidate pair should lie on the diagonal of a rectangular patch on $\mathcal{M}$
2. separatrices separating a pair of singularities should not appear in the matching subgraph.

The first condition is a measure to check how close a pair is to each other. The second condition makes sure that the subgraph layout if it exists does not contain triangular patches nor 5-gon patches. Let us denote by (s_3, s_5) a pair to be removed. If (1) is not satisfied then there exists a singularity s_k such that all edges adjacent to one port of s_k separates (s_3, s_5), condition (2) will then remove all of these edges in the subgraph leaving one of the port of s_k open. A perfect matching cannot exist in that case in G. We summarize the removal of a singularity pair by the following proposition.

Proposition 1 *Given a pair of singularities (s_3, s_5) a subgraph layout of $\mathcal{M} - \{s_3, s_5\}$ is a solution of $\mathbf{P}_1$ with the following constraint: $x_e = 0$ for all edges e adjacent to the ports of s_3 and s_5, and for all edges separating s_3 and s_5.*

It can happen that no singularities of $\mathcal{M}$ can be removed even though conditions (1),(2) are satisfied because we are bound to the combinatorics of $\mathcal{M}$. Extension of the graph G could be done in

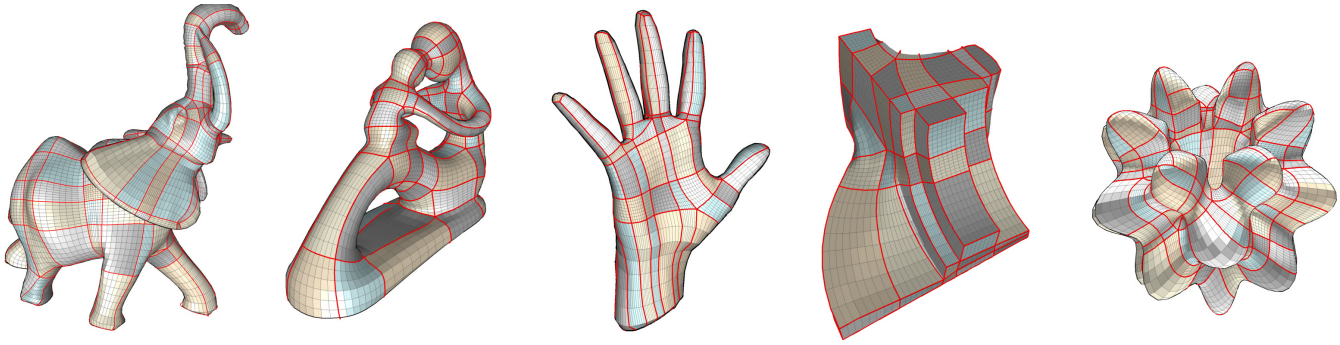


Figure 13: *Optimized layouts generated by the proposed approach. Input layouts (not shown) were generated by [Kälberer et al. 2007].*

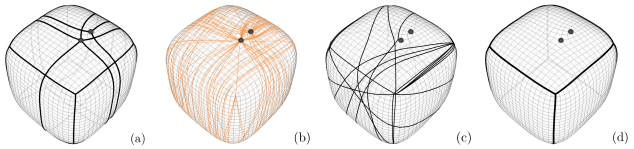


Figure 14: *(a) Subgraph layout from G (b) edges adjacent to a pair* $3-5$ *singularity (c) removing corresponding edges in G (d) a new layout subgraph in the reduced graph.*

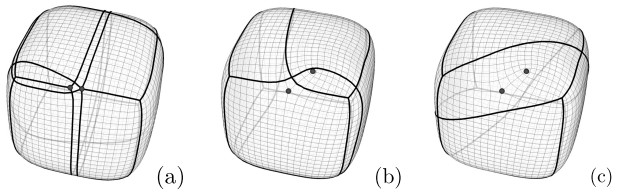


Figure 15: *(a) Not removable pair of singularities (b) removable pair inducing a non quad patches (c) removing edge separation induces a pure quad layout.*

this case but we will leave that as future work. As illustrated in figure 15 (c), removing singularities does not imply nice quality quad patches. Smoothing and more optimization needs to be done in that regards.

