Isotropic 2D Quadrangle Meshing with Size and Orientation Control

Bertrand Pellenard, Pierre Alliez, Jean-Marie Morvan

Isotropic 2D Quadrangle Meshing with Size and Orientation Control

Bertrand Pellenard[1], Pierre Alliez[1] and Jean-Marie Morvan[2,3]

[1] INRIA Sophia Antipolis - Méditerranée, France.
[2] Université Lyon 1/CNRS, Institut Camille Jordan, 43 blvd du 11 Novembre 1918, F-69622 Villeurbanne - Cedex, France. morvan@math.univ-lyon1.fr
[3] King Abdullah University of Science and Technology, GMSV Research Center, Bldg 1, Thuwal 23955-6900, Saudi Arabia.

Summary. We propose an approach for automatically generating isotropic 2D quadrangle meshes from arbitrary domains with a fine control over sizing and orientation of the elements. At the heart of our algorithm is an optimization procedure that, from a coarse initial tiling of the 2D domain, enforces each of the desirable mesh quality criteria (size, shape, orientation, degree, regularity) one at a time, in an order designed not to undo previous enhancements. Our experiments demonstrate how well our resulting quadrangle meshes conform to a wide range of input sizing and orientation fields.

1 Introduction

Quadrangle meshes are preferred over triangle meshes in a number of applications related to computer graphics, computer aided geometric design, computational engineering and reverse engineering. However, the automatic generation of isotropic quadrangle meshes for arbitrary 2D domains is still a scientific challenge due to the variety of requirements and quality criteria sought after.

The bare minimum requirement we impose that the input domain must be tiled with only *convex* quadrangles. We also wish to generate quadrangle meshes for which, *locally*, elements i) are well shaped in the sense of being close to squares, ii) are sized in accordance to an input sizing field, iii) are oriented in accordance to an input cross field (an orientation field modulo 90 degrees) and iv) have edges aligned to the domain boundary. A more *global* requirement practitioners often desire is that meshes should predominantly be composed of regular vertices, i.e., degree-4 vertices [4] inside the domain and (in general) degree-2 vertices on the domain boundary.

While these quality requirements are widely regarded as desirable, one key meshing difficulty is that they often conflict with one another: irregular vertices are often

[4] A degree-k vertex has k adjacent inside cells.

necessary for non-trivial sizing and cross fields, but they inevitably induce shape distortion and hence must be avoided whenever possible; a rapidly varying sizing field naturally induces both shape and orientation distortion; similarly, a rapidly varying cross field often results in both shape and size distortion. Another key challenge comes from the fact that some of the requirements or quality criteria, although locally defined, have global constraints—e.g., the number of edges on the domain boundary must be even, and the total index of irregular vertices must obey Gauss-Bonnet theorem [13].

1.1 Previous Work

The tension between local and global criteria may explain the variety of approaches proposed so far for the automatic generation of quadrangle meshes. Including methods devised to generate quadrangle surface tilings, the rich literature on this topic contains approaches which proceed by quadrangulation [9, 3], square packing [28], advancing front [22], conversion [8], decimation [18], Morse-Smale complexes [14, 33, 17], clustering [4, 20], local and global operators [23, 19], whisker weaving [32], medial axis [24], streamlining [1, 21] and parameterization [25, 29, 7].

Among these approaches, some favor the conformance of the final mesh to an input cross field either by construction [21], or by solving for the smoothest cross field given a set of orientation constraints [7]. Conformance to an input sizing field is either derived from the triangle mesh before conversion [8], or encoded in a density function before clustering [20]. Mesh regularity is controlled either explicitly by interactively placing a small number of irregular vertices before parameterization [29], or indirectly through a smooth cross field [7]. Regularity may be improved a posteriori through, e.g., grid-preserving operators so as to generate simple base complexes [6]. Strict local conformance to both sizing and cross fields is notoriously delicate for most approaches which involve a global variational formulation, and almost none of the fine-to-coarse approaches based on decimation [18, 23] leads to meshes that conform to both sizing and cross fields.

1.2 Contribution

In this paper we present a simple and practical isotropic quadrangle meshing algorithm for arbitrary 2D domains. We place a particular emphasis on having the resulting mesh conforming to both a sizing and a cross field. Our approach differs from previous work in that its methodology can be seen as antithetical to (e.g., Delaunay) refinement algorithms that locally refine a mesh one element at a time until all quality criteria are met. Rather than continuously fixing the element that most violate *any* of the requirements as in the Delaunay refinement strategy, we instead enhance the mesh by carefully *addressing one requirement at a time, in a processing order designed not to undo previous enhancements*. Table 1 describes how each step of the algorithm improves the various desirable quality criteria for flat 2D meshes.

	Size	*Shape*	*Orientation*	*Degree*	*Regularity*
Initialization (2.2)	●	•			
Relaxation (2.3)	◦	●	●		
Conforming relaxation (2.4)	◦	◦	◦	•	•
Local parameterizations (2.5)	◦	◦	◦	•	●
Barycentric subdivision (2.6)	◦	×	◦	●	◦
Smoothing (2.7)	◦	●	◦	◦	◦

Table 1. Each step of the algorithm improves different quality criteria. •: criterion is partially met; ●: criterion is met; ×: criterion may not be preserved; ◦: criterion is preserved.

2 Algorithm

The input of our algorithm is a closed domain, together with a sizing and a cross field. The sizing field is either user specified or automatically computed as a Lipschitz function from the local feature size estimate of the domain boundary [2]. The cross field is either specified by the user or automatically computed as the smoothest field that is tangential to the domain boundary [13]. Note that the mixed-integer approach [7] could also be used. The algorithm mainly proceeds by clustering and local parameterizations over a fine isotropic background triangle mesh $\mathcal{M}$ obtained by Delaunay refinement. We give pseudo-code of the algorithm below, while Figure 1 provides a visual depiction of its main steps.

