Volume xx (200y), Number z, pp. 1–24

State of the Art in Quad Meshing

David Bommes [1], Bruno Lévy [2], Nico Pietroni [3], Enrico Puppo [4], Claudio Silva [5,7], Marco Tarini [6], Denis Zorin [7]

[1]RWTH Aachen University, Germany
[2]ALICE / INRIA , France
[3]National Research Council, Italy
[4]University of Genova, Italy
[5]Polytechnic Institute of NYU, USA
[6]University of Insubria, Italy
[7]New York University, USA

Abstract

Triangle meshes have been nearly ubiquitous in computer graphics, and a large body of data structures and geometry processing algorithms based on them has been developed in the literature. At the same time, quadrilateral meshes, especially semi-regular ones, have advantages for many applications, and significant progress was made in quadrilateral mesh generation and processing during the last several years. In this State of the Art Report, we discuss the advantages and problems of techniques operating on quadrilateral meshes, including surface analysis and mesh quality, simplification, adaptive refinement, alignment with features, parametrization, and remeshing.

1. Introduction

Polygonal meshes are important representations with a large number of applications in geometric modeling, computer graphics, mechanical engineering, simulation, architecture, etc. Such representations are based on the idea of *cell decomposition*: a complex object is represented with an assembly of (possibly many) simple polygonal cells. Triangles and quadrilaterals are the most common cells used to for surfaces. *Quad meshes*, i.e., meshes made entirely of quadrilaterals, have been widely used for many years in CAD and simulation, because a number of tasks are better suited to quad meshes than to triangle meshes. While triangle meshes are much more common in computer graphics, and most research in geometry processing has focused on them, the advantages of quad meshes for graphics applications are also well understood.

This report provides a discussion of specific characteristics of quad meshes, as well as a survey of recent research on quad mesh processing.

1.1. Terminology and classification

The constituents of a two-dimensional polygonal mesh are: *facets* (a.k.a. *elements,* in the context of the finite-element method (FEM)), *edges* that bound facets, and *vertices* that bound edges. We consider primarily *conforming* meshes; in a conforming mesh, any two faces share either a single vertex, or an entire common edge. We also briefly discuss *T-meshes*, for which two faces may share only a part of an edge. In addition, we assume that the meshes are *manifold*, i.e., any edge may be shared by either one or two incident facets and the set of facets connected to each vertex forms a single fan (i.e., there is no "bow tie" configuration). An edge with two incident facets is said to be *internal,* while and edge with just one incident facet is said to be *boundary.* A vertex of a boundary edge is also said to be *boundary,* otherwise it is said to be *internal.*

The *valence* of a vertex is the number of its incident edges, while its *star* is the set of its incident facets and edges. A *triangle mesh* is a mesh in which all facets are triangles, while a *quad mesh* is a mesh in which all facets are quadrilaterals. In some cases, we consider *quad-dominant meshes,* in which

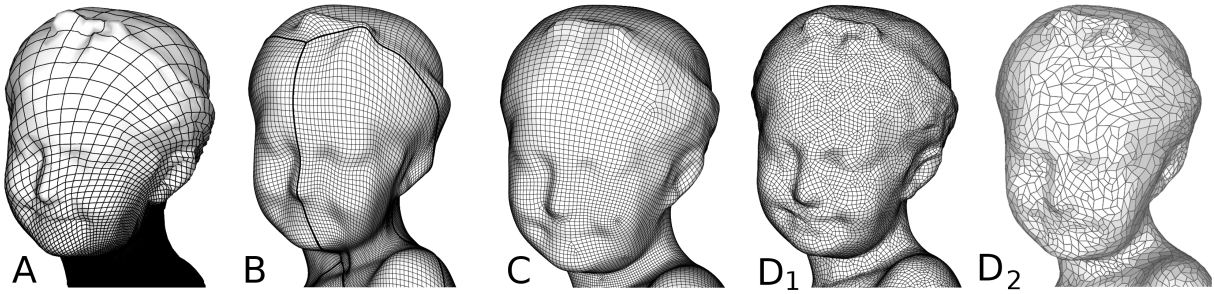


Figure 1: Quad meshes categories. A: regular; B: semi-regular; C: valence semi-regular; D1-D2: unstructured (D2 being more irregular than D1).

the majority of facets are quadrilaterals, while there may be a small fraction of non-quadrilateral facets, typically triangles and/or pentagons. An internal vertex in a quad mesh is *regular* if it has valence 4. For quad meshes, three types of regular boundary vertices can be distinguished. A regular *non-corner* boundary vertex of a quad mesh has valence 3, and convex/concave *corner* boundary vertex have valences 2 and 4 respectively. A vertex v that is not regular is called *irregular,* or *extraordinary*.

There is a natural relation between quad meshes and cross fields, i.e. an assignment of a pair of directions to each surface point: a cross field may be regarded as a quad mesh with infinitely small quads with edges aligned with cross field directions. Several mathematical notions relevant to the analysis of continuous cross fields can be used in the context of quad meshes. The *index theory* classifies the singularities of a vector field and studies their relation with the topology of the underlying manifold (see [Yau05] for a historical perspective). Similar theory exists for cross fields ([RVLL08]). In quad mesh context, field singularities correspond to the extraordinary vertices of quad meshes. Another important concept in the vector and cross field setting is *separatrices*, i.e. integral lines starting at singularities. In the cases when all separatrices also end at singularities, they define a natural partition of the manifold. The quad mesh counterpart of a separatrix corresponds to a path of edges connecting an extraordinary vertex to another extraordinary vertex or boundary (see Fig. 18 and Sec. 4.3.1 for more details).

Quad meshes can be loosely classified into several classes, based on the degree of regularity (Figure 1):

- A *regular mesh*, or a *geometry image* [GGH02], can be globally mapped to a rectangular subset of a square tiling, Regular meshes have a limited scope of applicability as these are suitable for surfaces of disk or toroidal topology only (a toroidal topology mesh can be obtained by identifying the opposite sides of a regular mesh without introducing irregular vertices).
- A quad mesh is *semi-regular* if it is obtained by gluing, in a conforming way, several regular 2D arrays of quads side to side. Each such regular submesh is called a *patch,* and the number of patches is assumed to be much smaller than the total number of facets. In the context of FEM, a semi-regular mesh is called a *multi-block grid* (blocks corresponding to patches). In a semi-regular quad mesh, all vertices that are internal to patches or lie along their boundary edges are regular, while only vertices that lie at corners of patches may possibly be extraordinary. Semi-regular meshes represent the most important class in terms of applications.
- A quad mesh is *valence semi-regular* if most of its vertices have valence 4. All semi-regular meshes are valence semi-regular, but not every valence semi-regular mesh can be partitioned into a small number of patches. While most authors conform to the definition of semi-regular given above, some authors use the term semi-regular to refer to valence semi-regular. However, the distinction is important as the difference between these two classes can be really dramatic (see Sec. 4.3) and has a significant impact on possible applications. Differentiating these two classes of quad meshes allows us to differentiate more precisely the algorithms that aim at producing meshes with a patch structure, from those algorithms that minimize the number of irregular vertices only.
- A quad mesh is *unstructured* if a large fraction of its vertices are irregular. An unstructured mesh is obtained for instance from splitting each facet of an arbitrary triangle mesh into three quads.

Note that, while the first class has a precise definition, there is a continuum of meshes joining the other three classes. The two parameters of this continuum are the number of irregular vertices and the minimal number of patches the mesh can be partitioned into. Starting from an unstructured mesh (most vertices are irregular), and decreasing the number of irregular vertices, results in a transition to a valence semi-regular mesh. A valence semi-regular mesh may have a large minimal number of patches (which can be obtained by partitioning the mesh along all separatrices); modifying the valence semi-regular mesh structure to reduce the number of patches leads to a semi-regular mesh. For any

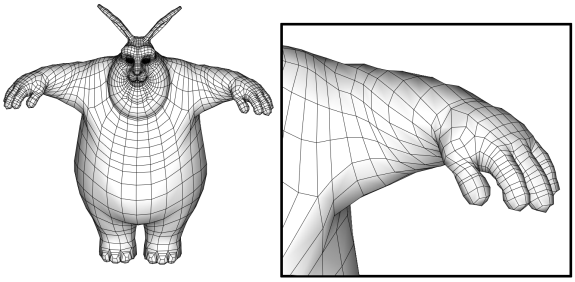


Figure 2: A quad mesh for computer animation (courtesy of the Peach project, Open Source "Big Buck Bunny" movie). Note how the mesh is "twisted" around the arms. This facilitates animating the character in a way that reflects his "don't mess with me" personality.

semi-regular mesh, the patches form a coarse unstructured mesh.

If non-conforming (*T-mesh*) patch structures are allowed, the relation between semi-regular and valence semi-regular meshes changes: the minimal number of patches now is directly tied to the number of irregular vertices [EGKT08, MPKZ10], so the transition from valence semi-regular to semi-regular is characterized by the decrease in the number of T-joints (face vertices interior to the edges of other faces).

1.2. Applications

Quad meshes are preferred in many applications for several reasons:

1. Most geometry has two dominant local directions, typically associated with principal curvature directions or local sharp features, to which quads can be aligned; use of triangle meshes necessitates an arbitrary choice of a third edge direction. The alignment of elements of a mesh to given directions is crucial in capturing shape features as well as the semantics of modeled objects, especially if they need to be segmented or animated. Tensor-product high-order bases, associated with quad meshes are preferred for surface representation, tensor-product B-splines and Catmull-Clark surfaces being the dominant industry standards.
2. Patches of semi-regular quad meshes, and more generally rectangular topology submeshes of valence semi-regular meshes, naturally match the sampling pattern of all types of textures, from images to displacement maps.

These properties are important in the context of a number of common applications.

Polygonal modeling: Many objects and characters used for real-time rendering, e.g., in the context of video-games, virtual reality, and CG animation, are represented as polygonal meshes, designed by artists using interactive modeling systems. Careful design requires that mesh elements be arranged to follow line features of the represented shape, as well as to adapt to deformations imposed by animation (see Figure 2). In this context, quad meshes are more convenient and more intuitive to manipulate than triangle meshes.

High-order surface modeling: Semi-regular quad meshes are very useful as base meshes for fitting tensor-product splines or NURBS: one spline patch is defined for each regular patch of the mesh, and different patches are glued together at common boundaries. Coarse quad or quad-dominant meshes (not necessarily semi-regular) can act as control meshes for subdivision surfaces that provide base shapes for arbitrarily complex objects and characters. In general, tensor product patches obtained from quad control meshes are much easier to manipulate than triangle-based Bernstein/Bezier bases. These techniques are important because Splines, NURBS and subdivision surfaces are the modeling techniques that dominate some industrial applications (CAD/CAM for Splines and NURBS, and the entertainment industry for subdivision surfaces).

Texturing: Semi-regular quad meshes are an excellent match for texturing, as each patch can be trivially mapped to a rectangular texture. Normal mapping and displacement mapping techniques, supported in hardware, provide means of adding fine details to such meshes. These advanced texture mapping techniques can also be combined with subdivision surfaces for both offline and real-time rendering in the context of most recent GPU pipelines (DirectX 11 and higher), which include the geometry shaders.

Finite element simulation: Quad meshes are preferred in some numerical analysis, such as finite element modeling within highly elastic and plastic domains, for which they reduce both the approximation error and the number of elements as compared to triangles [SJ08]. In iso-geometric analysis, tensor product function bases are often easier to manipulate than their triangular counterparts and have a better approximation power [D'A00], thanks to the ability of quad meshes to be aligned with the principal directions of curvature.

Compression: The possibility to store a high resolution shape as a coarse quad mesh (or quad-based higher order surface), plus compressed displacement detail on regular grids associated to quads, may greatly reduce storage space and transmission times. Since details are stored in image-like 2D arrays (textures, normal maps, displacement maps), they can be compressed with standard image compression techniques (sometimes adapted to the geometric content).

Note that, in most cases of the above applications, mesh elements should be nearly flat and nearly rectangular, and they should be arranged properly to follow prescribed directions, or to form suitable patches, or both. As we will see in the next sections, these requirements make geometry pro-

cessing on quad meshes much more difficult than on triangle meshes.

2. Characteristics of a Quad Mesh

In order to exploit the desired features outlined in the previous section, a quad mesh should feature a number of characteristics, many of which are strongly related to each other.

Most quad mesh properties are related to the quality of the approximation of the original shape (in the case of remeshing) or the desired shape (in the case of ab initio modeling or deformations), or suitability for a particular task (deformation or simulation).

To estimate the shape approximation quality, symmetric Hausdorff distance is a standard measure (a good approximation of that distance can be computed with the Metro algorithm[CRS98]). Depending on the application, other measures of approximation quality may be equally important. For instance, since the normal affects the shading, measures taking normal deviation into account are important for appearance-preserving remeshing. Such measures are also used in Finite Element Modeling[Mir11].

A set of quad mesh characteristics is related to shape approximation quality.

Some of these characteristics are related to individual quads, other to the quad arrangments. We start with the former.

Quad quality. The properties of general quads make them a more difficult primitive to handle compared to triangles. For example, a triangle is always flat and convex; it supports linear interpolation of functions defined at vertices; it can be easily projected to a plane; and triangle rasterization is straightforward, with all graphics hardware optimized for triangle rasterization. There is effectively two simple characteristics of triangle quality: the deviation from equilateral triangle (measured, e.g., by the inscribed-to-circumscribed radius ratio) and the largest angle.

In contrast, quad quality has many more aspects, and even simple operations present a greater challenge. A quad is not necessarily flat; planar quads may be non-convex; interpolating attributes on a quad requires extending the definition of barycentric coordinates which can be done in different ways (e.g. [MLBD05, Flo03]). Rendering general quads is intrinsically more difficult compared to triangles, and requires a more complex rasterizer, e.g. [HT04].

Ideally, a quad should be as much as possible as a triangle: it should be close to flat (this can be particularly crucial in architectural applications, where quad meshes featuring this property are often categorized as PQ: Planar-Quad); facet corners should be close to 90 degrees; and opposite sides of each quad should have approximately equal length (this can be particularly crucial in contexts like physical simulation), or, for anisotropic approximation, a ratio best for approximation quality.

Quad orientation and size. Several modeling contexts requires the quads to have a certain orientation or change in shape or resolution along the mesh.