7 Outlook

We have given an algorithm to optimize the base complex of a given quadrilateral mesh into coarse and geometric feature aligned new base complex. The approach follows the recently introduced constrained perfect matching formulation making the construction fast, robust and relatively simple to implement. The results prove that the new approach performs very well, both in terms of speed and quality of the resulting layouts compared to current state of the art in base complex optimization. We also showed that manipulating singularities such as merging can be achieved in the graph formulation. This could be explored further in the future. The space of solution is small and bounded by the combinatorics of the quadrilateral mesh. A drift aware construction [Pietroni et al. 2016] combined with a modified metric could be a solution. Finding a consistent extension of the graph respecting the conflict set and allowing singularities to be merged with optimality guarantee will be our main focus for future works.

Acknowledgments

The authors would like to thank the Labor für Biofluidmechanic Berlin and the AGGeom Freie Universität Berlin for further assistance in the successful publication of this paper.

References

BOMMES, D., ZIMMER, H., AND KOBBELT, L. 2009. Mixed-integer quadrangulation. *ACM Trans. Graph. 28*, 3 (July), 77:1–77:10.

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

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

CAMPEN, M., BOMMES, D., AND KOBBELT, L. 2012. Dual loops meshing: Quality quad layouts on manifolds. *ACM Trans. Graph. 31*, 4 (July), 110:1–110:11.

CHERCHI, G., LIVESU, M., AND SCATENI, R. 2016. Polycube simplification for coarse layouts of surfaces and volumes. *Computer Graphics Forum 35*, 5, 11–20.

DARMANN, A., PFERSCHY, U., SCHAUER, J., AND WOEGINGER, G. J. 2011. Paths, trees and matchings under disjunctive constraints. *Discrete Applied Mathematics 159*, 16, 1726 – 1735.

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

KÄLBERER, F., NIESER, M., AND POLTHIER, K. 2007. Quadcover - surface parameterization using branched coverings. *Comput. Graph. Forum 26*, 3, 375–384.

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

PENG, C.-H., ZHANG, E., KOBAYASHI, Y., AND WONKA, P. 2011. Connectivity editing for quadrilateral meshes. *ACM Trans. Graph. 30*, 6 (Dec.), 141:1–141:12.

PIETRONI, N., PUPPO, E., MARCIAS, G., SCOPIGNO, R., AND CIGNONI, P. 2016. Tracing field-coherent quad layouts. *Computer Graphics Forum (special Issue of Pacific Graphics)*.

RAZAFINDRAZAKA, F. H., REITEBUCH, U., AND POLTHIER, K. 2015. Perfect matching quad layouts for manifold meshes. *Computer Graphics Forum (proceedings of EUROGRAPHICS Symposium on Geometry Processing) 34*, 5.

TARINI, M., PIETRONI, N., CIGNONI, P., PANOZZO, D., AND PUPPO, E. 2010. Practical quad mesh simplification. *Computer Graphics Forum (Special Issue of Eurographics 2010 Conference) 29*, 2, 407–418.

TARINI, M., PUPPO, E., PANOZZO, D., PIETRONI, N., AND CIGNONI, P. 2011. Simple quad domains for field aligned mesh parametrization. *ACM Transactions on Graphics, Proceedings of SIGGRAPH Asia 2011 30*, 6.

VERMA, C. S., AND SURESH, K. 2015. A robust combinatorial approach to reduce singularities in quadrilateral meshes. *Procedia Engineering 124*, 252 – 264.

ZHANG, S., ZHANG, H., AND YONG, J.-H. 2016. Automatic quad patch layout extraction for quadrilateral meshes. *Computer-Aided Design and Applications 13*, 3, 409–416.