Algorithm 1: Quadrangle meshing

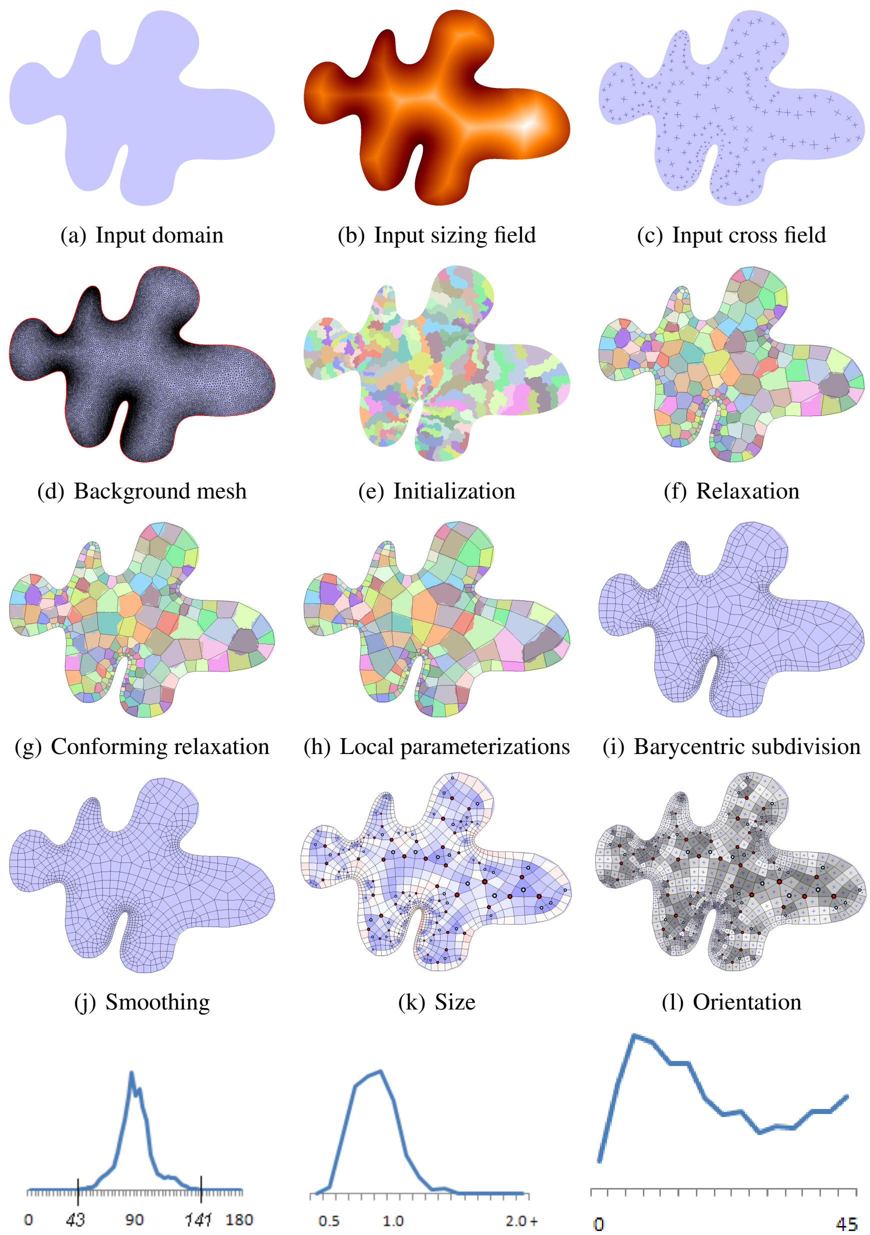
Input: $2D$ domain, sizing field, cross field
begin
1. Initialization (**2.2**)
2. Relaxation (**2.3**)
3. Conforming relaxation (**2.4**)
4. Local parameterizations (**2.5**)
5. Barycentric subdivision (**2.6**)
6. Smoothing (**2.7**)

Output: Quadrangle mesh

2.1 Preliminaries

The main steps of the algorithm (from initialization to local parameterizations through relaxations) act on the background mesh $\mathcal{M}$. In the reminder of this paper, we will make heavy use of the following terms (see Figure 2):

- **Tile.** A tile is a union of triangles of $\mathcal{M}$ defining a subdomain homeomorphic to a disk.
- **Meta-vertex.** A meta-vertex is a vertex of $\mathcal{M}$ which is either incident to 3 or more tiles inside the domain, or incident to 2 or more tiles on the domain boundary, or incident to a single tile when the vertex is tagged as a corner boundary vertex.



(a) Input domain (b) Input sizing field (c) Input cross field

(d) Background mesh (e) Initialization (f) Relaxation

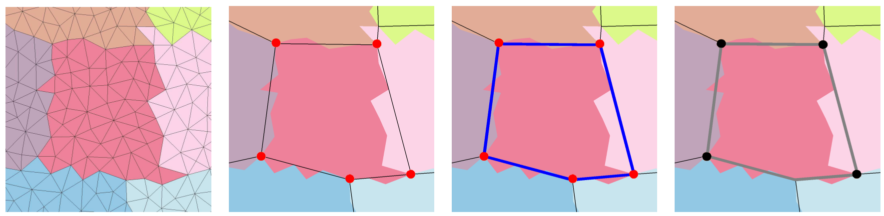
(g) Conforming relaxation (h) Local parameterizations (i) Barycentric subdivision

(j) Smoothing (k) Size (l) Orientation

(m) Angles (n) Ratio to desired sizing (o) Angle to desired orientation

Fig. 1. Overview. The algorithm takes as input a closed domain (a), a sizing field (b) and a cross field (c). It then operates on a triangle background mesh refined according to the sizing field (d). The initialization clusters background mesh triangles (e) so that the tiling roughly meets the size and shape criteria; A relaxation (f) then improves the tiling for shape and orientation while preserving size; A conforming relaxation (g) improves the degree of the tiles and the regularity of the tiling; A series of local parameterizations (h) further improves the degrees and regularity; Barycentric subdivision (i) generates a pure quadrangle mesh; Smoothing (j) finally improves the shape of the quadrangles. We depict the conformance to the sizing field with a color ramp ranging from white to blue (resp. white to red), for elements smaller (resp. larger) than the specified sizing field (k). We depict the conformance to the cross field with a color ramp ranging from white to gray (l). Irregular vertices are outlined in red for degree excess and in blue for degree deficit. We show the distribution of angles, as well as distributions measuring conformance to sizing and orientation. 1000 quadrangles, total time: 40 s.

- **Meta-edge.** A meta-edge connects two meta-vertices through an edge path of $\mathcal{M}$ such that all edges along the path are incident to 2 tiles in the interior, or 1 tile on the boundary. A tile is thus surrounded by a cycle of meta-edges.
- **Side.** A side is a chain of meta-edges around a tile. In particular, assuming a tile is surrounded by a cycle of at least 4 meta-edges, a subset of four of its meta-vertices can be chosen to represent quadrangle corners so that the tile has 4 sides. This assignment of sides will be useful in our algorithm since we target quadrangle elements.



(a) Tiling (b) Meta-vertices (c) Meta-edges (d) Sides (gray)

Fig. 2. Terminology defined in Section 2.1.

2.2 Initialization

We construct the background mesh $\mathcal{M}$ as a 2D constrained Delaunay triangulation that fits a polyline approximating the domain boundary. We then refine $\mathcal{M}$ through Delaunay refinement [27, 26] until all triangles are both well shaped (isotropic) and sized in accordance to the input sizing field within a small fraction (by default 0.1).