Feature and line and principal curvature alignment. Just as in triangle meshes, features lines, when present, should be explicitly represented as edge sequences; for example sharp crease lines in mechanical objects, or lines where some attribute other than normals (e.g. color) varies. More generally, in the presence of cylinder-like regions, alignment of edges with principal curvature directions is beneficial, as this leads to better-shaped (e.g. flatter) quads and improves surface approximation, especially as measured by normal differences. A different type of feature lines are Langer's lines, for animabile human models.

Resolution adaptivity. it is sometimes beneficial to let the tessellation density vary over the surface of the mesh (for example to allow tessellation density adapt to local shape complexity, or to devote denser sampling to more important parts of a surface). In order to allow for spatial transition through different levels of resolution, extra irregular vertices must be included (see Figure 3, left), unless T-junctions are introduced (as in [MPKZ10]). Therefore in many contexts it is desirable to achieve the right trade-off between adaptivity and regularity (see Figure 3, right).

Anisotropy. A different form of resolution adaptivity is achieved by using anisotropic quads. Approximation theory predicts that for a given surface and a given budget of points, the best approximation will be obtained with a mesh that has elements squeezed along the principal direction of curvature [D'A00]. While this type of adaptivity may result in optimal surface approximation, quads close to squares may be preferable if the quad mesh is used, e.g. for isotropic finite-element simulations. At the same time, if the physical problem itself is anisotropic, it requires mesh generation to be guided by a prescribed anisotropy field. For instance, in computational fluid dynamics, it is desired to squeeze the elements in the direction normal to the wing of a plane since the mostly significant physics occur in the limit layer.

Connectivity characteristics. A number of important quad mesh characteristics are related to connectivity. Most of them (e.g. the number of irregular vertices and their placement) are related to geometric approximation properties.

Regularity. In a sense, "unstructured", "valence semi-regular", "semi-regular" and "regular" meshes can be seen as a continuum of cases with an increasing degree of regularity. Depending on the applications, an either lower or higher degree of regularity is required. On the one hand,

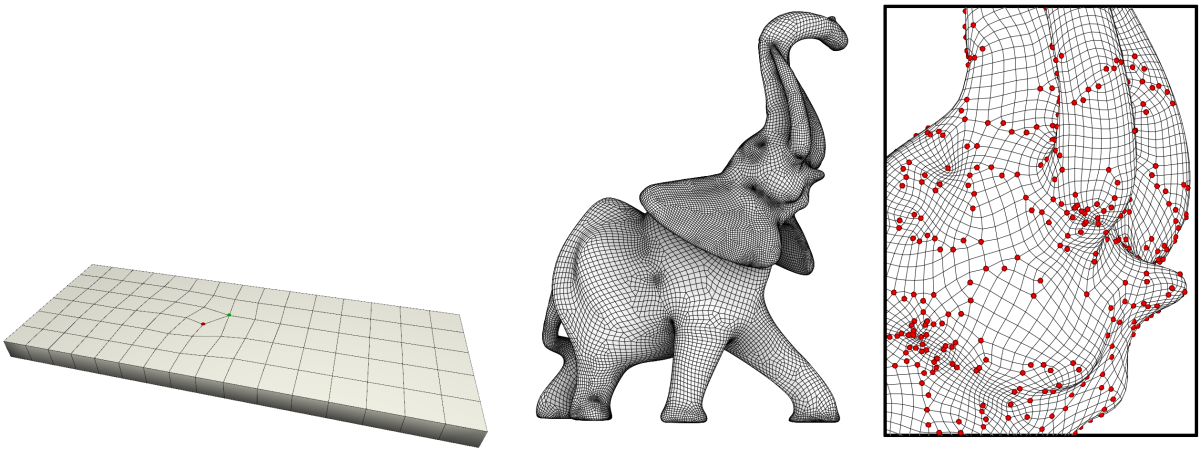


Figure 3: Left: An otherwise semi-regular quad meshing of a bar includes two irregular vertices (red and green) in order to change tessellation density from six quads (left end of the bar) to five (right end of the bar). Right: A quad-mesh with adaptive density. Adaptivity (i.e., quad size adapting to local shape complexity) is obtained at the expense of introducing many singular vertices (in red).

in most applications irregular vertices constitute a complication, hindering usability to some extent. For instance, in subdivision surfaces, they often cause some wrinkles or curvature irregularities, and patches incident to them require special evaluation rules. On the other hand, a certain number of irregular vertices is necessary, depending on shape genus. Irregular vertices also play a role either in improving shape quality (especially for complex surfaces containing high variation of Gauss curvature, caused by bifurcations, protrusions, depressions, handles, etc.), or to let changes in tessellation densities.

Good placement of irregular vertices. Not only the number but also the positioning of irregular vertices is crucial to achieve many objectives in this list. As a rule of thumb, irregular vertices should appear in regions with a strong negative or positive Gaussian curvature (other than where needed to change resolution). An independent desiderata for positioning of irregular vertices is that straight sequences of edges stemming from them (a.k.a. separatrices) should connect them in a graph that is as simple as possible, exhibiting few crossings; if this holds, the quad mesh implies a simpler structure, and it is more likely to be semi-regular (see later in Sec. 4.3.1 for more details).

Other connectivity constraints. in most contexts, quad mesh connectivity must be assumed to be: two-manifold, closed, conforming (i.e., free from T-junctions), and pure (i.e. all polygons are quads). Situations where a vertex has valence 2 (sometime called a *doublet*) or, even more so, 1 (sometime called a *singlet*) are usually considered inconsistent, but they can be accepted in some specific contexts.

3. Quad mesh generation

Many manually created meshes are "born quad". In several other contexts (e.g. reverse engineering, range scanning, iso-surface extraction, etc.) a surface is typically first created in another form, in most cases, a triangle mesh. A naive way to convert any polygonal mesh into a quad mesh is to perform topological Catmull-Clark subdivision. The obtained quad mesh has the desirable property of preserving all original edges, but it also has several important drawbacks: it increases the number of elements (splitting each k-gonal facet into k distinct quads), and it introduces a large number of irregular vertices (one for each original facet that is not a quad). This is a practical option only if the starting mesh is already quad-dominant and a increase in complexity is acceptable. In most cases, more complex approaches are needed.

The task has been approached from different directions. *Quad conversion* (Section 3.1) explicitly targets the problem of converting a triangle mesh into a quad mesh, often working just on connectivity. On the other hand, *quad remeshing* also implies a re-sampling of the original surface. A popular class of quad-remeshing methods consists in parameterizing the original triangle-mesh over a base domain, and then regularly sampling over this domain (Section 3.3). The domain can be created by partitioning the original surface into quadrilateral patches and using the resulting coarse quad mesh as a domain (Section 3.2). Alternatively, a base domain can be defined implicitly by cutting the surface along a set of curves to a disk, and mapping the disk to the plane, subject to certain boundary conditions at cuts that ensure that regular resampling pattern along parametric lines is continued smoothly across cuts (Sections 3.4 and 3.5). Unlike patch-based methods, this class of methods produces valence semi-

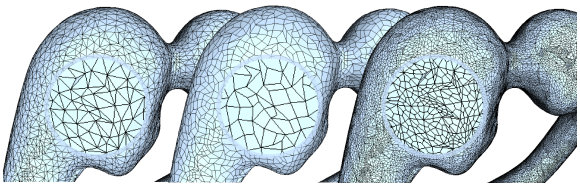


Figure 4: Tri-to-Quad conversion of [TPC*10]: Left: input triangle mesh. Middle: quad mesh built with edge flip operations. Right: quad mesh built with one step of Catmull-Clark subdivision.

regular meshes, but not semi-regular meshes, in the general case. Additional constraints and processing may be required to obtain semi-regular meshes. In the case of field-aligned methods, remeshing can be done by tracing field lines on the surface (*explicit methods* discussed in (3.5)). An independent path to quad remeshing is offered by Centroidal Voronoi diagrams (Section 3.6).

While most techniques start with a manifold triangle mesh, several recent methods bypass the triangle-mesh phase and strive to obtain the quad mesh directly from raw data preceding it, e.g., a point cloud or a set of range scans (Section 3.8).

3.1. Tri-to-quad conversion

This class of methods combines a sequence of local operations on connectivity to convert triangular meshes into quad meshes. One main mechanism is to fuse two original triangles into one quad; so basically the tri-mesh to quad-mesh conversion is based on pairing original adjacent triangles. This class of methods can only be expected to produce unstructured quad meshes, not regular ones. To obtain a pure quad-meshes in this way, the original mesh must have an even number of triangles. This is always the case for closed meshes; otherwise it can be trivially enforced by splitting one border triangle in two. Several of these methods have been presented, often as side results in papers focussing in other parts of related areas:

SQuad [GLLR11] is designed to improve the internal representation of meshes but can be used to define a quad mesh out of a triangle mesh.

BlossomQuad [RLS*11] exploits a *perfect matching* algorithm from combinatorial optimization that provably finds the global optimum, with the best element shapes through a sequence of local operators on the connectivity; on the one hand, this approach offers to solve the triangle pairing problem in a globally optimal way. On the other hand, the algorithm complexity is quadratic with the number of elements. This makes the method time consuming, and mainly suitable to cases where the optimality of the result is more important than the run-time.

In [VZ01] most eligible pairs are first identified, and the resulting quad-dominant intermediate mesh is then subdivided into a pure-quad mesh: first each face of the intermediate mesh is split into triangles by barycentric subdivision; then, pairs of such triangles are merged to form quads by deleting all edges that existed prior to subdivision; overall the number of vertices increasing by less than a factor of 2.

In [TPC*10] a greedy approach is presented where most eligible pairs are first identified, and remaining triangles are brought together by sequences of *edge-flip* operations and fused (an open source implementation is available in MeshLab [CCR08]); the number of vertices is not increased; one example of result is shown in figure 4.

One common problem with this class of approaches is that the quality of results in terms of quad shape is strongly dependent on the input, and it is often low. Similarly to what happens with quad simplification (see Section 4.1), a popular partial countermeasure is tangent space smoothing.

Note that these methods can only produce, in the general case, unstructured meshes, even though each method strives to maximize regularity either implicitly or explicitly.

3.2. Defining patches over the surface

The techniques described in this section work by constructing a 1-to-1 mapping of the original surface onto a set of square patches. Then, the final quad mesh is trivially generated by sampling regularly each patch in parametric space (e.g. by subdivision, or by sampling each quad with a regular grid). Resulting quad mesh is semi-regular by construction.

This type of methods was initially developed for semiregular triangle meshes (most importantly, in [EDD*95, GVSS00, PTC10, KLS03]). Many of these techniques can be adapted to work directly on a quad-based domain. Alternatively, triangle base domains can be converted to quad meshes by simple operations similar to triangle-to-quad mesh conversion. However, this approach introduces angle distortion and more irregular vertices in the final mesh.

Direct patch-based techniques for quad meshes include

Parameterization of Triangle Meshes over Quadrilateral Domains [BMRJ04]: Their technique starts with a *normal-based clustering* that classifies the input into flat regions, and uses a center-based clustering technique to extract the coarse mesh (see Section 3.6 below for more details). The base mesh is computed by a clustering approach that attempts to place extraordinary vertices at regions of high curvature. Further processing continues with cluster refinement and cleanup. Once the clusters are finalized, the boundaries are extracted, and a coarse quad mesh is generated which can be used for parametrization.

Semi-regular, Quadrilateral-only Remeshing from Simplified Base Domains [DISC09b]: This work is inspired

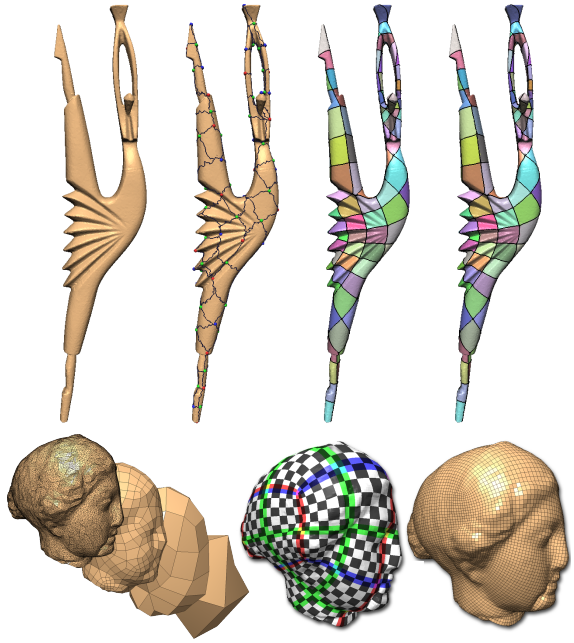
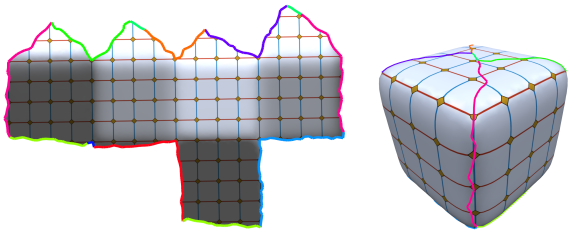


Figure 5: Top: different phases composing [DBG*06]; given a triangle mesh, an initial Morse-Smale complex is extracted, then boundaries are improved thought a sequence of relaxation steps, this finally allows to gather an high quality quadrangulated model. Bottom: The step composing [DISC09b]; a base domain is obtained thought a sequence of simplification operations, original vertices were mapped to the base domain, then sampling the base domain is possible to gather a semiangular quad mesh.

by the well-known MAPS [LSS*98] triangle remeshing technique. First, this technique generates a quad-only model through Catmull-Clark subdivision of the input polygons, then it simplifies it to a base domain that is homeomorphic to the original mesh using a variation of [DISSC08]. During the simplification, a hierarchical mapping method, *keyframe mapping*, stores specific levels-of-detail to guide the mapping of the original vertices to the base domain. The algorithm implements a scheme for refinement with adaptive resampling of the base domain and backward projects to the original surface. As a byproduct of the remeshing scheme, a surface parameterization is associated with the remesh vertices to facilitate subsequent geometric processing, i.e. texture mapping, subdivision surfaces and spline-based modeling.

Polycube-maps [THCM04] are a special sub-class of global quad-based parameterization where the layout of cone singularities is such that the parametric domain has a trivial, axis-aligned embedding in 3D. This is designed to ease texture mapping, but resulting re-sampling have several good properties too: semi-regularity, few irregular vertices, uniform tessellation densities, and well-shaped quads. On the other hand, the task of *automatically* (or semi-automatically) producing this kind of parameterization in a robust way is still an open problem (e.g. see [XGH*11]).



(a) (b)

Figure 6: Parametrization based approaches cut and flatten the surface to the domain (a) such that a trivial quad tessellation of the domain stitches to a valid quad mesh on the surface (b).