The initialization step aims at generating a tiling that conforms to the sizing field, and that roughly meets the shape criterion (see Table 1). First, we proceed by generating one tile per triangle of $\mathcal{M}$. We then recursively merge pairs of tiles using a priority queue of merging operations until the sizing field requirement is met (see Figure 1(e)). In order to favor isotropic tiles the merging operations are popped out of the queue in order of decreasing compactness, the latter being defined as the ratio between area and squared perimeter. We have experimented with another score which favors squares instead of discs but the simpler compactness score is sufficient. The idea of this step is similar to [16].

2.3 Relaxation

The first relaxation step aims at optimizing the initial tiling for the shape and orientation criteria while preserving the size (see Table 1). Optimization is carried out through a discrete clustering algorithm which operates over the background mesh so as to favor squares tiles which are both well sized and well oriented with respect to the sizing and cross fields ([31]).

Similar in spirit to [20], we consider the $L_\infty^{\mathcal{R}}$ metric related to a local Cartesian coordinate frame $\mathcal{R} = (\mathbf{u}, \mathbf{v})$ specified by the cross field. We compute an approximate discrete Voronoi diagram over the background mesh instead of computing the exact continuous Voronoi diagram [5], so as to deal with a simple triangle labeling procedure. The $L_\infty^{\mathcal{R}}$ distance between two points $p, q \in \mathbb{R}^2$ is defined as:

$$d_\infty^{\mathcal{R}}(p,q) = \max\left(|(p-q)\cdot\mathbf{u}|, |(p-q)\cdot\mathbf{v}|\right).$$

Using the continuous formulation the minimized energy is defined as:

$$\mathcal{G}\left(\{z_i\}_{i=1}^N, \{V_i\}_{i=1}^N\right) = \sum_{i=1}^N \int_{V_i} \rho(x)\, d_\infty^{\mathcal{R}}(x, z_i)\, dx,$$

where ρ is a density function defined on the domain: $\rho(x) = s(x)^{-4}$ (s denotes the sizing function to be preserved); z_i is a point generator and V_i is the Voronoi cell of z_i. Using a discrete formulation [30] now involving the background mesh $\mathcal{M}$ we consider a set of triangle generators $\{g_i\}_{i=1}^N$, and a tiling $\{T_i\}_{i=1}^N$. The energy to minimize is defined as:

$$\mathcal{H}\left(\{g_i\}_{i=1}^N, \{T_i\}_{i=1}^N\right) = \sum_{i=1}^N \left(\sum_{t_j \in T_i} \rho(c(t_j))\mathrm{area}(t_j) d_\infty^{\mathcal{R}}(c(t_j), c(\mu(T_i)))\right),$$

where $c(t)$ denotes the centroid of triangle t and μ denotes the triangle that contains the centroid of a tile. Energy $\mathcal{H}$ is iteratively minimized by alternating $L_\infty^{\mathcal{R}}$ discrete Voronoi partitioning and relocation of the generators to their tile centroids. Algorithm 2 provides a pseudo-code for the relaxation step, where N is the number of tiles and K is the total number of iterations.

Algorithm 2: Relaxation

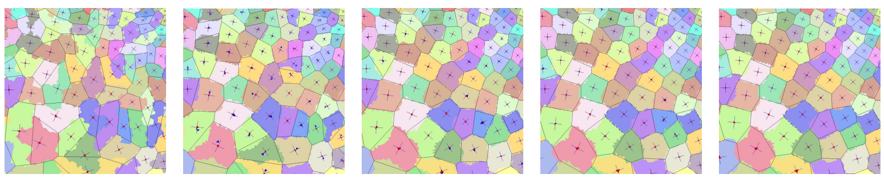
Input: Initial triangle generators $\{g_i^0\}_{i=1}^N$ and corresponding tiles $\{T_i^0\}_{i=1}^N$.
begin
  while *no convergence* **do**
    Discrete partitioning $\left(\{T_i^k\}_{i=1}^N \text{ partition associated to } \{g_i^k\}_{i=1}^N\right)$
    Relocate generators to centroids $\left(\forall i,\ 1 \le i \le n,\ g_i^{k+1} = \Gamma(g_i^k)\right)$
Output: Optimized triangle generators $\{g_i^K\}_{i=1}^N$ and corresponding tiles $\{T_i^K\}_{i=1}^N$.

Discrete partitioning is achieved by flooding the domain one triangle at a time from their generators with a dynamic priority queue [12]. Each tile is initialized to be its triangle generator, and a priority queue is filled with (up to three) incident triangles (candidates for flooding) per generator. In order to favor square tiles, the triangles are popped out of the queue and added to tiles in increasing order of $L_\infty^{\mathcal{R}}$ distance to their respective triangle generator, where $\mathcal{R}$ is the local Cartesian coordinate frame specified by the cross field at the generator centroid.

Relocation is achieved through computing the (triangle) center of mass of each tile. In the continuous case the center of mass of a tile does not depend on the metric chosen to measure the object, and we observe a very similar behavior in our discrete setup. After the k-th iteration, for each tile T_i^k, we choose to find the triangle that minimizes the following energy:

$$f(t) = \sum_{t_j \in T_i^k} \rho(c(t_j)) \operatorname{area}(t_j) \, \|c(t_j) - c(t)\|_2^2,$$

where $t \in T_i^k$. We denote by $\Gamma : g_i^k \mapsto g_i^{k+1}$ the operation that computes the triangle centroid of the tile T_i^k associated to g_i^k. In such discrete algorithm convergence is reached when $\forall i,\ 1 \leq i \leq N, \Gamma(g_i^k) = g_i^k$. As convergence for the $L_\infty^{\mathcal{R}}$ metric on arbitrary domains is not guaranteed, we also specify a maximum number of iterations (by default set to 50). Figure 3 depicts some iterations.



(a) Initial tiling (b) 1 iteration (c) 5 iterations (d) 10 iterations (e) Converged

Fig. 3. Discrete $L_\infty^{\mathcal{R}}$ iterations with a non-uniform sizing field (the cross field is shown).

As expected, the relaxation leads to a tiling of the domain with mostly square tiles which are well oriented and well sized (similar in spirit to a square packing approach [28]), even if a varying sizing field inevitably implies shape distortion. We further observe in Figure 4 that although well shaped, the tiles are generally not conforming (see the many T-junctions), hence most of them would generate degree-6 polygons and the final mesh would contain many irregular vertices (generally of degree 3). We describe next a conforming relaxation procedure which aims at generating quasi 2-conforming configurations.

2.4 Conforming Relaxation

A closer look at Figure 4 reveals that the tiling is in general already conforming in one direction. We call this configuration 1-conforming. This is explained by the fact that a general tiling of the plane with equally sized square tiles is 1-conforming (see Figure 5, left). Our goal is to further relax the tiling so as to obtain quasi 2-conforming configurations (5, middle).