3.3. Parametrization based methods

A formulation of the global parametrization problem and constraints needed for quadrangulation shared by both conformal/harmonic and field-guided methods share a similar view of global parametrization. We provide a brief description here and refer to [BZK09] and [TACSD06] to two closely related descriptions. An alternative view based on 1-forms on covering spaces can be found in [KNP07].

We assume that the surface is cut by a set of curves to a topological disk or several disks. The main principle of parametrization based methods is the construction of a mapping from the surface embedded in 3D to a domain in 2D such that the quadrangulation in the domain becomes trivial. Usually the domain is tessellated by a regular tiling like the canonical quad mesh formed by the Cartesian grid of integer isolines. One example is shown in Figure 6 where a smoothed cube is parametrized (a) such that the grid of integer isolines stitches to a quad mesh on the surface (b). The tricky part in this setting is the design of consistency conditions that ensure a correct stitching of the isolines along the cut graph which is essential for non-disk topologies. Continuity of the isolines can be enforced by coupling the position of points $\mathbf{u}$ on one side of a cut to points on the other side $\mathbf{u}'$ by *transition functions* of the form

$$\mathbf{u}' = \mathbf{g}(\mathbf{u}) = R_{90}^{i}\mathbf{u} + (j,k)^T \qquad (1)$$

where $i,j,k \in \mathbb{Z}$ and R_{90} is the 2D matrix performing rotations by 90 degree. These transition functions assure that when traversing the cut graph, although the point position changes from $\mathbf{u}$ to $\mathbf{g}(\mathbf{u})$, the new position is similar w.r.t. the Cartesian grid of integer isolines such that no distortion is introduced by the cut. Near points where several cut curves meet, the regular sampling pattern may be broken even if

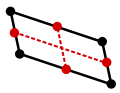
(1) is satisfied for each cut curve. The parametrization with constraints (1) defines a flat metric everywhere on the surface away from these points (in this metric, distances are measured in parametric domain). It turns out that near these points there is also a natural metric, but it is not flat; rather, it corresponds to the metric of a cone. For this reason, these points are called *cones*. Cone angles in this metric are closely related to the valence of irregular vertices appearing at cones after resampling: specifically a cone with cone angle $k\pi/2$ corresponds to a vertex of valence k.

3.4. Conformal and harmonic global parameterization

Locally conformal parametrizations are good candidates for quad mesh generation since they are angle preserving and in particular maintain orthogonality; as a result, small quads in a quadrangulation are close to squares. For an object that is homeomorphic to a disk, a quad mesh can be obtained by contouring the iso-u and iso-v curves of a parameterization [Flo97]. *Harmonic* parametrization can be viewed as as-conformal-as-possible for given boundary constraints. In general, perfect conformality cannot be maintained without severe restrictions on cone angles and without causing significant area distortion, with the exception of objects of spherical topology [SSP08, BCGB08a]. For an object with arbitrary genus, conformal and harmonic parameterization methods admit several generalizations, such as [GY03, SF05, DBG*06, TACSD06]. The method of [GY03] yields discrete conformal parametrizations but restricts the cone angles to be multiples of 2π; and as a consequence, the valence of irregular vertices in the quadrangulation to be multiples of 4, leading to high area distortion of areas of positive Gaussian curvature. [TACSD06] lets the user define the placement of cones and requires connecting them into quadrilateral patches, which allows to infer the rotations and translation in (1). In order to increase the flexibility, this setting can be further generalized to transition functions acting between arbitrarily shaped (non-reactangular) patches as done in [BVK08]. [DBG*06] determines parametrization domain topology starting with a scalar field on the triangle mesh, finding its critical points, which become cones and extracting the Morse-Smale complex of that field. The cells of this complex are quadrilateral, and similarly to manually defined cells of [TACSD06] can be used to infer the rotations and translations in (1). Although in principle, any scalar field should work, since we need a sparse and well-sampled set of quads over the surface, the paper proposes to explore Laplacian eigenfunctions as the scalar field. Such functions locally resemble a product of sines, which forms a mostly regular lattice of bumps and pits. A more recent variant of the algorithm [HZM*08] introduces a degree of directional control.

3.5. Field-guided methods

This class of methods is characterized by explicit control over local properties of quad elements in the mesh by means of the guiding fields. Typically, the most interesting local properties are the orientation and the size of quad elements which can be specified by a *cross field*, also called *frame field*, which smoothly varies over the entire surface. A single cross can be seen as the representative of a parallelogram which is formed by parallel translation of both intersecting lines, as illustrated on the right. For each cross there are essentially four degrees of freedom that can be encoded in different ways. Often a cross field is given in a polar representation where we split the cross into its angular and length components which are then stored in two individual fields, namely an *orientation field* and a *sizing field*. Important subclasses with a reduced number of degrees of freedom (DOF's) are *4-symmetric direction fields* [RVLL08, LJX*10] which represent orthogonal crosses where both orientations are rigidly coupled and *isotropic sizing fields* where both lengths are equal.



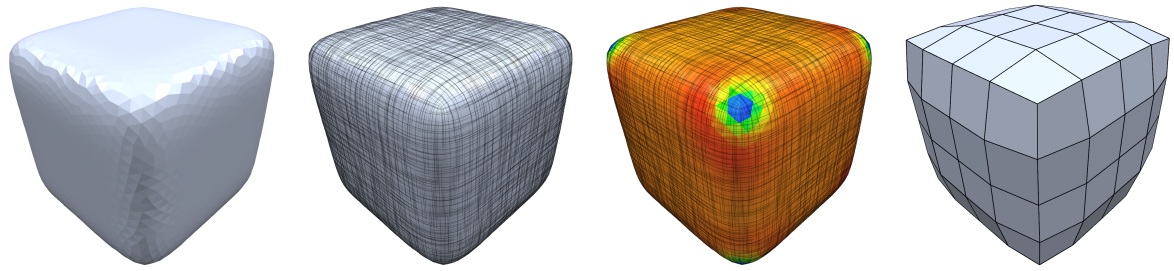
A cross field exhibits the same types of singularities that can be observed in quad meshes and consequently the generation of a highly regular quad mesh is strongly related to the generation of a cross field with few singular points. Depending on the application, a cross field can be either designed manually or generated automatically. Automatic methods are typically driven by principal curvature information which can be shown to optimize the approximation quality [D'A00].

Apart from the pure guidance point of view, note that field guided methods decompose the difficult quad mesh generation problem into several simpler subproblems. This advantage alone motivates their usage since in each sub-step different aspects of the quad mesh can be optimized individually which turns out to be much more tractable than optimizing all aspects simultaneously. A prototypical field guided method is depicted in Figure 7 which consists of three steps:

1. Orientation field generation
2. Sizing field generation
3. Quad mesh synthesis exploiting the results of 1 and 2.

One advantage of field guided methods is that in each step the most suitable data representation can be chosen independently of the other steps. For example, a polar representation is often more powerful for steps 1 and 2 while a vector based representation may be preferred in step 3. The downside of this decomposition is that it is more difficult to integrate direct optimization of quadrangulation quality measures into the choice of cone locations which are determined at step 1. An iteration repeating the steps, and using information from step 3 in step 1 and 2 offers one possible solution.

Many published field guided methods are equipped with their own methodology for the generation of a cross field. However, since the cross field generation is typically independent of the actual quad mesh synthesis, the pool of available algorithms can be seen as the outer product of all orientation field generation methods with all sizing field con-



(a) (b) (c) (d)

Figure 7: Prototype of a field guided method: Given an input triangle mesh (a) in the first step an orientation field (b) is computed which represents the local rotation of quad elements. In the second step a sizing field (c) is determined which specifies the sample density, which in this example is isotropic and close to uniform, with slight deviations color coded from blue to red. In the third step, a consistent quadmesh (d) is generated that closely reproduces both guiding fields.

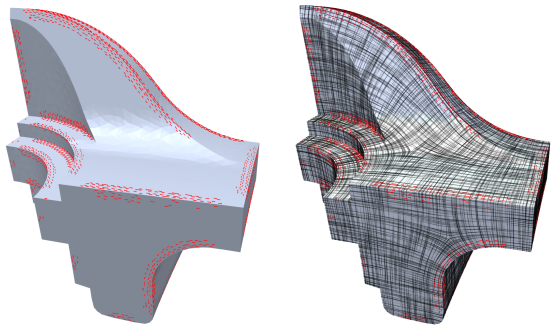


Figure 8: A typical orientation field generation algorithm first identifies the most important orientations (left) which are then smoothly extrapolated over the surface (right).

struction methods with all quad mesh synthesis approaches. Accordingly, the following three sections are devoted to the generation of orientation fields, sizing field construction and quad mesh synthesis, respectively.

3.5.1. Orientation field generation

A typical orientation field generation algorithm computes the smoothest orientation field on the surface subject to some boundary conditions and/or target directions. Often target directions are derived from principal curvature information [CSM03, CP05], so that the most important orientations, i.e. those in parabolic regions where it is obvious how the quads have to be oriented in order to achieve a good surface approximation, are smoothly extrapolated over the surface (see Figure 8). Other frequently used sources of boundary conditions are feature curves, manually painted strokes or user-provided singularity information.

It is important to understand that an orientation field behaves quite differently as compared to a direction field, i.e. a unit vector field. Hence, unfortunately the huge arsenal of vector field design approaches like e.g. [FSDH07, ZMT06] cannot directly be used for the design of orientation fields. In the view of Quadcover [KNP07], an orientation field can be seen as four interlinked direction fields $d_0 \dots d_3$ which are pairwise anti-symmetric, i.e. $d_i = -d_{i+2 \bmod 4}$. However, notice that the space of orientation fields is richer than the combination of four independent direction fields. In an orientation field around a singularity these direction fields can be interlinked in such a way that while traversing a small loop around the singularity the cross rotates by an arbitrary integer multiple of 90 degree, i.e. a jump from one vector field to another is possible. Accordingly, in contrast to direction fields, orientation fields allow for singularities with fractional indices that are integer multiples of $\frac{1}{4}$.

In the literature two different approaches can be found that were developed in order to handle orientation field topologies. The first class of approaches uses nonlinear formulations based on periodic functions like [HZ00, PZ07, RVAL08], while the other class is based on an integer valued representation like, e.g., [RVLL08, BZK09]. In both formulations, finding globally optimal solutions, i.e. the smoothest orientation field subject to some boundary conditions or fitting data, is a hard task. Especially the placement of singularities turns out to be a crucial but also complicated step within the automatic generation of orientation fields. As a result, orientation field optimization algorithms often get stuck in local minima with suboptimal singularities.

To overcome the above problems, several interactive methods were developed which allow the user to either modify or completely specify the singularities of the orientation field [PZ07, RVLL08, LJX*10, RVAL08, CDS10]. Unfortunately, the specification of all singularities is a tedious task which additionally requires expert knowledge in order to achieve good results. Therefore in practice automatic methods like [HZ00, RLL*06, RVAL08, BZK09] are highly desirable.

3.5.2. Sizing field computation

Depending on the application, the sizing field can be computed in different ways. The shape of the elements in the quad mesh can be influenced by the type of the sizing field which can be either isotropic or anisotropic. If squares are preferred, an isotropic sizing function should be chosen, while an anisotropic one offers the possibility to create rectangles by controlling two independent sizing values as was done in [ZHLB10].

The trivial constant sizing field is applied in the context of uniform remeshing where only a constant target edge length is specified. A second possibility is to choose the sizing w.r.t. the curvature in order to achieve a good approximation quality as proposed in [ACSD*03]. A variation of this strategy is to use *lfs* (local feature size), a more global surface characteristic that corresponds to both curvature and local thickness of the surface [AB99].

The third often-used strategy is to compute a sizing field which is compatible with the desired orientation field. To understand the rationale behind this methodology, imagine a cone with a smooth orientation field that diverges from the apex to the base. Clearly a quad mesh which interpolates these orientations, like e.g. a polar parametrization w.r.t. the apex, requires an increasing sizing function in the angular coordinate direction. As observed in [RLL*06], it is feasible to generate a quad mesh that exactly matches a cross field only if the curl of the cross field is zero. Therefore, if precise orientation reproduction is required, it is desirable to compute a sizing field that compensates the directional variations of the cross field by resizing the quads appropriately. Since no solution exists in general, in practice a sizing field that minimizes the curl is computed [RLL*06].

3.5.3. Quad mesh synthesis

Once the guiding fields are available, the next step consists of generating a quad mesh where the individual elements closely follow the local guiding. Here the most difficult aspect is consistency, i.e. finding a pure quad mesh without e.g. triangles or pentagons. Since there typically exists no quad mesh which exactly reproduces the guiding field it is desirable to distribute the required deviation smoothly over the surface instead of concentrating it at some “stitching areas“. Obviously this task requires a global formulation and it turns out that again the placement of irregular vertices is crucial in order to find high quality solutions.

The quad mesh synthesis algorithms can be categorized into *explicit methods* and *global parametrization methods*. The first class of methods tries to explicitly generate curves which on the one hand align to the orientation field and on the other hand exhibit a spacing as desired by the sizing fields. The second class of methods searches for a mapping in a function space which automatically implies consistency and thus leads to a quad mesh that fits as close as possible to the guiding fields. Both approaches are discussed in more detail in the following two paragraphs.

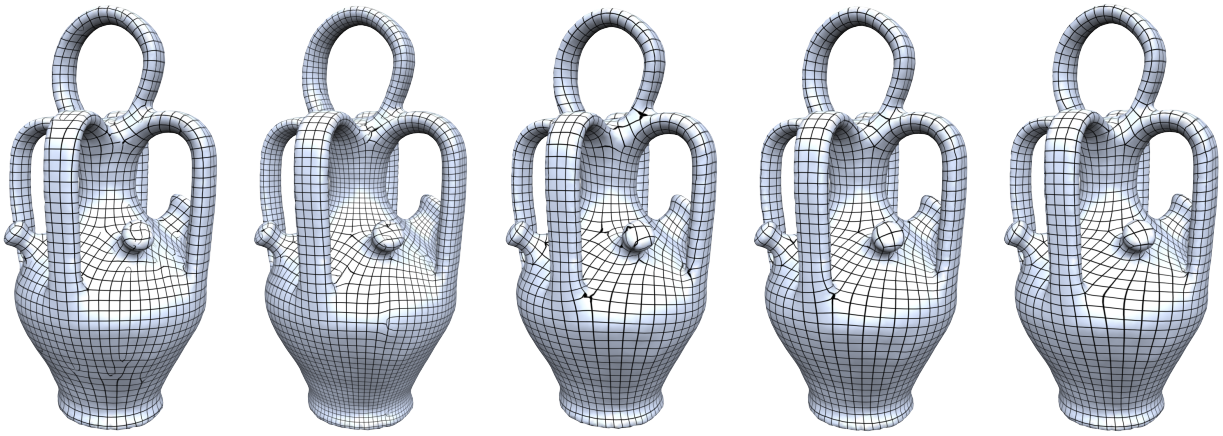
Explicit methods. Streamlines in a cross field intersect orthogonally and naturally form quads as long as no singularities are enclosed. Thus, a straightforward way for synthesizing a quad mesh from guiding fields is to explicitly trace curves within the orientation field as was done in [ACSD*03]. In this approach, the tracing was performed in 2D by means of a parametrization. The difficult part in such an algorithm is to achieve a curve distribution which is in consent with the sizing field. In the aforementioned approach, techniques known from *streamline placement* were adapted in order to achieve adequate results. Later on a variant of this method was developed which does not require any parametrization and thus is applicable to objects with arbitrary genus [MK04].