An intuitive understanding of the conforming relaxation procedure can be gathered by looking at Figure 5(left) and realizing that we could shift the three square columns vertically as little as possible so as to tend toward a perfect 2-conforming configuration. In order to obtain the aforementioned shift, we propose to simply shift the centroid during the relocation of a relaxation iteration. The only remaining technicality now resides in the way to compute the shift. Although simple at first glance,

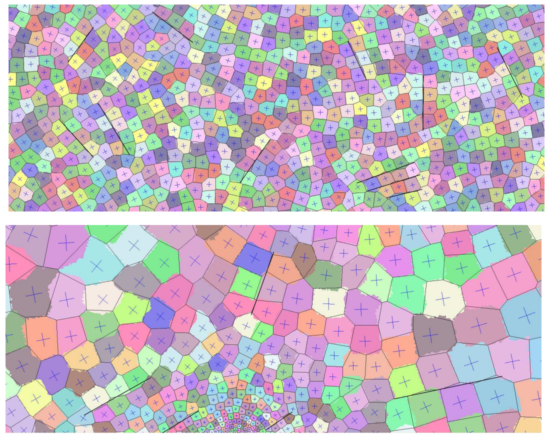
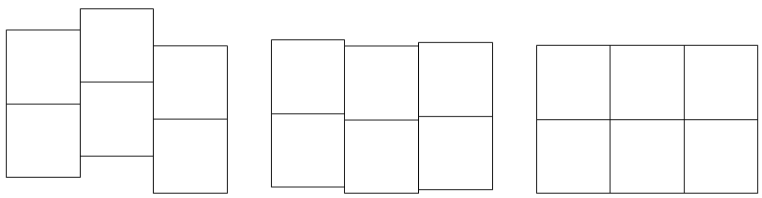


Fig. 4. After relaxation, for uniform (top) and non-uniform (bottom) sizing field. Tiles are in general conforming to one direction of the local cross field. The black lines depict some local conforming directions.



(a) 1-conforming (b) Quasi 2-conforming (c) 2-conforming

Fig. 5. Tiling with squares. Left: general configuration after relaxation. Middle: general configuration after conforming relaxation. Right: ideal configuration sought after.

recovering the local 1-conforming direction (vertical or horizontal) is already non trivial, and so is finding the shift magnitude and orientation (see Figure 6). In addition, both size and cross fields vary over the domain, requiring not just shifting the tiles but also sacrificing aspect ratio to reach conforming.

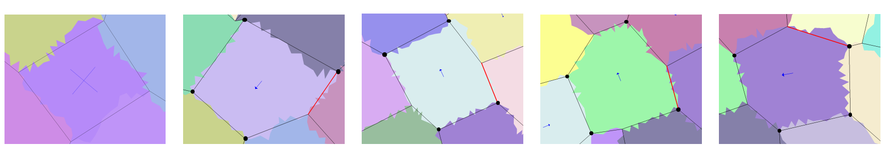
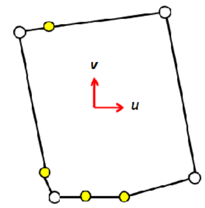


Fig. 6. Shifting centroids. We depict examples of shifts with increasing ambiguity. A tile which is already quadrangle is not shifted (left). A tile with one side split into two meta-edges (middle left), the chosen quadrangle corners are depicted in black. A tile with two parallel sides split (middle). A tile with three sides split (middle right). Another ambiguous case (right).

Tiles which are already quadrangles or triangles (with 4 or 3 meta-vertices) are

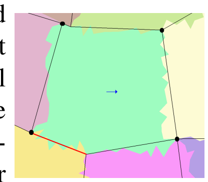


not shifted. While we assume that after relaxation all tiles are squares (geometry-wise), in general they are degree-6 tiles (see Figure 5(a)). We thus first need to choose which four of its meta-vertices form a square locally aligned to the cross field. We first compute the triangle centroid of the tile as during relaxation (see 2.3). Its cross (given by the input cross field) is now taken as local Cartesian coordinate frame $\mathcal{R} = (\mathbf{u}, \mathbf{v})$. Denote by $v_1, \ldots, v_p$ ($p > 4$) the meta-vertices of the tile, ordered by circulating along the tile boundary. To decide which of the meta-vertices are chosen as (ordered) corners c_1, c_2, c_3, c_4 we maximize through dynamic programming the alignment of the sides (see 2.1) with the axes of $\mathcal{R}$ through maximization of the following energy:

$$\mathcal{E} = \max_{\substack{\mathbf{a} \subset \{\mathbf{u},\mathbf{v}\} \\ \{c_1,c_2,c_3,c_4\} \subset \{v_1,\ldots,v_p\}}} \left[\min \left(|(c_2 - c_1) \cdot \mathbf{a}|, |(c_3 - c_2) \cdot \mathbf{a}^{90}|, |(c_4 - c_3) \cdot \mathbf{a}|, |(c_1 - c_4) \cdot \mathbf{a}^{90}| \right) \right],$$

where $\mathbf{a}^{90}$ denotes vector $\mathbf{a}$ (which stands for $\mathbf{u}$ or $\mathbf{v}$) rotated by 90 degrees.

In addition to assigning corners, the maximum of $\mathcal{E}$ for each side provides a local reference direction of the cross field ($\mathbf{u}$ or $\mathbf{v}$). For each meta-edge, we then compute the length of its projection on its reference direction ($\mathbf{u}$ or $\mathbf{v}$). Among all meta-edges, we then select only the ones with exactly one end meta-vertex coinciding to a corner meta-vertex, and pick the longest one, denoted by e. We then shift the centroid along a line parallel to $\mathbf{a}$, in the opposite direction to the corner meta-vertex of e. The magnitude of the shift is chosen as a fraction (0.2 in our experiments) of the length of the projection of e (see inset). The number of iterations of conforming relaxations is a user-specified parameter (set to 20 for all examples shown). Figure 7 depicts how such conforming relaxation brings some initial 1-conforming configurations to quasi 2-conforming configurations. The latter can be fixed by a series of local parameterizations which we describe next.



2.5 Local parameterizations

The previous conforming relaxation step favored quasi-2-conforming configurations (Figure 5(b)) that we wish to turn into 2-conforming configurations (Figure 5(c)) so as to improve both degree and regularity criteria (see Table 1) while not negatively affecting the previous efforts made by previous steps.

Figure 7 reveals that the many quasi-2-conforming configurations exhibit similar topological arrangements of the tiles. We call *butterfly* a set of four tiles incident to a short meta-edge connecting two degree-3 meta-vertices (see Figure 8). Inspired by [23] we remove many of these butterfly configurations through local parameterizations on square domains, which merge 2 degree-3 meta-vertices into one degree-4 meta-vertex.

For each butterfly, we consider the union of its four tiles A, B, C, D as a subdomain Ω_{ABCD}, and first perform a classification of its meta-vertices as depicted in

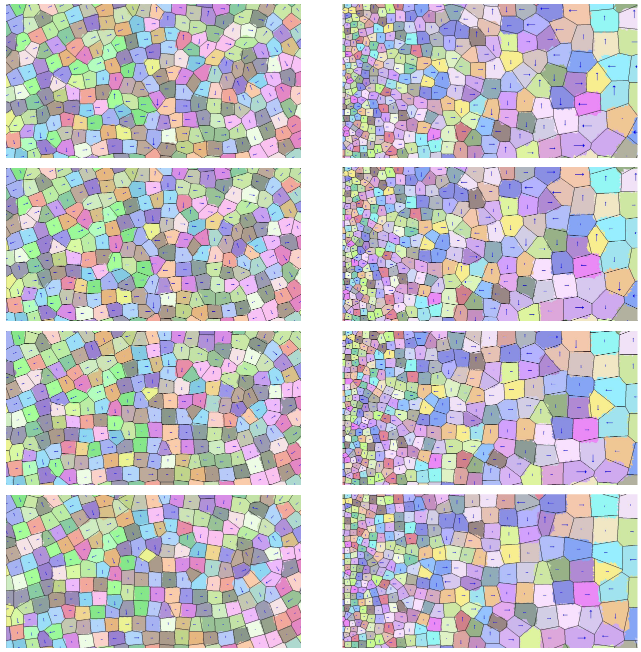


Fig. 7. Conforming relaxation with uniform sizing (left) and non-uniform sizing (right). From top to bottom: initial, 1 iteration, 5 iterations and 10 iterations. In order to bring some initial 1-conforming configurations to quasi 2-conforming configurations, centroids of two adjacent tiles are locally shifted, which can induce size distortion.

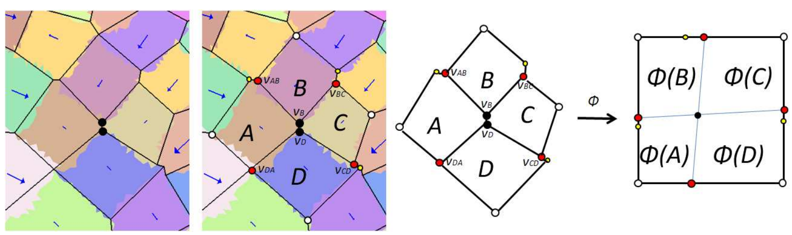


Fig. 8. Butterfly configuration. Left: a short meta-edge connects two degree-3 meta-vertices (outlined in black) which share four tiles. Middle left: among all meta-vertices of the four-tile subdomain we identify two inner meta-vertices (black), 4 interface meta-vertices separating the four tiles A, B, C, D pairwise (red), four corner meta-vertices (white) and the remaining meta-vertices (yellow). Middle right and right: the four tiles parameterized on a square. The parameterization preserves the topology of the interfaces on the boundary of the four tile subdomain. Some tiles may be smaller in parameter space so as to better fit the sizing field.

Figure 8. While the inner (black) and interface (red) meta-vertices can be easily classified from topological (adjacency) relationships between the tiles, distinguishing the four corners among the other meta-vertices (possibly many for rapidly varying sizing fields) requires incorporating a geometric criterion related to the local orientation of the tiles.

Contrary to the way we choose the corners for the conforming relaxation step, the orientation is, this time, not given by the cross field but instead depends on the relaxed four tiles (the rationale being, once again, to avoid undoing the previous enhancements). Denote by $V_{AB}, V_{BC}, V_{CD}, V_{DA}$ the four interface meta-vertices. We estimate a reference Cartesian frame by fitting two lines through principal component analysis to the segment sets $([V_{AB}, V_B], [V_{CD}, V_D])$ and $([V_{DA}, V_D], [V_{BC}, V_B])$ [11]. Among these two lines the most reliable one (i.e., the line with minimum variance in the orthogonal direction) is chosen as reference direction. To select four corners among the meta-vertices we again resort to a dynamic programming approach similar to the one used in Section 2.4.

Our goal is to parameterize Ω_{ABCD} on a square domain and to label its triangles (again with labels A,B,D,C) such that i) the chosen corner meta-vertices coincide with the corners of the square, ii) the interfaces at the boundary of Ω_{ABCD} are preserved, iii) the two inner meta-vertices are merged into a degree-4 vertex and iv) Ω_{ABCD} is partitioned with 4 tiles with a disk topology. Call ϕ the parameterization that maps Ω_{ABCD} on the square domain. We first constrain the whole boundary of $\phi(\Omega_{ABCD})$ so as to respect the chosen corners and using an arc-length parameterization in-between these corners. We then parameterize Ω_{ABCD} using the mean value coordinate approach [15] and compute, in parameter space, the intersection point $\phi(v^*)$ between the two line segments $(\phi(V_{AB})\phi(V_{CD}))$ and $(\phi(V_{BC})\phi(V_{DA}))$. The nearest vertex (of degree at least 4) from $\phi(v^*)$ is then chosen as inner vertex, which means that the issuing vertex v^* will be the center degree-4 meta-vertex. While simple at first glance once the inner and boundary vertices are decided upon, a naive triangle labeling step based on localization within quadrilaterals in parameter space can lead to improper topological partitioning. For this reason we trace four edge paths from $\phi(v^*)$ to $\phi(V_{AB}), \phi(V_{BC}), \phi(V_{CD}), \phi(V_{DA})$ in order of increasing length (shorter segments first) so as to determine proper interfaces between the triangle labels. These edge paths are constrained not to intersect except at the inner vertex and to connect no other boundary vertices than their target vertex $V_{AB}, V_{BC}, V_{CD}, V_{DA}$. Upon successful completion the triangles of Ω_{ABCD} are labeled, and this ends the butterfly removal procedure.

To avoid distorting the shape and orientation of the final mesh, we only remove butterflies whose inner meta-edge length is smaller than a fraction of the local sizing field (0.5 in our experiments). We use a dynamic priority queue to gracefully deal with butterflies in order of increasing inner meta-edge length. Figure 9 depicts how local parameterizations improve degree and regularity criteria for both uniform and non-uniform sizing field cases.

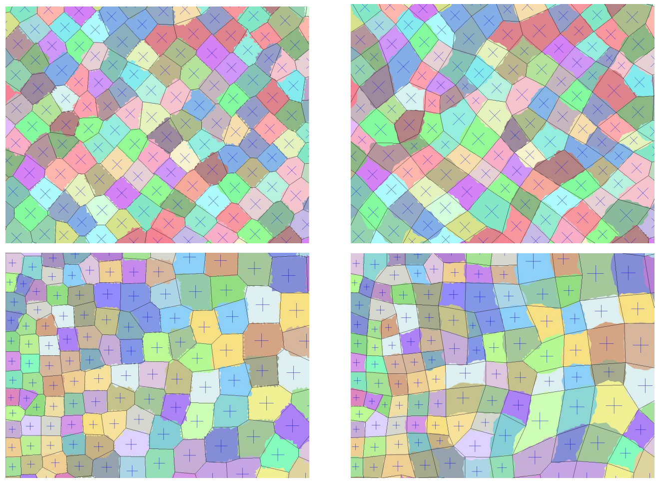


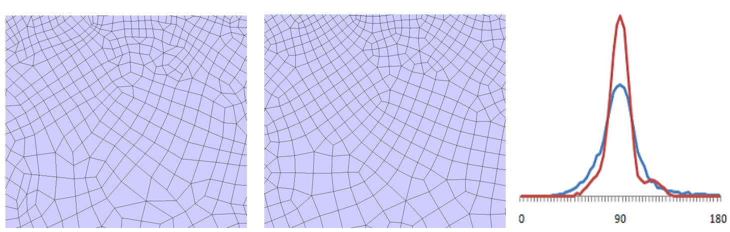
Fig. 9. Local parameterizations. Uniform sizing field (top left and top right) and non-uniform sizing field (bottom left and bottom right) before (left) and after (right) local parameterizations. The cross field is shown.