Even simpler is the approach proposed in [LKH08] where a triangle mesh is successively converted into a quad dominant mesh by first constructing a new triangle mesh with a sampling that is compatible to the sizing field, followed by a smoothing operation which aligns chains of edges with the orientation field. In a final step, non-aligned diagonal edges are removed, similarly to the tri-to-quad conversion approaches described in Section 3.1.

One drawback of all explicit methods is that they are only able to generate quad dominant meshes. Therefore, a final subdivision step is required in order to achieve a pure quadrilateral mesh. A second drawback is that the construction process is performed iteratively without being guided by a global topology. Consequently, these approaches cannot be expected to achieve a globally well-behaved irregular vertex distributions comparable to those produced by parametrization based methods discussed in the next section.

Global parametrization methods. In the context of field guided methods, one be exploited the fact that the rotational degree of freedom i in (1) can be adopted from the orientation field, turning the equation into a linear integer condition. This is exactly the approach taken in [KNP07, BZK09], which as a consequence leads to irregular vertices which coincide with the orientation field singularities. Notice that, in order to capture the rotational changes around singular vertices, it is important to ensure that all singularities of the cross field lie on the cut graph (see Figure 6 and [BZK09]).

By restricting the parametrization function to transition functions of form (1), a function space with "built-in" continuity is designed. Hence, it is possible to search for the mapping within this function space that best reproduces both guiding fields. Unfortunately, due to the integer conditions introduced by the transition functions, the construction of the mapping requires the solution of a mixed-integer problem which in general is NP-hard. Therefore, instead of searching the globally optimal solution, practical methods apply



(a) PGP [8s] (b) PGP+CC [10s] (c) QC [0.5s] (d) MIQ [3s] (e) MIQ+ST5 [15s]

Figure 9: Quad mesh synthesis comparison based on identical guiding fields: The PGP method provides the best length distortion at the cost of additional singularities (a). This effect can be reduced by a curl corrected sizing field (b). QC and MIQ are based on the same function space construction and consequently behave similarly with a clear trade-off between mapping distortion and runtime due to different heuristics for the estimation of integer DOF's (c) and (d). The mapping distortion can be further reduced by the iterative stiffening approach (e).

cheaper heuristics in order to estimate adequate integers from a previously computed continuous solution either in one step [KNP07] or iteratively one after another [BZK09].

Instead of the formulation as a mixed-integer problem, it is possible to express the invariance of grid transformations with the help of (non-linear) periodic functions, as proposed in [RLL*06] (implementation available in Graphite [ALI12]). In terms of the previous representation, this means that the cut graph separates all the triangles of the surface (there is a transition function for each pair of triangles that share an edge). In this formulation, the grid lines stitch correctly only in regular regions while singular regions have to undergo a special treatment including a splitting and a re-parametrization step.

Comparison of parametrization based methods. In order to investigate the behavior of different parametrization based methods, Figure 9 compares the quad mesh synthesis of *Periodic Global Parameterization* (PGP) [RLL*06], *QuadCover-Surface Parameterization using Branched Coverings* (QC) [KNP07] and *Mixed-Integer Quadrangulation* (MIQ) [BZK09] against each other. For all methods the same guidance fields are used which consist in an orientation field produced by the PGP method and a constant sizing field. The only exception is Figure 9 (b) where the sizing field was adjusted by the curl correction method proposed together with the PGP method in [RLL*06].

Figure 9 shows that all synthesis methods behave quite similar in regular regions, showing that the orientation and sizing fields have a strong influence on the result. As mentioned before, a suitable distribution of singularities in the orientation field is crucial for the success of the quad mesh synthesis step. In accordance to that, our experiment shows that interesting differences mostly occur close to singularities which will be the first aspect of our discussion. As expected, the PGP method generates additional singularities in order to capture the given constant sizing field, while QC and MIQ exactly reproduce the orientation field singularities at the cost of some length distortion. Some of the additional singularities can be compensated by a curl corrected sizing function as shown in Figure 9 (b), however, the PGP method does not provide explicit control. Clearly, the favored behavior strongly depends on the application. However, in most practical applications regularity and explicit control over singularities is preferred over moderate length distortion.

QC and MIQ search for an optimal mapping within the same function space and consequently their results shown in (c) and (d) are closely related. While QC is extremely fast since it requires only the solution of two sparse linear systems, MIQ is able to estimate integers that induce less distortion at the cost of increased runtime. The impact of the integer estimation technique strongly depends on how close singular vertices get in the quadmesh. If the goal is the generation of a very coarse quad mesh, it is very important to apply more expensive integer estimation schemes like those of MIQ, while for the generation of finely tessellated quad meshes a simple and fast heuristic like the one of QC is sufficient.

The last aspect we want to analyze here is the quality of the mapping. An often neglected aspect of parametrization based approaches are degeneracies in the mapping function (e.g. foldovers) which easily destroy the quad mesh con-

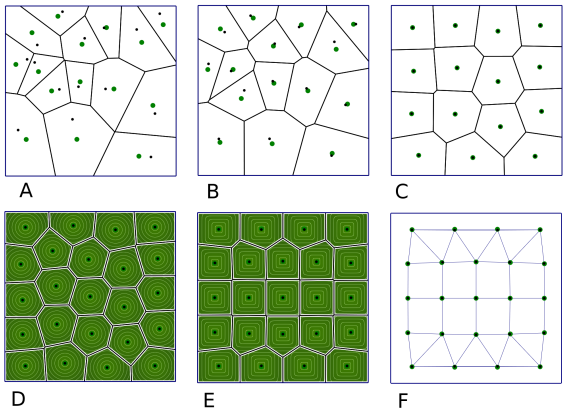


Figure 10: A: random sampling. The samples are shown in black and the centroids in green. B: result after one iteration of Lloyd relaxation. C: convergence of Lloyd relaxation. D: standard CVT (with L_2 norm). E: L_p-CVT. F: quad-dominant mesh obtained from (E) by triangle merging.

sistency and necessitate a repair step comparable to that of PGP. One reason for such defects is that singularities in a parametrization behave similar to point constraints, which are well known to often introduce heavy distortions and foldovers. Although there is currently no fundamental solution to this problem, the *stiffening* heuristic of MIQ in practice often leads to sufficient results by iteratively updating a weighting function in order to minimize the maximal distortion. Figure 9 (e) depicts the solution of MIQ with 5 stiffening iterations where especially the distortion around singularities is greatly reduced, again at the cost of an increased runtime.

In summary, for the quadmesh synthesis algorithms analyzed here, there is clearly a trade-off between speed and quality. While conceptually comparable, in practice the MIQ approach is often preferred over QC since, on the one hand, it naturally handles sharp features and boundaries and, on the other hand, the required greedy mixed-integer solver is freely available [BZK10], enabling a cost-efficient implementation.

While all field guided approaches discussed here lead to valence semi-regular meshes, one interesting direction for future research includes the design of methods that are directly able to generate semi-regular meshes with a coarse patch structure. Additional constraints described in [MPKZ10] offer a step in this direction. Another important aspect which would deserve some attention is the improvement of robustness. While MIQ with stiffening is able to generate valid mappings leading to quad meshes of moderate coarseness, the construction of degeneracy-free mappings for arbitrarily coarse sizing fields is still unsolved.

3.6. Centroidal Voronoi tessellation

In a certain sense, meshing a surface requires to compute a *sampling* of the surface, i.e. generating a set of vertices (or samples) on the surface. In this context, the notion of Centroidal Voronoi Tessellation (CVT) optimizes a measure of the quality of a sampling (see the survey in [DFG99]). From a computational point of view, as suggested by the term *Centroidal Voronoi Tessellation*, a CVT can be obtained by starting from an initial random sampling of the surface (Figure 10-A) and iteratively relocating the samples at the *centroids* of their Voronoi cells[Llo82] (Figure 10-B). This algorithm, known as Lloyd's relaxation, converges to a configuration such that the samples and the centroids of their Voronoi cells coincide (Figure 10-C). Lloyd's relaxation can be used to compute a triangle mesh of a surface with nearly equilateral triangles [DGJ03, AdVDI05].

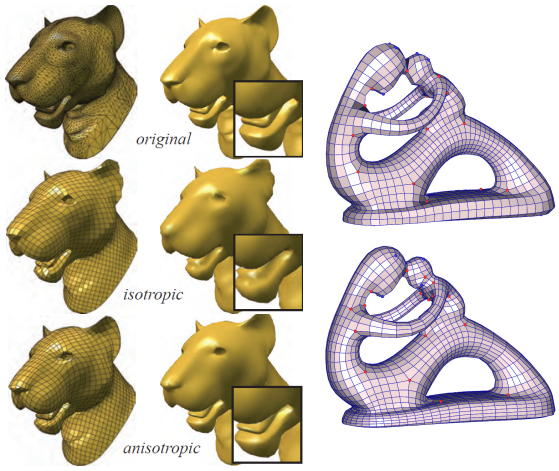
Centroidal Voronoi Tessellations can be used to generate quad (or quad dominant) meshes, both coarse ones, to be used as base domains, as well as fine, quad-dominant, valence semi-regular ones.

Based on the observation that at convergence, the Voronoi cells form a 'honeycomb-like' pattern, with mostly hexagonal cells, a quad mesh can be obtained by splitting each hexagonal Voronoi cell into two quads [BMRJ04]. Another approach is based on a physical analogy between CVT and the configuration of soap bubbles or particles [IS01]. With a specific definition of the attracting and repelling forces between two particles, the algorithm generates a distribution of the samples aligned with a prescribed direction field. A quad-dominant mesh is then obtained by one of the triangle merging techniques in Section 3.1. A similar approach consists in changing the definition of centroids in Lloyd relaxation, in a way that takes the alignment with the direction field into account [Hau01].

Alternatively, it can be shown that Lloyd's relaxation minimizes an objective function, known as the quantization noise power [DEJ06], that corresponds to the inertia moments of the Voronoi cells. The quantization noise power is of class C^2 [LWL*09], and can be minimized by a Newton-type solver[LN89] more efficiently than Lloyd's relaxation. Moreover, considering this variational point of view (CVT defined as the minimizer of F) allows to generalize the definition of CVT. For instance, replacing the L_2 norm (Figure 10-D) with the L_p norm (Figure 10-E) results in a set of (mostly) square Voronoi cells from which a quad-dominant mesh can be extracted (Figure 10-F) by triangle merging.

3.7. Allowing anisotropy

The aims of the above mentioned techniques is to generate squared quads. A complementary approach is to allow for rectangular elements, which may have a significantly better



(a) (b)

Figure 11: (a) The anisotropic remeshing provided by [KMZ10]. (b)Isotropic (top) and Anisotropic (bottom) quadrangulation, using [ZHLB10]

approximation quality. Two recent quadrangulation methods focus on anisotropy:

Anisotropic quadrangulation [KMZ10]: This method for computing anisotropic quadrangulations adapts quad aspect ratios to local curvature to obtain a good surface approximation with fewer quads. This method can be effectively applied to improve the visual quality of a rendering, as illustrated in figure 11.a.

A Wave-based Anisotropic Quadrangulation Method [ZHLB10]: As in Periodic Global Parameterization [RLL*06], the method uses the periodicity of the sine and cosine functions to represent seamless coordinates, combined with the Morse-Smale complex used in Spectral Surface Quadrangulation [DBG*06]. More control is provided to the user. In addition to a cross field, an anisotropic sizing function is used as guidance in order to construct a standing wave which provides a quasi-dual Morse-Smale complex, that approximates the input data. An example showing the degree of anisotropy provided by this method is show in figure 11.b.

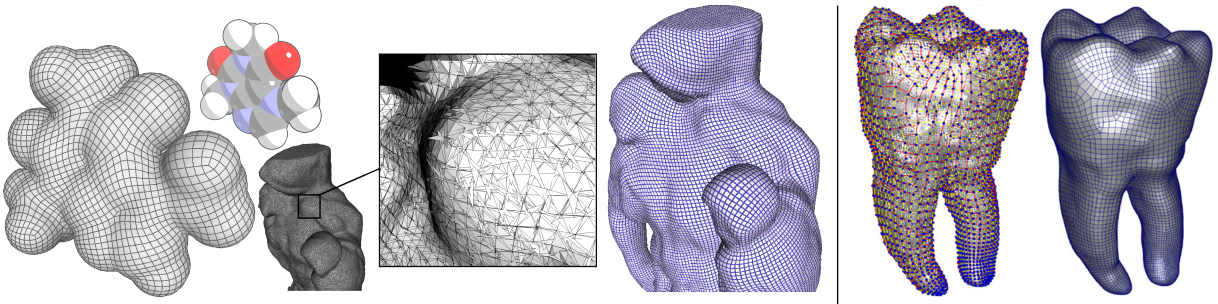
3.8. Starting from different shape representations

All the methods discussed so far require a manifold triangular mesh as input. However, in many cases, the original geometric data has a different form: for example, 3D scanning devices typically produce a set of range images; in other situations, the surface may be defined as a level set of a scalar volumetric function, an implicit function, or a point cloud. A commonly used quad-meshing pipeline starts with constructing a general triangular mesh using a method such as marching cubes, which is then converted to a quad mesh. The ability to work directly with the original representation has several advantages: it reduces the complexity of the meshing pipeline; it eliminates the need to deal with artifacts due to the initial meshes; and, by removing an intermediate resampling step, it makes it possible to incorporate information extracted from original data more directly into quad mesh generation.