2.6 Barycentric Subdivision

After conforming relaxation there is no guarantee that all tiles are quadrangles. Experimentally we obtain an order of 75% quadrangles for uniform sizing and of 60% quadrangles for non-uniform sizing. We thus resort to barycentric subdivision (see Figure 1(i)) in order to meet the degree criterion (Table 1). The edge lengths of the quadrangles are reduced by a factor of two compared to the mesh before barycentric subdivision. This factor is taken into account during all previous steps of the algorithm.

2.7 Smoothing

Near T-junctions the previous barycentric subdivision step generates highly distorted quadrangles. We thus resort to a few iterations of Laplacian smoothing (see Figures 1(j) and 10) so as to enhance the shape criterion.



(a) Before smoothing (b) After smoothing (c) Angle distributions

Fig. 10. Smoothing. Mesh before (a) and after smoothing (b). We compare the distribution of angles before (blue) and after (red) smoothing.

3 Results

Our algorithm is implemented in C++ using the CGAL library [10]. All examples were computed on a 2.40GHz PC with a single thread. Each of our figures highlights irregular vertices in blue for degree deficit, and in red for degree excess. To depict conformance to the input cross field we use a color ramp ranging from white to gray, where white means perfect conformance and gray means 45 degree distortion. To depict conformance to the input sizing field we use a color ramp ranging from white to blue (resp. red) for tiles smaller (resp. larger) than targeted. We also depict distributions of angles, conformance to sizing, and conformance to cross field.

We ran a large number of examples of various size and complexity in order to assess our results and compare them to state-of-the-art methods. Figure 11 depicts how a final quadrangle mesh conforms to both size and orientation on a geographic map. The input sizing field is set to be the largest 1-Lipschitz function constrained to match the local feature size estimate of the input domain boundary [2]. The cross field is set to be the smoothest cross field constrained to be tangential to the input domain boundary. The distribution of angles is shown, with over 80% of angles within the interval $[75 - 105]$. The algorithm takes 350 seconds, with two third of the time spent on local parameterizations. The peak memory usage is 200 MBytes.

Figure 12 depicts an example with a constant cross field combined with a rapidly varying sizing field. Irregular vertices inevitably appear between dense and sparse mesh areas, and the orientation is partially distorted. Figure 13 shows a trivial domain example with a constant cross field and compares uniform vs. non-uniform sizing. Figure 14 depicts examples of uniform sizing and varying cross fields set to be smooth and tangential to the input domain boundary.

Figure 15 compares our results with [7] using a uniform sizing as this method is not primarily aimed at handling rapidly varying sizing fields. Our approach better preserves orientations near the domain boundary at the price of a larger number of irregular vertices.

We do not provide direct control over the final number of vertices of the final mesh as it depends on both the input sizing field and the number of degree-4 tiles after local parameterizations (experimentally near 70%). The efficiency of the algorithm can be improved by accelerating the relaxation step [30] and by numerical optimizations during the local parameterizations.

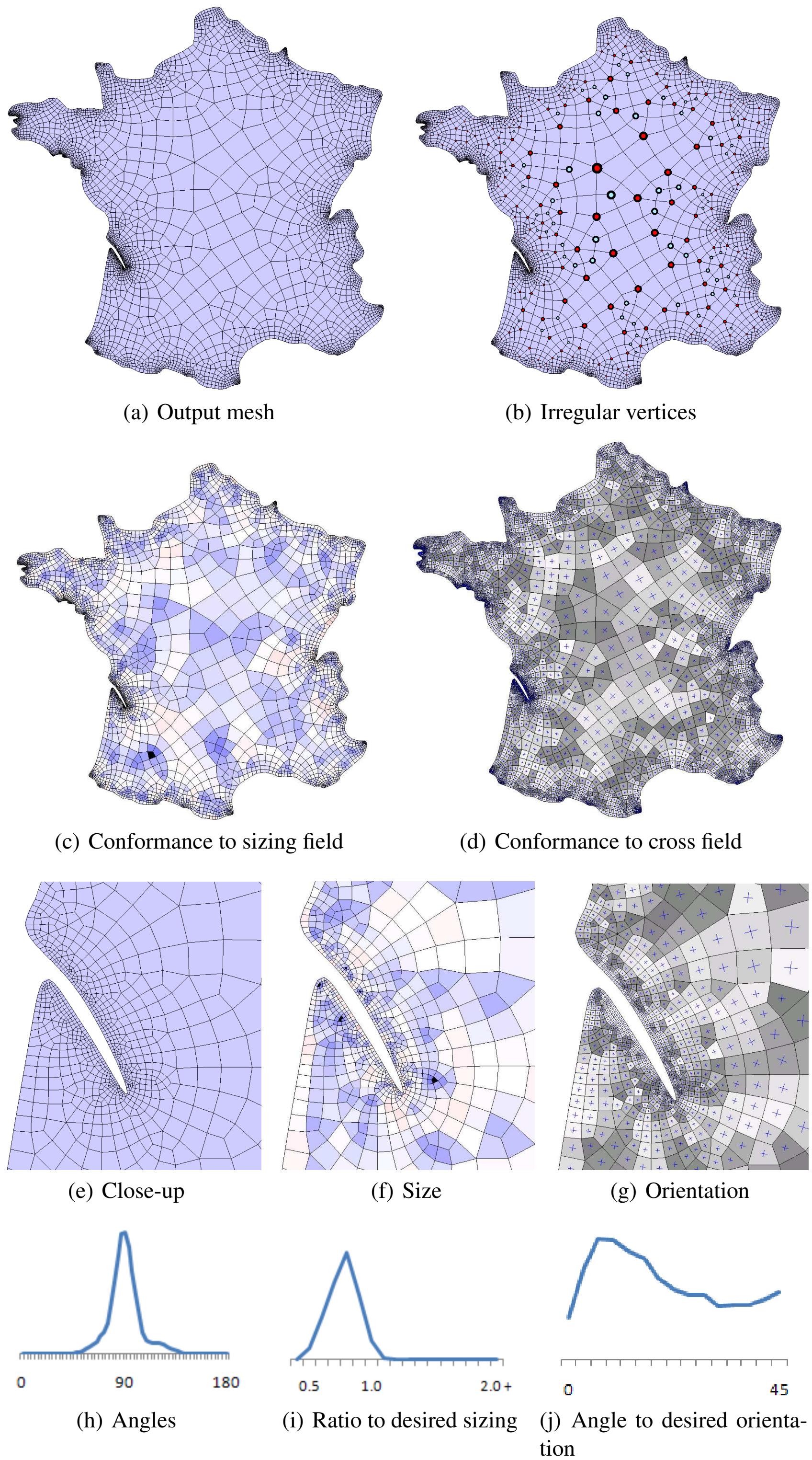
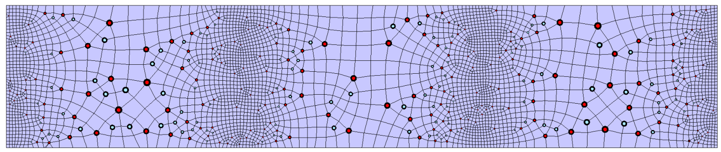
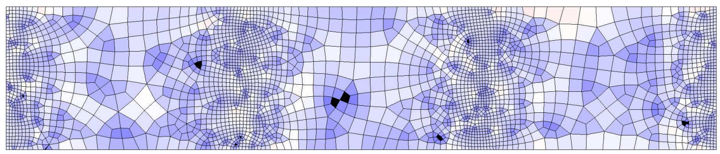


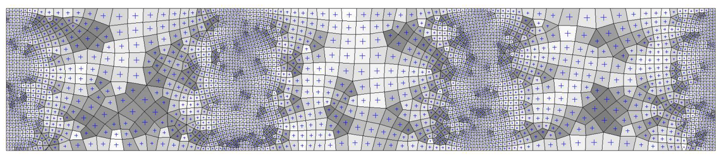
Fig. 11. France. The final mesh (a) contains 77% regular vertices (b). It conforms to the sizing field (c) as well as to the cross field (d). The close-up depicts an area where size and orientation vary rapidly. 4500 quadrangles, total time: 300 s.



(a) Output mesh and irregular vertices

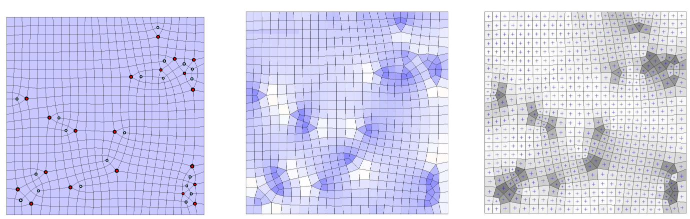


(b) Conformance to sizing field

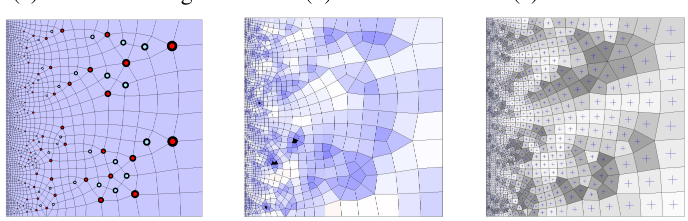


(c) Conformance to cross field

Fig. 12. Non-uniform sizing field defined as $h(x) = 0.01\left(2 + \sin\left(6x\pi - \frac{\pi}{2}\right)\right)$, $0 < x < 1$. Total time: 480 s.

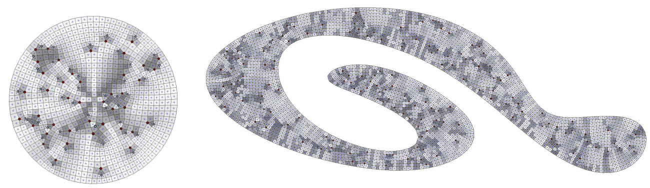


(a) Uniform sizing (b) Size (c) Orientation



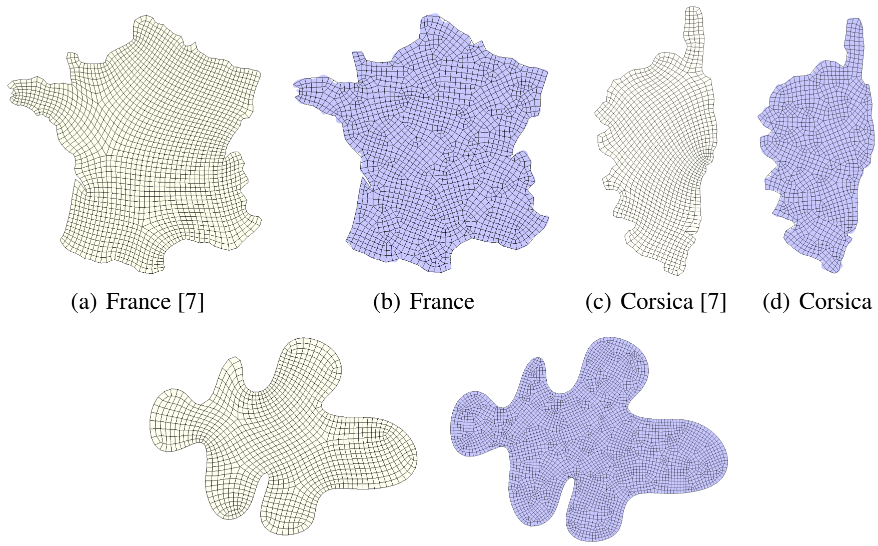
(d) Non-uniform sizing (e) Size (f) Orientation

Fig. 13. Uniform vs. non-uniform sizing. The cross field is constant and axis-aligned. Total time: 30 s. and 60 s.



(a) Disc (b) Free form

Fig. 14. Non-uniform cross fields. Sizing fields are uniform. Total time: 45 s. and 450 s.



(a) France [7] (b) France (c) Corsica [7] (d) Corsica

(e) Freeform [7] (f) Freeform

Fig. 15. Comparison with Bommes et al. [7]. The cross field is set to be tangential to the input domain boundary. The meshes produced by [7] are more regular at the price of increased shearing distortion.

4 Conclusion

We proposed a principled approach for the automatic generation of quadrangle meshes from arbitrary 2D domains. The main methodology consists of enhancing a rough initial tiling by carefully addressing one meshing requirement at a time (size, shape, orientation, degree, regularity) in an order designed not to undo previous enhancements. While other similar approaches (such as [28]) only show examples on simple domains and smooth cross fields, our experiments confirm that the output quadrangles meshes conform both to the input sizing and cross fields, even on complex domains and for rapidly changing fields.