Two recent quadrangulation methods use non-mesh input data:

Global parametrization of range image sets [PTSZ11]: This method directly recovers a global parametrization from a set of range images, produced by a range scanner or obtained by projection and sampling of other geometry representations. Range image sets occupy an intermediate place between point clouds or triangle soups, and manifold meshes. On one hand, they exhibit a regular connectivity and implicitly define a global manifold structure for the object, with transition maps determined by reprojection. On the other hand, each point on the surface may be represented by multiple positions inside different range images, and the connectivities of different range images, while highly regular, are inconsistent with each other.
This method is based on a discretization of the seamless global parametrization equations and constraints on a collection of overlapping triangles. In contrast with conventional discretization on a single mesh, the method globally takes into account the compatibility constraints between all the parts. The equations for the parametrization are discretized on each range image separately, with constraints ensuring that the projections of the same surface point to different range images have the same parametric position, up to an admissible transformation.
The most important advantage of using range images is that they can be obtained from any geometry type that can be rendered. Quadrangulation requires only a way to project surface data onto a set of planes, and can be applied directly to implicit surfaces, non-manifold surfaces, very large meshes, and collections of range scans, as shown by Figure 12.a.

Meshless Quadrangulation by Global Parametrization [LLZ*11]: This method generates a quadrangulation computing a global parametrization of an unorganized point cloud. This method is an adaptation of [RLL*06] to point sets, using local vertex neighborhoods consisting of k nearest neighbors. A tangent plane is assigned to each vertex, and the set of neighbors is projected onto this tangent plane and triangulated. The cross field is obtained using the smoothness energy of [HZ00], with connectivity defined by the local triangulations. The parametrization is obtained by minimizing the energy closely related to the energy of [RLL*06]: the formulation is changed, so that the energy terms are defined per-edge rather than per-facet. Quadrangulations produced by this method from point clouds are shown in Figure 12.b.



(a) (b)

Figure 12: (a) Several geometry representations that can be parameterized and quadrangulated by using [PTSZ11] (a set of implicit surfaces and a triangle soup). (b) A point cloud that has been quadrangulated by using [LLZ*11].

4. Quad mesh processing

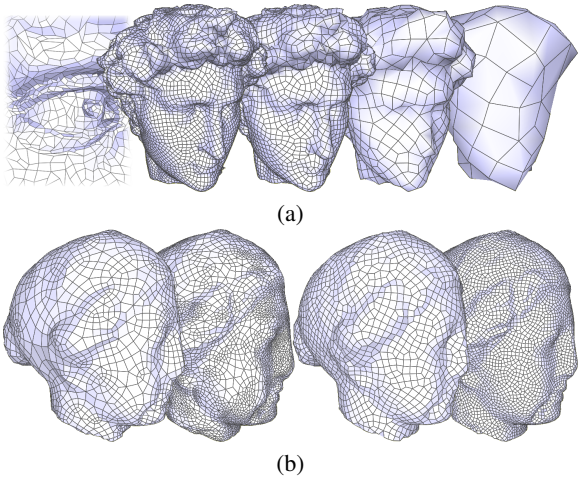
Geometry processing algorithms are techniques that transform geometric representations and shapes (see the book [BKP*10] for a systematic treatment of the subject). They are similar in spirit to digital signal processing, but focus on data representing geometry: most commonly, triangle meshes, point clouds and levelsets of scalar fields.

A significant part of mesh processing research efforts in computer graphics has been traditionally dedicated to triangle meshes. There are all types of efficient techniques for operating on triangle meshes, including smoothing, simplification, compression and parametrization. More recently, a number of efficient algorithms performing similar tasks on quad meshes were developed.

4.1. Quad mesh simplification

Simplification methods aim at reducing the total number of elements forming a given input mesh, keeping the introduced error low and meshing quality high. The typical application is the construction of a (potentially continuous) level-of-detail (LOD) pyramid. Simplification of triangle meshes has been studied in depth during the 1990s and can now be considered a mature technology (see surveys in [CMS97, Lue01]). One typical approach consists of iteratively applying coarsening operations, each one affecting a limited part of the mesh. The order in which operations are performed is crucial; so potential operations must be carefully prioritized. Typically, the best operations are incrementally identified and performed in a greedy fashion, striving to minimize a global *energy*, e.g. quadric error metrics [GH97].

Quad mesh simplifications inherently target only unstructured quad meshes. However, it is useful in a number of situations. One main advantage of the simplification approaches with respect to parametrization based quadrangulation is their ability to produce meshes with fewer quads, e.g., to be used as base domains for either a quad-based parameterization or for higher-order surfaces. The maximal simplification



(a)

(b)

Figure 13: (a) The quad mesh of David can be simplified to obtain very coarse base mesh (b) simplified mesh pairs (5K and 3K) obtained from Igea dataset with [DISC09a] (left) and [TPC*10] (right)

reachable is only limited by topology conditions (see Figure 13.a).

Simplification of quad meshes is inherently more difficult that simplification of triangle meshes due to the stronger global connectivity dependencies. Preservation (or induction) of good meshing quality is a far more challenging and important objective, particularly in terms of element shapes, valence regularity, and field alignment. Also, degenerate configurations are more frequent in quad meshes and avoiding them requires more care. A consequence of these extra difficulties is that minimization of introduced geometrical error can only be pursed to a lesser extent, and it is often sought indirectly.

Another consequence is that quad mesh simplification cannot be considered as mature as triangle mesh simplifi-

cation, as revealed by the size of the meshes which can be tackled. Off-core or streamed triangle mesh simplification techniques (e.g. presented in [ILGS03, CGG*05]) can process on modern computers meshes composed of order of 10^9 elements, and yet newly presented quad mesh simplification approaches are tested by their authors on meshes with less than 10^5 elements (but the gap is reducing).

All methods tend to make the shape of the quads worse. A popular countermeasure, e.g. as proposed among others in [DISSC08, SDW*10, TPC*10, RLS*11], is to apply *tangent space smoothing*, either to the final result, or earlier, during the simplification process. Vertex positions are modified to improve the shape of quads, but without allowing them to leave the mesh surface, and without changing connectivity.

Iterative quad mesh simplification methods presented in literature rely on specific sets of operations. Each operation modifies the mesh connectivity, either to reduce the number of elements composing the mesh, or to improve the individual quad shape. In most frameworks, the application of each single operation must leave a consistent quad connectivity.

4.1.1. Local methods vs Non-Local methods

Quad mesh simplification methods can be classified into local and non-local, depending on the footprint of atomic operations: in local methods, the atomic operations affect only a small neighborhood of a quad or a vertex, while for non-local methods the size of the footprint can be arbitrarily large, and a single operation can affect the entire mesh.

Local methods have the advantage of a finer granularity. Also, they can target only a portion of the mesh, leaving the rest untouched. However they also tend to introduce more irregular vertices, especially when the input mesh is more regular (regularity can actually increase with very irregular meshes, as those obtained by tri-to-quad mesh conversion).

Non-local methods preserve the regularity of the original mesh better. However, since their atomic operations affect a larger part of the mesh, they are intrinsically less adaptive (e.g. it is not possible to reduce tessellation density locally) and granularity of simplification is harder to control.

4.1.2. Non-Local methods

The most common non-local operation is the *poly-chord collapse*, used for example in [DISSC08, MBBM97, BBS02]. A *poly-chord* is a strip of consecutive quads which either self-connects in a loop, or (in open mesh) it has both ends at border edges. If a poly-chord is collapsed, removing all its elements, the quad structure of the mesh is preserved. An attractive property of this operation is that it never introduces novel irregular vertices. However, poly-chords are often too long, winding over the mesh and exhibiting multiple self crossings. Indeed, an entire quad mesh is sometime spanned by a single poly-chord. The collapse of such poly-chords may be avoided either because of consistency constraints, or because it will coarsen the mesh too much.

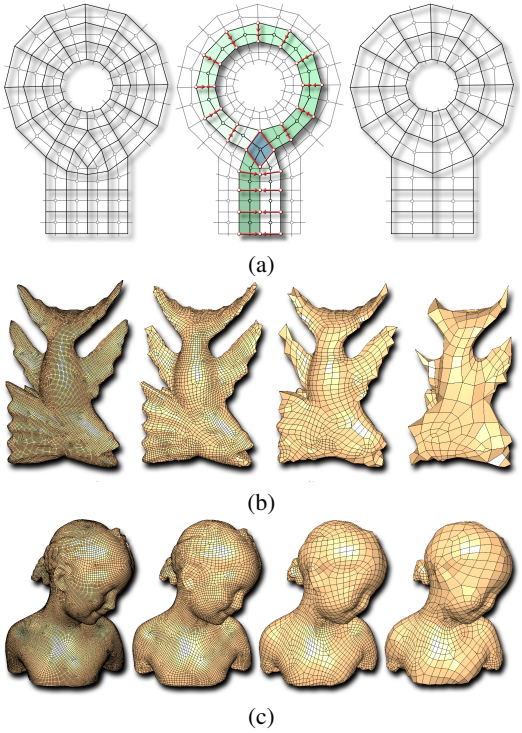


Figure 14: (a) The poly-chord collapse operation. (b) and (c), progressive simplification using [DISSC08].

In [DISSC08], local operations are interleaved to poly-chord collapses to shorten poly-chord prior to collapse. It is significant that, while the minimized energy includes a Quadric Error [GH97] to reflect geometric fidelity, as common with triangle meshes, the weight of this energy competent is kept extremely small compared to the components accounting for meshing quality (like regularity). This reflects the nature of quad mesh decimation, as discussed above.

In [SDW*10] the poly-chord collapses are interleaved by local operations (see below) in order to "steer" the poly-chords around the mesh.

4.1.3. Local methods

Each local method proposed in literature uses its own set of atomic operations, but several operations can be found in all sets, even though they are referred to with different names. We group these operations into three classes:

Coarsening operations, which reduce the number of quad elements composing the mesh.

Optimizing operations, which change local connectivity without affecting the number of elements.

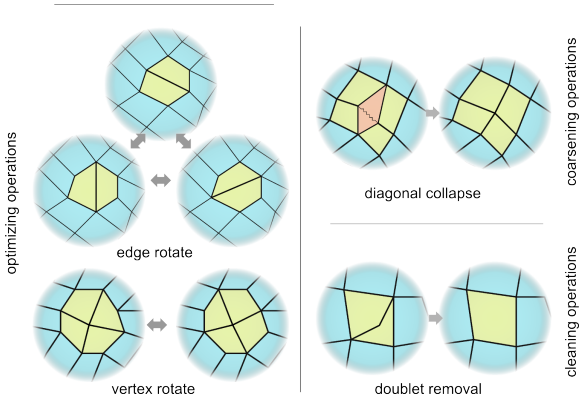


Figure 15: Several common local operations used in the literature: *Edge rotate*: an internal edge, shared by two quads, is dissolved leaving a hexagonal face, which can be split into a pair of quads in other two ways; *Vertex rotate*: the edges incident at an internal vertex v are deleted and they are replaced by the diagonals incident at v of its incident facets. *Diagonal collapse*: a quad is collapsed along a diagonal, merging the two vertices at the end of the collapsing diagonal; *Doublet removal*: a doublet is a configuration with a valency 2 vertex, which is eliminated by dissolving the two shared edges;

Cleaning operations, which address local configurations which are considered degenerate.

Degenerate configurations are for example "doublets" or "singlets", consisting of interior vertices with valency two or one respectively. Note that certain class of applications can allow for these configurations. Figure 15 shows a few commonly used local operations of each class. Methods proposed in literature include:

Localized Quadrilateral Coarsening [DISC09a]: here the poly-coord collapse defined by [DISSC08], is broken into a sequence of simpler atomic operations, each removing one element of the poly-chord. Even if each operation reduces the connectivity quality, a sequence of operations, taken as a whole, inherits the quality preserving properties of poly-chord collapse. The minimized energy function measures the ratio of ideal vertices to total vertices, and a weighted average distance with the original surface.

Practical Quad Mesh Simplification [TPC*10]: In this method, the set of basic operations shown in Figure 15 is extended by adding two new local operations: a coarsening operation called *edge collapse* and a cleaning operation, called *singlet removal*, to erase vertices with valency one. See original paper for further information.
The local operations are used to pursue a simple objective: *homeometry*. Homeometry means equal length of all edges, and proportional length of all diagonals, which, to some extent, implies most other desiderata: flatness, squareness, regularity, isometry, and even some correspondence between vertex valency and Gaussian curvature. It is trivial to prioritize local operations to maximize homeometry in a greedy way.

Automatic Construction of Adaptive Quad-Based Subdivision Surfaces Using Fitmaps [PPT*11]: this method introduces a different way to seek adaptivity: instead of favoring local coarsening operations which minimize introduced error, in a pre-processing step the ideal tessellation density around each point of the surface is estimated *a priori*, stored in a "*fitmap*", and sought during the simplification. A fitmap is designed to target a specific class of shape representations (e.g., either flat quads, or cubic patches). It describes the local tessellation density approximately required to reproduce the original shape, using elements of the targeted class, with any given precision. While the idea is general, it is particularly suited for quad mesh simplification, where, as discussed, the need of ensuring a sufficient meshing quality leaves fewer degrees of freedom to the task of seeking adaptivity.

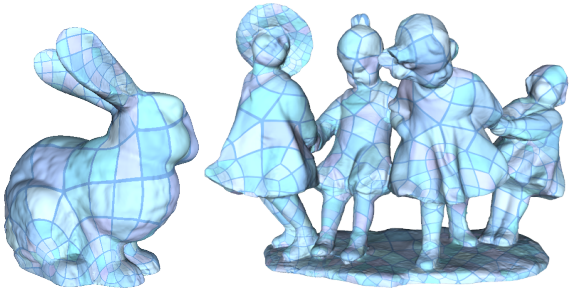


Figure 16: Examples of displaced subdivision surfaces automatically generated by [PPT*11].

A comparisons between [DISC09a] and [TPC*10] is shown in Figure 13: results in [DISC09a] are closer to the original surface, while with [TPC*10] the measure of homeometry keeps the shape of individual quads better and globally uniform. Finally, figure 16 shows the adaptivity provided by [PPT*11] to produce high-order quad-based subdivision surfaces.

4.2. Geometry optimization

The optimization of *geometry* of a quad mesh aims to optimize the shape of its facets, by changing the 3D position of vertices, while leaving connectivity unaffected. It is a fairly straightforward task, usually presented as one step inside more complex methods (as is the case of all papers cited in this subsection).

Available techniques include tangent space smoothing (e.g. used in [RLS*11, TPC*10, DISC09a]), where vertices are allowed to move along tangent directions without leaving the surface, or, almost equivalently, an unconstrained small scale shape optimization followed by a re-projection on the original surface. Other desiderata, like preservation of feature lines, can also be improved by this kind of geometric

optimization, by moving vertices over creases (e.g., as in [TPC*10]).

When the quad mesh is obtained by a re-sampling over a parameterization domain of an initial triangular mesh (as described in Sec. 3.3), an alternative approach to optimize the geometry consists of minimizing the global stretch of the parameterization prior to re-sampling (e.g. as described in [BcGB08b]). This is more robust than tangent space smoothing as optimization is performed in parametric space, thus in 2D, and fitting to the input surface is guaranteed by construction (as opposed to, e.g., 3D re-projection over the surface). In parameterizations defined globally thanks to transition functions (like in [RLL*06, BZK09]), stretch is just minimized globally, by explicitly including the transition functions to enforce continuous isolines across seams (Figure 6.b).

For parameterizations defined over 2D domains composed by the union of a set of patches, optimization takes place inside each patch, but this poses the problem of how to optimize the areas around the cuts separating the patches, which includes the important case of placements of irregular points. This problem can has be tackled in several different ways. In [THCM04] the trivial embedding of the parametric domain in a 3D space allows for a trivial re-projection over the polycube. Such reprojection is interleaved with a global iterative stretch minimization approach. In [KLS03] the positions of irregular points are optimized separately, and this is interleaved with with stretch minimization inside each patch. In [TPP*11] (and similarly in [PTC10]) a "quincux" strategy is adopted where optimization is performed in several stages; at each stage, the parameterization domain is partitioned differently in a set of so-called "parameterization domains", and the stretch energy is minimized inside each domain separately; different partitions are used in succession, designed in such a way that each point of the surface is guaranteed to be mapped in the interior of a parameterization domain in at least one partition; in this way, each point is guaranteed to undergo optimization (including the corners of the original domains, i.e., all irregular points).

4.2.1. Geometry optimization in architectural applications

In the field of Architectural Geometry, quad meshes are often used to represent physical structures to be constructed (e.g. domes). In this scenario, the geometric optimization is used to enforce, within some tolerance, specific geometric constraints [CHP*10]. Often these requirements are not expressible as linear constraints, and non-linear solvers are called for.

One possible constraint is flatness of quad elements (e.g. [GSC*04, YYPM11]), which eases construction. In other cases the geometrical optimization strives to produce quads that are several repeated instances of the same shape, so that that they can be produced using the same mold [EKS*10, FLHCO10].

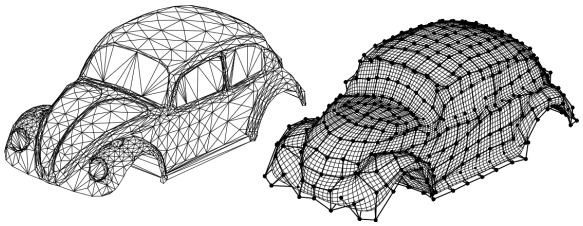


Figure 17: Fitting a Catmull-Clark subdivision surface (right) to a triangle mesh (left) with VSDM (Voronoi Squared Distance Minimization). The control mesh is superimposed (thick black lines).

In [YYPM11] a general unified system is introduced that is able to enforce several different constraints, like planarity of sequence of edges, quad planarity, flatness or *circularity* of facets (i.e., the property of having a specific circumcircle, or positioning of specific vertices). The system is interactive: given a set of constraints, an end user can explore the space of possible shapes satisfying it. The user works inside a solution space of all possible embeddings of the mesh in 3D (a space which has three dimensions for each vertex of the mesh). The key of the interactivity is to let the user move inside "osculants": parameterized manifolds embedded in the solution space, passing through the current embedding, and which matches up to the second order derivatives the user-defined constraints.

4.2.2. Fitting high-order surfaces to a given shape

One of the original goals of algorithms for mesh parameterization was to provide a way to fit a spline surface to a mesh [Flo97]. Many methods for *semi-regular* triangle and quad mesh generation have, as one of the applications, approximation of surfaces with high-order surfaces, including subdivision surfaces, various types of surface splines, and T-splines. Starting with [EDD*95] for triangle meshes, [BMRJ04] aimed at constructing a multiresolution Loop or Catmull-Clark approximation to the original mesh (note that this work uses Centroidal Voronoi Tessellation, see also Section 3.6). These methods typically split the highly nonlinear problem of finding the best-approximating high-order surface for a given mesh, into two parts. The first step, essentially equivalent to semi-regular remeshing, defines a parametric domain (provided by a collection of patches of the semi-regular mesh) and a parametrization of the original mesh over the parametric domain. This reduces the problem to the functional setting: both the surface and the approximating high-order surface basis functions are defined on the same parametric domain. At the second stage, a standard function approximation problem is solved. In a similar way, [GHQ06, HWW*06, WHL*08] use conformal

parametrization to construct several types of spline and T-spline surface approximations. The same process is a part of lossy mesh compression algorithms based on remeshing [KSS00], adapted to quad meshes in [BDHJ04] and other related works.

Valence semi-regular quad meshes are not in general compatible with fitting multiresolution subdivision surfaces and related spline constructions, due to absence of coarse patch structure. However, a coarse *T-mesh* structure [MPKZ10] can always be extracted from parametrizations produced by methods like PGP, QC and MIQ (see Section 3.5.3). Then T-Splines[LRL06] and related T-mesh subdivision surfaces can be used to approximate the mesh.

Another family of methods is inspired by the notion of Polycube Map [THCM04], a generalization of cube mapping, where a given surface is put in one-to-one correspondence with a set of cubes. In the original method, the mapping is constructed via user interaction. Some automatic methods compute the mapping by fitting an initial polycube template to the surface, using relaxation both on the surface and on the template (dual domain relaxation) [YLSL11]. VSDM (Voronoi Squared Distance Minimization, Figure 17) operates similarly, with the difference that it minimizes a well defined objective function that corresponds to a Voronoi-based approximation of the overall distance between the surface and the template [NYL11].

4.3. Connectivity optimization

As we discussed previously, several applications require that the quad mesh is semi-regular and its elements are nearly flat and rectangular. On the other hand, most methods for automatic mesh generation described in the previous sections cannot guarantee that all such requirements are fulfilled. Connectivity optimization is aimed at producing a semi-regular mesh (i.e., made of few gridded patches) from an input mesh by just changing its connectivity. This is a challenging task for quad meshes, since local changes in the structure usually propagate globally across the whole mesh. There exist some automatic methods that transform a valence semi-regular mesh into a semi-regular mesh maintaining the overall alignment of elements; as well as manual methods that provide basic tools to optimize both the placement of singularities and their connectivity.

4.3.1. Separatrices and connectivity graph

At any interior regular vertex, two pairs of *opposite* edges meet. Starting from an irregular vertex, any chosen edge can be followed, reaching another vertex: if that vertex is regular, then the opposite edge can be followed too, thus reaching a third vertex, and so on until an irregular vertex is eventually reached. We term all the (necessarily distinct) edges traversed in this fashion as forming a separatrix (as defined in Sec. 1.1). Note that the same regular vertex can be encountered twice by the same separatrix: in that case we say

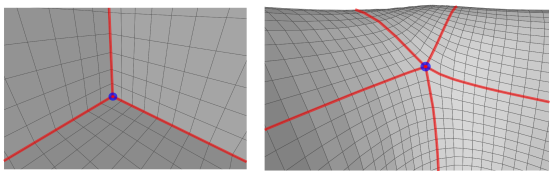


Figure 18: Left: three *separatrices* stemming from a valency 3 irregular vertex. Right: five *separatrices* stemming from a valency 5 irregular vertex. In a close mesh, each separatrix is bound to end in another (or the same) irregular vertex.

that the separatrix crosses itself at that vertex. In general, a regular vertex can be reached by zero, one or two different separatrices. In the latter case, the two separatrices are said to *cross* at that vertex. Conversely, any edge of the mesh belongs to either one or zero separatrices. An irregular vertex of valency n is one endpoint of n (non necessarily distinct) separatrices.

Separatrices are important because they determine the intrinsic connectivity complexity of a given quad mesh: they subdivide the mesh into quad patches. The quad mesh defined by these patches is sometimes termed a "base mesh" because the connectivity of the original mesh can be trivially obtained by a regular re-tessellations of this mesh. The base mesh will have the same irregular vertices of the original mesh, and one regular vertex for each vertex of the original mesh where two separatrices cross.

A semi-regular quad mesh can be defined precisely as a quad-mesh where the base mesh turns out to be coarse. This requires not only that the irregular vertices are few, but also that the separatrices connecting them are short and without too many crossings (see Fig. 1, A). In a general valence semi-regular mesh, this is usually not the case: separatrices may wind over the mesh for long distances, usually with many crossings (see Fig. 19, A). In the worst case, separatrices may traverse all edges of a quad-mesh, making the base-mesh equal to the mesh itself.

4.3.2. Automatic methods

The algorithms proposed in [BLK11, TPP*11] aim at changing the connectivity of the mesh, while maintaining the same singular vertices and the alignment of elements nearly unchanged, with the objective of obtaining a semi-regular quad mesh from an input mesh that is just valence semi-regular. This is done, ultimately, by changing the connectivity of the mesh so that irregular vertices "align" to each other, and separatrices become much shorter (see Fig. 19).

In [BLK11], a generalization of the polychord collapse operation is introduced, which can be used to change the global connectivity of a quad-mesh while maintaining its quad-consistency as well as its irregular vertices. These so called GP-Operators are closed chains of local operations

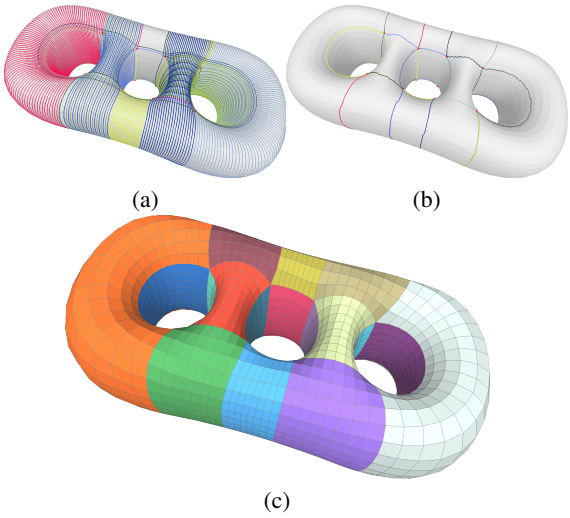


Figure 19: (a) A valence semi-regular quad mesh with just sixteen irregular vertices and an entangled graph of separatrices defining almost eight thousand domains in the base mesh; (b) by using [TPP*11] the graph is disentangled to twenty-eight domains; (c) a semi-regular remeshing obtained by resampling the base mesh.

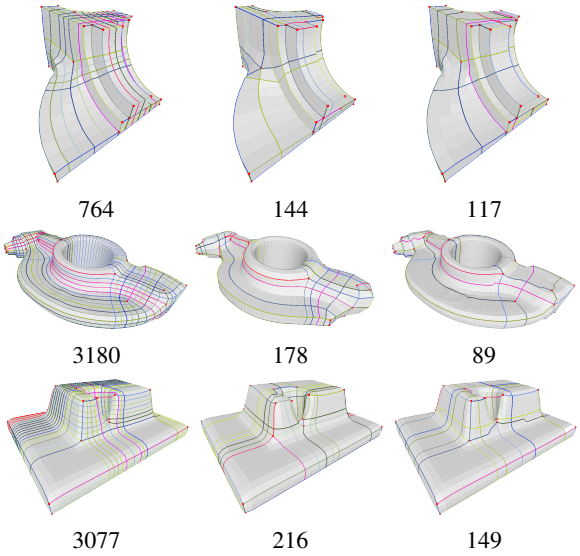


Figure 20: The separatrix simplification process. Starting from the left: the separatrix graph of the input mesh, the separatrix graph extracted from [BLK11],and [TPP*11]. The number in the bottom of every cell represents the number of quadrilateral patches.

acting on mesh edges. Since arbitrary combinations of the three available local operations, namely *collapse*, *shift left* and *shift right*, would easily destroy the mesh consistency, a directed graph is constructed which represents valid GP-Operators as closed loops in the graph. Consequently simple graph search algorithms can be used in order to find the "best" operator which changes the connectivity in a desired way. For clarity we give a simple example here. Assume a single edge shift, i.e. a local connectivity change, is requested. There are many different GP-Operators realizing this local shift, characterized by closed loops in the graph which contain the specific node, encoding the desired edge shift. In this context the "best" choice typically is the shortest loop, inducing the smallest number of additional operations to be performed in order to maintain consistency. In [BLK11] GP-Operators are selected by a simple greedy strategy with the goal of removing helical configurations which are known to yield a dense base mesh. Although experiments show a drastic reduction of the base mesh complexity, the simple helix removal strategy often is not able to generate base meshes that are comparable to manually designed ones.

In [TPP*11] the graph of separatrices is explicitly extracted and simplified with a greedy strategy that consists of a sequence of subatomic moves, without changing the underlying mesh. A step of graph simplification consists of: an initial deletion of a separatrix, which leaves two open *ports* at the singular vertices previously connected by such a separatrix; this deletion is followed by a sequence of moves, each deleting another separatrix and reconnecting two (out of four) open ports with another (new) separatrix; and it is ended by a creation operation that connects the last two open ports, thus bringing back the graph to a consistent state. In the whole process, an energy is considered, which depends on the total length of separatrices and on their *drift,* i.e., their deviation from the cross field underlying the input mesh: the choice of moves is guided from a greedy criterion that tends to minimize such energy. Thus the algorithm aims at disentangling the graph by selecting separatrices that are as short as possible, while maintaining it as aligned as possible to the original cross field. The resulting base mesh is used as a domain for mesh parametrization: a subsequent phase of geometric smoothing (as described in Sec. 4.2) is applied to redistribute the vertices of the original mesh inside the base domains. Finally a new semi-regular mesh is extracted by using such parametrization to regularly sample the facets of the base mesh.