Size and orientation conformance comes at the price of a larger number of irregular vertices. For applications requiring coarse base complexes, we could potentially improve our approach by allowing the user to trade conformance to input fields for increased regularity. We also wish to improve the conforming step by resorting to a different strategy when it fails locally, and the parameterization step by making it more general for an arbitrary set of tiles clustered around a target irregular vertex.

Our approach is primarily based on relaxation and local parameterizations. Because the main steps of the algorithm only deal with labeling the triangles of a background triangle mesh, our implementation is simple and reliable, and is less prone to numerical robustness issues. Resorting to local parameterizations on trivial convex domains such as [23] provides us with scalability and robustness. For these reasons our approach could be extended to reliable quadrangle surface remeshing (including anisotropic quadrangle surface remeshing). Most of the main steps also seem to generalize to hexahedron domain tiling, except for the barycentric subdivision step. As future work we plan to alleviate this problem so as to extend the main principles behind our approach for hexahedron domain tiling.

Acknowledgments

The authors are grateful to Mathieu Desbrun for discussion and advice, and to David Bommes for experiments. This work was funded by the European Research Council (ERC Starting Grant 'Robust Geometry Processing', Grant agreement 257474), and by a French ANR Grant (GIGA ANR-09-BLAN-0331-01).

References

1. P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. Anisotropic polygonal remeshing. *ACM Trans. Graph.*, 22(3):485–493, 2003.
2. L. Antani, C. Delage, and P. Alliez. Mesh sizing with additively weighted Voronoi diagrams. In *Proc. of the 16th Int. Meshing Roundtable*, pages 335–346, 2007.
3. M. W. Bern and D. Eppstein. Quadrilateral meshing by circle packing. *Int. J. Comput. Geometry Appl.*, 10(4):347–360, 2000.
4. I. M. Boier-Martin, H. E. Rushmeier, and J. Jin. Parameterization of triangle meshes over quadrilateral domains. In *Symposium on Geometry Processing*, pages 197–208, 2004.
5. J.-D. Boissonnat, M. Sharir, B. Tagansky, and M. Yvinec. Voronoi diagrams in higher dimensions under certain polyhedral distance functions. *Discrete & Computational Geometry*, 19(4):485–519, 1998.
6. D. Bommes, T. Lempfer, and L. Kobbelt. Global structure optimization of quadrilateral meshes. *Comput. Graph. Forum*, 30(2):375–384, 2011.
7. D. Bommes, H. Zimmer, and L. Kobbelt. Mixed-integer quadrangulation. *ACM Trans. Graph.*, 28(3), 2009.
8. H. Borouchaki and P. J. Frey. Adaptive triangular-quadrilateral mesh generation. *International Journal of Numerical Methods in Engineering*, 41:915–934, 1996.
9. D. Bremner, F. Hurtado, S. Ramaswami, and V. Sacristan. Small strictly convex quadrilateral meshes of point sets. *Algorithmica*, 38(2):317–339, 2003.
10. CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.
11. Principal component analysis in CGAL. http://www.cgal.org.

12. D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM Trans. Graph.*, 23:905–914, 2004.
13. K. Crane, M. Desbrun, and P. Schröder. Trivial connections on discrete surfaces. *Comput. Graph. Forum*, 29(5):1525–1533, 2010.
14. S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart. Spectral surface quadrangulation. *ACM Trans. Graph.*, 25(3):1057–1066, 2006.
15. M. S. Floater. Mean value coordinates. *Comput. Aided Geom. Des.*, 20:19–27, 2003.
16. M. Garland, A. J. Willmott, and P. S. Heckbert. Hierarchical face clustering on polygonal surfaces. In *SI3D*, pages 49–58, 2001.
17. J. Huang, M. Zhang, J. Ma, X. Liu, L. Kobbelt, and H. Bao. Spectral quadrangulation with orientation and alignment control. In *ACM SIGGRAPH Asia 2008 papers*, SIGGRAPH Asia '08, pages 147:1–147:9, 2008.
18. J. Daniels II, C. T. Silva, and E. Cohenn. Localized quadrilateral coarsening. *Comput. Graph. Forum*, 28(5):1437–1444, 2009.
19. Y.-K. Lai, L. Kobbelt, and S.-M. Hu. An incremental approach to feature aligned quad dominant remeshing. In *Proceedings of the ACM symposium on Solid and physical modeling*, SPM '08, pages 137–145, 2008.
20. B. Lévy and Y. Liu. L^p centroidal voronoi tessellation and its applications. *ACM Trans. Graph.*, 29(4), 2010.
21. M. Marinov and L. Kobbelt. Direct anisotropic quad-dominant remeshing. In *Pacific Conference on Computer Graphics and Applications*, pages 207–216, 2004.
22. S. J. Owen, M. L. Staten, S. A. Canann, and S. Saigal. Q-morph: an indirect approach to advancing front quad meshing. *International Journal for Numerical Methods in Engineering*, 44(9):1317–1340, 1999.
23. N. Pietroni, M. Tarini, and P. Cignoni. Almost isometric mesh parameterization through abstract domains. *IEEE Trans. Vis. Comput. Graph.*, 16(4):621–635, 2010.
24. W. R. Quadros, K. Ramaswami, F. B. Prinz, and B. Gurumoorthy. Laytracks: A new approach to automated quadrilateral mesh generation using medial axis transform. In *IMR*, pages 239–250, 2000.
25. N. Ray, W. C. Li, B. Lévy, A. Sheffer, and P. Alliez. Periodic global parameterization. *ACM Trans. Graph.*, 25(4):1460–1485, 2006.
26. L. Rineau and M. Yvinec. A generic software design for delaunay refinement meshing. *Comput. Geom.*, 38(1-2):100–110, 2007.
27. J. R. Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Comput. Geom.*, 22(1-3):21–74, 2002.
28. K. Shimada and J.-H. Liao andT.i Itoh. Quadrilateral meshing with directionality control through the packing of square cells. In *IMR*, pages 61–75, 1998.
29. Y. Tong, P. Alliez, D. Cohen-Steiner, and M.Desbrun. Designing quadrangulations with discrete harmonic forms. In *Symposium on Geometry Processing*, pages 201–210, 2006.
30. S. Valette and J.-M. Chassery. Approximated centroidal voronoi diagrams for uniform polygonal mesh coarsening. *Comput. Graph. Forum*, 23(3):381–390, 2004.
31. S. Valette, J.-M. Chassery, and R. Prost. Generic remeshing of 3d triangular meshes with metric-dependent discrete voronoi diagrams. *IEEE Trans. Vis. Comput. Graph.*, 14(2):369–381, 2008.
32. P. Wolfenbarger, J. Jung, C. R. Dohrmann, W. R. Witkowski, M. J. Panthaki, and W. H. Gerstle. A global minimization-based, automatic quadrilateral meshing algorithm. In *IMR*, pages 87–103, 1998.
33. M. Zhang, J. Huang, X. Liu, and H. Bao. A wave-based anisotropic quadrangulation method. *ACM Trans. Graph.*, 29(4), 2010.