4.3.3. Quad mesh generation for FEM

It is worth mentioning that optimization methods, which were originally developed for the more difficult problem of hexahedral volume meshing (important for some numerical analysis such as plastic deformations of computational fluid dynamics) can be applied to quadrilateral surface meshing. See for instance the survey in [SJ08]. Note that in the context of FEM, the geometry is often available in a form that is different from a triangle mesh, namely a set of Splines

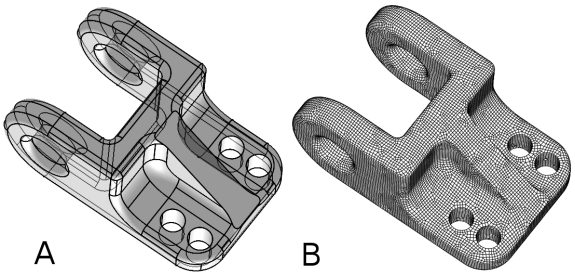


Figure 21: A: a CAD/CAM part, represented by NURBS patches. B: quad-mesh obtained by sampling the surface with Lp-CVT and merging the triangles with Quad-Blossom (data courtesy of C. Geuzaine).

or NURBS, together with all the adjacency information between them (shown as thick black lines in Figure 21-A). Such a meta-graph provides more information about the structure of the object, which can be exploited by a surface quad meshing (or a volumetric hex meshing) algorithm, such as the *multiple sweeping* method [SMKW00]. Another strategy, called *paving*, generates a quad mesh row-by-row. The volumetric version that generates hexahedra is called *plastering*. The *unconstrained* versions of plastering improves the treatment of the center of the object where advancing fronts meet [SOB05].

Note that the additional structuration of the data is not always available, or sometimes shows too complicated relationships to be directly exploited by the algorithm. In this situation, an octree-based method with a special treatment to recover the initial surface works in nearly all situations [Mar09], at the expense of imposing three directions that do not necessarily match the orientation of the features of the surface. L_p-CVT [LL10] (see Section 3.6) is less robust to some degeneracies (features thinner than a quad), but generates quad dominant (or hex dominant) meshes that follow a prescribed direction field. Combined with Blossom-Quad [RLS*11] (see also Section 3.1), it can generate a quad mesh suitable to FEM applications (see Figure 21-B).

4.3.4. Manual connectivity editing

As discussed, good placement of irregular points is crucial to achieve a good meshing quality in a quad mesh. It is part of the skill of any good artistic modelers to be able to identify good placement of irregular points over an intended 3D shape (this is especially stereotyped when targeting certain categories of objects; consider, for example, modeling of human heads, or human bodies). However, in most automatic methods for quad meshing, it is difficult to include the possibility of manual interaction. One reason behind the inability of any fully automatic method to determine good placement, is that the latter is also determined by the intended use of the object (e.g. how it has to be animated) as well as by its shape. Unfortunately, manual editing the connectivity of a quad mesh is not an intuitive task, especially when pure quad meshes are needed. Naïve approaches to manual editing are problematic: a careless local operation on connectivity will create many irregular vertices, repairing which is far from intuitive. The main challenge here is therefore to provide an intuitive interface capable of driving the user in this task, and to find the right abstractions. Recently there has been some significant advancements in this field, which is still overall at an immature stage. We are only starting to orient ourselves in a unexpectedly complex game:

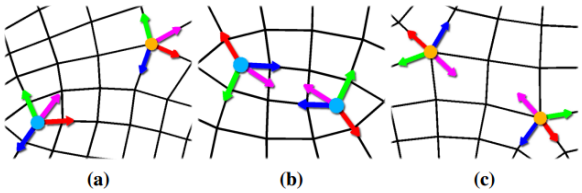


Figure 22: A GUI, offer to an end user several possible ways to change the connectivity around a pair of irregular vertices (a: a valency 3 and a valency 5 vertex; b: a pair of valency-3 vertices; c: a pair of two valency-5 vertices). The arrows approximately point toward the position of irregular vertex after each potential move; all operations will necessarily affect both vertices in predefined ways, are arrows are color-coded to illustrate that to the end-user (image courtesy of C.-H. Peng).

Connectivity Editing for Quadrilateral Meshes [CHPW11]: this introduces a set of operations which affect a pair of irregular vertices at a time, moving them over the mesh but affecting only a small area around them. Interface mechanisms are proposed to show the potential operations to an end user, so that they can edit the connectivity (see Fig. 22).

Interactive Quadrangulation with Reeb Atlases and Connectivity Textures [TIN*ar]: by combining scalar field topology and combinatorial connectivity techniques, this follows a coarse-to-fine design philosophy, which allows for explicit control of the subjective quality criteria on the output quad mesh, at interactive rates. Their quadrangulation framework uses the new notion of *Reeb atlas* editing, to define through user interactions a coarse quadrangulation of the model, capturing the main features of the shape, in particular, they allow the user to prescribed extraordinary vertices and alignment. Fine grain tuning is achieved with the notion of *connectivity texturing*, which allows for additional extraordinary vertices specification and explicit feature alignment, to capture the high-frequency geometries.

5. Conclusions

The state-of-the-art in quad mesh generation and processing was rapidly improving over the past several years, but many

important questions remain unsolved. Since their introduction, semi-regular meshes have potential to change considerably the way surface geometry is routinely stored and processed in many applications, potentially simplifying and improving efficiency of many algorithms. One of the important obstacles on this path is the lack of fully satisfactory algorithms for quad mesh generation, with guarantees on various aspects of the resulting meshes, such as the number of singularities and patches, and the quality of geometry approximation.

One can identify several aspects of the quad mesh generation problem that require further work.

- *Approximation and quad mesh quality.* To make further progress on improving mesh quality, geometry approximation measures need to be integrated more directly into generation algorithms. In particular, singularity placement has a substantial impact on approximation quality and quad mesh quality, and automatic algorithms for singularity placement have to take this into account.
- *Robustness.* Different quad mesh generation approaches suffer from a variety of robustness problems. For example, methods based on global parametrization typically do not guarantee that the parametrization is locally bijective everywhere, resulting in irregular vertices and non-quad faces in the mesh not corresponding to the desired singularities. On the other hand, for simplification-based methods, both parametrization smoothness and quality is often hard to control. Another important aspect is robustness with respect to incomplete, noisy and geometrically inconsistent inputs. The first steps in this direction are discussed in this paper, but further work is needed.
- *Efficiency and interactive control.* Ultimately, best-quality quad remeshing may require taking semantics of the shapes into account, which may require user adjustments, best achieved through direct manipulation of quad meshes (e.g. repositioning of singularities or patch boundaries). This, in turn, requires highly efficient algorithms for modification of quad meshes and related global parametrizations.
- *Scalability.* Semi-regular quad meshes are particularly appealing as a way to represent large-scale geometry of relatively simple topological structure, a common situation for scanned objects. This requires quad remeshing at high resolution of large datasets; current techniques are primarily tested on relatively small data or low mesh resolutions, and further work is needed to handle scans of moderate to large size (millions to billions of points).
- *Mathematical Foundations.* Many of the practical goals outlined above are likely to require a deeper fundamental understanding, that requires to create more connections with other domains of computational sciences, such as approximation theory, applied mathematics and finite elements modeling. Some techniques belonging to these fields, continuously developed since the 1940's, are now mature and advanced. Transferring these techniques into the geometry processing domain is likely to result in important advances.
- *Evaluation.* With the variety of application domains, it will be of paramount importance to be able to compare the relevance and efficiency of different methods, using a scientific approach based on well-defined and objective metrics. For instance, this may include, to name but a few, stability and convergence of Finite Element simulations, faithfulness (Hausdorff distance) for surface approximation, editability and rigging quality for computer animation, and constructibility / structural mechanics evaluation for computer aided architecture.

6. Acknowledgments

Silva has been supported by grants from the National Science Foundation, the Department of Energy, and ExxonMobil. Lévy has been partly supported by the ERC (StG 205693 GOODSHAPE) and by the ANR (MORPHO). Pietroni has been supported by the EC 7thFW NoE "V-Must.Net: Virtual Museum Transnational Network", no. 270404, 2011-2015.

References

[AB99] AMENTA N., BERN M. W.: Surface reconstruction by voronoi filtering. *Discrete & Computational Geometry 22*, 4 (1999), 481–504. 10

[ACSD*03] ALLIEZ P., COHEN-STEINER D., DEVILLERS O., LÉVY B., DESBRUN M.: Anisotropic polygonal remeshing. In *ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), SIGGRAPH '03, ACM, pp. 485–493. 10

[AdVDI05] ALLIEZ P., DE VERDIÈRE É. C., DEVILLERS O., ISENBURG M.: Centroidal Voronoi diagrams for isotropic surface remeshing. *Graphical models 67*, 3 (May 2005), 204–231. 12

[ALI12] ALICE: Graphite: an experimental 3d modeler, 2000-2012. http://alice.loria.fr/software/graphite. 11

[BBS02] BORDEN M., BENZLEY S., SHEPHERD J.: Hexahedral sheet extraction. In *11th International Meshing Roundtable* (September 2002), pp. 147–152. 15

[BCGB08a] BEN-CHEN M., GOTSMAN C., BUNIN G.: Conformal Flattening by Curvature Prescription and Metric Scaling. In *Computer Graphics Forum* (2008), vol. 27, Blackwell Synergy, pp. 449–458. 8

[BcGB08b] BEN-CHEN M., GOTSMAN C., BUNIN G.: Conformal flattening by curvature prescription and metric scaling. *Computer Graphics Forum 27* (2008). 17

[BDHJ04] BERTRAM M., DUCHAINEAU M., HAMANN B., JOY K.: Generalized b-spline subdivision-surface wavelets for geometry compression. *Visualization and Computer Graphics, IEEE Transactions on 10*, 3 (2004), 326–338. 18

[BKP*10] BOTSCH M., KOBBELT L., PAULY M., ALLIEZ P., LÉVY B.: *Polygon Mesh Processing*. AK Peters, 2010. 14

[BLK11] BOMMES D., LEMPFER T., KOBBELT L.: Global structure optimization of quadrilateral meshes. *Comput. Graph. Forum 30*, 2 (2011), 375–384. 18, 19

[BMRJ04] BOIER-MARTIN I., RUSHMEIER H., JIN J.: Parameterization of triangle meshes over quadrilateral domains. In *Eurographics Symposium on Geometry Processing* (Nice, France,

2004), Scopigno R., Zorin D., (Eds.), Eurographics Association, pp. 197–208. 6, 12, 17

[BVK08] BOMMES D., VOSSEMER T., KOBBELT L.: Quadrangular parameterization for reverse engineering. In *Mathematical Methods for Curves and Surfaces* (2008), pp. 55–69. 8

[BZK09] BOMMES D., ZIMMER H., KOBBELT L.: Mixed-integer quadrangulation. *ACM Trans. Graph. 28* (July 2009), 77:1–77:10. 7, 9, 10, 11, 17

[BZK10] BOMMES D., ZIMMER H., KOBBELT L.: Practical mixed-integer optimization for geometry processing. In *Proc. of Mathematical Methods for Curves and Surfaces, to appear* (2010). 12

[CCR08] CIGNONI P., CORSINI M., RANZUGLIA G.: Meshlab: an open-source 3d mesh processing system. *ERCIM News*, 73 (April 2008), 45–46. 6

[CDS10] CRANE K., DESBRUN M., SCHRÖDER P.: Trivial connections on discrete surfaces. *Computer Graphics Forum (SGP) 29*, 5 (2010), 1525–1533. 9

[CGG*05] CIGNONI P., GANOVELLI F., GOBBETTI E., MARTON F., PONCHIO F., SCOPIGNO R.: Batched multi triangulation. In *Proceedings IEEE Visualization* (Conference held in Minneapolis, MI, USA, October 2005), IEEE Computer Society Press, pp. 207–214. 15

[CHP*10] CECCATO C., HESSELGREN P., PAULY M., POTTMANN H., WALLNER J.: *Advances in Architectural Geometry 2010*. Springer, 2010. 17

[CHPW11] CHI-HAN PENG EUGENE ZHANG Y. K., WONKA P.: Connectivity editing for quadrilateral meshes. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA) 30*, 6 (2011). 20

[CMS97] CIGNONI P., MONTANI C., SCOPIGNO R.: A comparison of mesh simplification algorithms. *Computers & Graphics 22* (1997), 37–54. 14

[CP05] CAZALS F., POUGET M.: Estimating Differential Quantities using Polynomial fitting of Osculating Jets. *Computer Aided Geometric Design 22*, 2 (2005), 121–146. 9

[CRS98] CIGNONI P., ROCCHINI C., SCOPIGNO R.: Metro: measuring error on simplified surfaces. *Computer Graphics Forum 17*, 2 (1998), 167–174. 4

[CSM03] COHEN-STEINER D., MORVAN J.-M.: Restricted delaunay triangulations and normal cycle. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry* (2003), pp. 312–321. 9

[D'A00] D'AZEVEDO E. F.: Are bilinear quadrilaterals better than linear triangles? *SIAM Journal on Scientific Computing 22*, 1 (Jan. 2000), 198–217. 3, 4, 8

[DBG*06] DONG S., BREMER P.-T., GARLAND M., PASCUCCI V., HART J. C.: Spectral surface quadrangulation. *ACM Transactions on Graphics 25*, 3 (July 2006), 1057–1066. 7, 8, 13

[DEJ06] DU Q., EMELIANENKO M., JU L.: Convergence of the Lloyd algorithm for computing centroidal Voronoi tessellations. *SIAM Journal on Numerical Analysis 44*, 1 (2006), 102–119. 12

[DFG99] DU, FABER, GUNZBURGER: Centroidal voronoi tessellations: Applications and algorithms. *SIREV: SIAM Review 41* (1999). 12

[DGJ03] DU Q., GUNZBURGER M. D., JU L.: Voronoi-based finite volume methods, optimal voronoi meshes, and pdes on the sphere. *Computer Methods in Applied Mechanics and Engineering 192*, 35-36 (2003), 3933 – 3957. 12

[DISC09a] DANIELS II J., SILVA C. T., COHEN E.: Localized quadrilateral coarsening. In *Proceedings of the Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, Switzerland, 2009), SGP '09, Eurographics Association, pp. 1437–1444. 14, 16

[DISC09b] DANIELS II J., SILVA C. T., COHEN E.: Semi-regular quadrilateral-only remeshing from simplified base domains. *Comput. Graph. Forum 28*, 5 (2009), 1427–1435. 6, 7

[DISSC08] DANIELS II J., SILVA C. T., SHEPHERD J., COHEN E.: Quadrilateral mesh simplification. *ACM Trans. Graph. 27* (December 2008), 148:1–148:9. 7, 15, 16

[EDD*95] ECK M., DEROSE T., DUCHAMP T., HOPPE H., LOUNSBERY M., STUETZLE W.: Multiresolution analysis of arbitrary meshes. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1995), SIGGRAPH '95, ACM, pp. 173–182. 6, 17

[EGKT08] EPPSTEIN D., GOODRICH M. T., KIM E., TAMSTORF R.: Motorcycle graphs: canonical quad mesh partitioning. In *Proceedings of the Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, Switzerland, 2008), SGP '08, Eurographics Association, pp. 1477–1486. 3

[EKS*10] EIGENSATZ M., KILIAN M., SCHIFTNER A., MITRA N. J., POTTMANN H., PAULY M.: Paneling architectural freeform surfaces. *ACM Trans. Graph. 29* (July 2010), 45:1–45:10. 17

[FLHCO10] FU C.-W., LAI C.-F., HE Y., COHEN-OR D.: K-set tilable surfaces. *ACM Trans. Graph. 29* (July 2010), 44:1–44:6. 17

[Flo97] FLOATER M. S.: Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design 14*, 3 (1997), 231–250. 8, 17

[Flo03] FLOATER M. S.: Mean value coordinates. *Comput. Aided Geom. Des. 20* (March 2003), 19–27. 4

[FSDH07] FISHER M., SCHRÖDER P., DESBRUN M., HOPPE H.: Design of tangent vector fields. *ACM TOG 26*, 3 (2007), 56. 9

[GGH02] GU X., GORTLER S. J., HOPPE H.: Geometry images. In *SIGGRAPH conference proceedings* (San Antonio, TX, 21/07/2002 2002), vol. 21 (3), Association for Computing Machinery, pp. 355–361. 2

[GH97] GARLAND M., HECKBERT P. S.: Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., pp. 209–216. 14, 15

[GHQ06] GU X., HE Y., QIN H.: Manifold splines. *Graphical Models 68*, 3 (2006), 237–254. 17

[GLLR11] GURUNG T., LANEY D., LINDSTROM P., ROSSIGNAC J.: Squad: Compact representation for triangle meshes. *Computer Graphics Forum 30*, 2 (2011), 355–364. 6

[GSC*04] GLYMPH J., SHELDEN D., CECCATO C., MUSSEL J., SCHOBER H.: A parametric strategy for free-form glass structures using quadrilateral planar facets. *Automation in Construction 13*, 2 (2004), 187 – 202. <ce:title>Conference of the Association for Computer Aided Design in Architecture</ce:title>. 17

[GVSS00] GUSKOV I., VIDIMČE K., SWELDENS W., SCHRÖDER P.: Normal meshes. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 95–102. 6

[GY03] GU X., YAU S.-T.: Global conformal surface parameterization. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing* (Aire-la-Ville, Switzerland, Switzerland, 2003), SGP '03, Eurographics Association, pp. 127–137. 8

[Hau01] HAUSNER A.: Simulating decorative mosaics. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2001), SIGGRAPH '01, ACM, pp. 573–580. 12

[HT04] HORMANN K., TARINI M.: A quadrilateral rendering primitive, Aug. 2004. 4

[HWW*06] HE Y., WANG K., WANG H., GU X., QIN H.: Manifold t-spline. *Geometric Modeling and Processing-GMP 2006* (2006), 409–422. 17

[HZ00] HERTZMANN A., ZORIN D.: Illustrating smooth surfaces. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 517–526. 9, 13

[HZM*08] HUANG J., ZHANG M., MA J., LIU X., KOBBELT L., BAO H.: Spectral quadrangulation with orientation and alignment control. *ACM Trans. Graph. 27* (December 2008), 147:1–147:9. 8

[ILGS03] ISENBURG M., LINDSTROM P., GUMHOLD S., SNOEYINK J.: Large mesh simplification using processing sequences. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)* (Washington, DC, USA, 2003), VIS '03, IEEE Computer Society, pp. 61–. 15

[IS01] ITOH T., SHIMADA K.: Automatic conversion of triangular meshes into quadrilateral meshes with directionality. *International Journal of CAD/CAM* (2001). 12

[KLS03] KHODAKOVSKY A., LITKE N., SCHRÖDER P.: Globally smooth parameterizations with low distortion. In *ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), SIGGRAPH '03, ACM, pp. 350–357. 6, 17

[KMZ10] KOVACS D., MYLES A., ZORIN D.: Anisotropic quadrangulation. In *Proceedings of the 14th ACM Symposium on Solid and Physical Modeling* (New York, NY, USA, 2010), SPM '10, ACM, pp. 137–146. 13

[KNP07] KÄLBERER F., NIESER M., POLTHIER K.: Quadcover - surface parameterization using branched coverings. *Comput. Graph. Forum 26*, 3 (2007), 375–384. 7, 9, 10, 11

[KSS00] KHODAKOVSKY A., SCHRÖDER P., SWELDENS W.: Progressive geometry compression. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (2000), ACM Press/Addison-Wesley Publishing Co., pp. 271–278. 18

[LJX*10] LAI Y.-K., JIN M., XIE X., HE Y., PALACIOS J., ZHANG E., HU S.-M., GU X.: Metric-driven rosy field design and remeshing. *IEEE Trans. Vis. Comput. Graph.* (2010), 95–108. 8, 9

[LKH08] LAI Y.-K., KOBBELT L., HU S.-M.: An incremental approach to feature aligned quad dominant remeshing. In *SPM '08: Proceedings of the 2008 ACM symposium on Solid and physical modeling* (2008), pp. 137–145. 10

[LL10] LÉVY B., LIU Y.: Lp Centroidal Voronoi Tesselation and its applications. *ACM Transactions on Graphics 29*, 4 (2010). 20

[Llo82] LLOYD S. P.: Least squares quantization in PCM. *IEEE Transactions on Information Theory 28*, 2 (1982), 129–137. 12

[LLZ*11] LI E., LÃČÂĽVY B., ZHANG X., CHE W., DONG W., PAUL J.-C.: Meshless quadrangulation by global parametrization. *Computer and Graphics* (2011). 13, 14

[LN89] LIU D. C., NOCEDAL J.: On the limited memory BFGS method for large scale optimization. *Mathematical Programming: Series A and B 45*, 3 (1989), 503–528. 12

[LRL06] LI W.-C., RAY N., LÉVY B.: Automatic and interactive mesh to t-spline conversion. In *EG/ACM Symposium on Geometry Processing* (2006). 18

[LSS*98] LEE A., SWELDENS W., SCHRODER P., COWSAR L., DOBKIN D.: Maps: Multiresolution adaptive parameterization of surfaces. In *ACM SIGGRAPH 1998* (1998), pp. 95–104. 7

[Lue01] LUEBKE D. P.: A developer's survey of polygonal simplification algorithms. *IEEE COMPUTER GRAPHICS AND APPLICATIONS 21* (2001), 24–35. 14

[LWL*09] LIU Y., WANG W., LÉVY B., SUN F., YAN D.-M., LU L., YANG C.: On centroidal Voronoi tessellation—energy smoothness and fast computation. *ACM Transactions on Graphics 28*, 4 (2009), 1–17. 12

[Mar09] MARÉCHAL L.: Advances in octree-based all-hexahedral mesh generation: handling sharp features. In *18th International Meshing Roundtable conference proceedings* (2009). 20

[MBBM97] MURDOCH P., BENZLEY S., BLACKER T., MITCHELL S.: The spatial twist continuum: A connectivity based method for representing all-hexahedral finite element meshes. *Finite Element in Analysis and Design 28*, 2 (December 1997), 137–149. 15

[Mir11] MIREBEAU J.-M.: Optimally Adapted Meshes for Finite elements of Arbitrary order and W1,p norms. Aug. 2011. 4

[MK04] MARINOV M., KOBBELT L.: Direct anisotropic quad-dominant remeshing. In *Computer Graphics and Applications, 2004. PG 2004. Proceedings. 12th Pacific Conference on* (oct. 2004), pp. 207 – 216. 10

[MLBD05] MEYER M., LEE H., BARR A., DESBRUN M.: Generalized barycentric coordinates on irregular polygons. *Computer 7*, 1 (2005), 0–4. 4

[MPKZ10] MYLES A., PIETRONI N., KOVACS D., ZORIN D.: Feature-aligned t-meshes. *ACM Trans. Graph. 29*, 4 (July 2010), 117:1–117:11. 3, 4, 12, 18

[NYL11] NIVOLIERS V., YAN D., LÃČÂĽVY B.: Fitting polynomial surfaces to triangular meshes with voronoi squared distance minimization. In *International Meshing Roundtable conference proceedings* (2011). 18

[PPT*11] PANOZZO D., PUPPO E., TARINI M., PIETRONI N., CIGNONI P.: Automatic construction of adaptive quad-based subdivision surfaces using fitmaps. *IEEE Transaction on Visualization and Computer Graphics 17*, 10 (october 2011), 1510–1520. http://doi.ieeecomputersociety.org/10.1109/TVCG.2011.28. 16

[PTC10] PIETRONI N., TARINI M., CIGNONI P.: Almost isometric mesh parameterization through abstract domains. *IEEE Transaction on Visualization and Computer Graphics 16*, 4 (July/August 2010), 621–635. 6, 17

[PTSZ11] PIETRONI N., TARINI M., SORKINE O., ZORIN D.: Global parametrization of range image sets. *ACM Transactions on Graphics, Proceedings of SIGGRAPH Asia 2011 30*, 6 (2011). 13, 14

[PZ07] PALACIOS J., ZHANG E.: Rotational symmetry field design on surfaces. *ACM Trans. Graph. 26*, 3 (2007), 55. 9

[RLL*06] RAY N., LI W. C., LÉVY B., SHEFFER A., ALLIEZ P.: Periodic global parameterization. *ACM Trans. Graph. 25* (October 2006), 1460–1485. 9, 10, 11, 13, 17

[RLS*11] REMACLE J.-F., LAMBRECHTS J., SENY B., MARCHANDISE E., JOHNEN A., GEUZAINE C.: Blossom-quad: a non-uniform quadrilateral mesh generator using a minimum cost perfect matching algorithm. *International Journal for Numerical Methods in Engineering* (2011). accepted. 6, 15, 16, 20

[RVAL08] RAY N., VALLET B., ALONSO L., LÉVY B.: *Geometry Aware Direction Field Design*. Tech. rep., INRIA - ALICE Project Team, 2008. Accepted pending revisions. 9

[RVLL08] RAY N., VALLET B., LI W. C., LÉVY B.: N-symmetry direction field design. *ACM Trans. Graph. 27*, 2 (2008), 1–13. 2, 8, 9

[SDW*10] SHEPHERD J. F., DEWEY M. W., WOODBURY A. C., BENZLEY S. E., STATEN M. L., OWEN S. J.: Adaptive mesh coarsening for quadrilateral and hexahedral meshes. *Finite Elements in Analysis and Design 46*, 1-2 (2010), 17 – 32. Mesh Generation - Applications and Adaptation. 15

[SF05] STEINER D., FISCHER A.: Planar parameterization for closed manifold genus-g meshes using any type of positive weights. *Journal of Computing and Information Science in Engineering 5*, 2 (2005), 118–125. 8

[SJ08] SHEPHERD J. F., JOHNSON C. R.: Hexahedral mesh generation constraints. *Engineering with Computers 24*, 3 (2008), 195–213. 3, 19

[SMKW00] SHEPHERD J., MITCHELL S. A., KNUPP P., WHITE D.: Methods for multisweep automation. In *9th Intl. Meshing Roundtable conference proceedings* (2000). 20

[SOB05] STATEN M. L., OWEN S. J., BLACKER T. D.: Unconstrained paving and plastering: A new idea for all hexahedral mesh generation. In *International Meshing Roundtable conference proceedings* (2005). 20

[SSP08] SPRINGBORN B., SCHRÖDER P., PINKALL U.: Conformal equivalence of triangle meshes. *ACM Trans. Graph. 27* (August 2008), 77:1–77:11. 8

[TACSD06] TONG Y., ALLIEZ P., COHEN-STEINER D., DESBRUN M.: Designing quadrangulations with discrete harmonic forms. In *Proceedings of the fourth Eurographics symposium on Geometry processing* (Aire-la-Ville, Switzerland, Switzerland, 2006), SGP '06, Eurographics Association, pp. 201–210. 7, 8

[THCM04] TARINI M., HORMANN K., CIGNONI P., MONTANI C.: Polycube-maps. *ACM Trans. Graph. 23* (Aug. 2004), 853–860. 7, 17, 18

[TIN*ar] TIERNY J., II J. D., NONATO L. G., PASCUCCI V., SILVA C.: Interactive quadrangulation with reeb atlases and connectivity textures,. *IEEE Transactions on Visualization and Computer Graphics* (to appear). 20

[TPC*10] TARINI M., PIETRONI N., CIGNONI P., PANOZZO D., PUPPO E.: Practical quad mesh simplification. *Comput. Graph. Forum 29*, 2 (2010), 407–418. 6, 14, 15, 16, 17

[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). 17, 18, 19

[VZ01] VELHO L., ZORIN D.: 4ÂŰ8 subdivision. *Computer Aided Geometric Design 18*, 5 (2001), 397 – 427. <ce:title>Subdivision Algorithms</ce:title>. 6

[WHL*08] WANG H., HE Y., LI X., GU X., QIN H.: Polycube splines. *Computer-Aided Design 40*, 6 (2008), 721–733. 17

[XGH*11] XIA J., GARCIA I., HE Y., XIN S.-Q., PATOW G.: Editable polycube map for gpu-based subdivision surfaces. In *Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2011), I3D '11, ACM, pp. 151–158. 7

[Yau05] YAU S.-T.: *The Founders of Index Theory: Reminiscences of Atiyah, Bott, Hirzebruch, and Singer*. International Press, 2005. 2

[YLSL11] YEH I.-C., LIN C.-H., SORKINE O., LEE T.-Y.: Template-based 3d model fitting using dual-domain relaxation. *IEEE Trans. Vis. Comput. Graph. 17*, 8 (2011), 1178–1190. 18

[YYPM11] YANG Y.-L., YANG Y.-J., POTTMANN H., MITRA N. J.: Shape space exploration of constrained meshes. *ACM Trans. Graph. 30*, 6 (Dec. 2011), 124:1–124:12. 17

[ZHLB10] ZHANG M., HUANG J., LIU X., BAO H.: A wave-based anisotropic quadrangulation method. *ACM Trans. Graph. 29* (July 2010), 118:1–118:8. 10, 13

[ZMT06] ZHANG E., MISCHAIKOW K., TURK G.: Vector field design on surfaces. *ACM Trans. Graph. 25*, 4 (2006), 1294–1326. 9