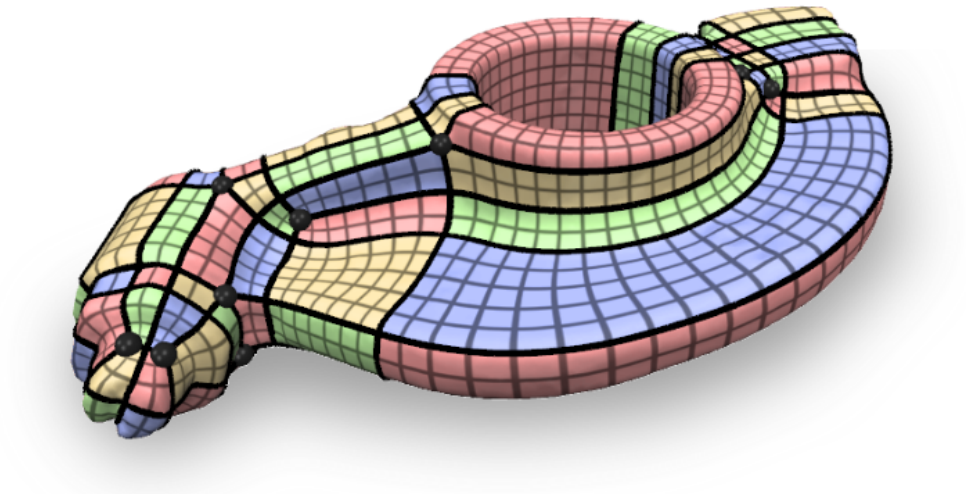
Partitioning Surfaces into Quad Patches

Eurographics 2017 Tutorial



Presenter

Marcel Campen New York University

Abstract

The efficient and practical representation and processing of geometrically or topologically complex shapes often demands a partitioning into patches, each of which is of a simpler nature. Possibilities range from unstructured arrangements of arbitrarily shaped patches on the one end, to highly structured conforming networks of all-quadrilateral patches on the other end of the spectrum. Due to its regularity, this latter extreme of conforming partitions with quadrilateral patches, called *quad layouts*, or in particular instances *quad meshes*, is most beneficial in many application scenarios, for instance enabling the use of tensor-product representations based on NURBS or Bézier patches, grid-based multiresolution techniques, and discrete pixel-based map representations. However, this type of partition is also most complicated to create due to the strict inherent structural restrictions. Traditionally often performed manually in a tedious and demanding process, research in computer graphics and geometry processing has led to a number of computer-assisted, semi-automatic, as well as fully automatic approaches to address this problem more efficiently. This tutorial provides a detailed introduction to this range of methods, treats their strengths and weaknesses, discusses their applicability and practical limitations, and outlines open problems in this field.

Keywords

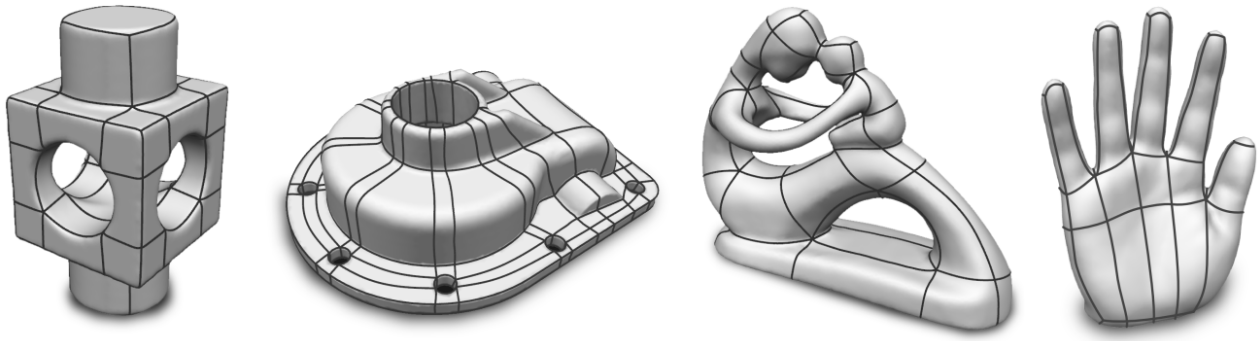
Quad Layouts, Quad Meshes, Surface Parametrization, Tensor-Product Splines

Intended Audience

The course targets researchers and developers in areas involving geometric data, who seek to understand the variety of concepts and techniques that are available to automatically or interactively partition surfaces into quadrilateral patches, and to learn about the most recent developments in this field. Participants will get a comprehensive overview, and obtain the knowledge required to choose the proper combination of techniques for specific tasks where quad layouts are of particular interest.

Prerequisites

The audience ideally should have had some prior exposure to geometric model representations, such as triangle meshes and spline patches, and have a basic working knowledge of general computer graphics fundamentals. Some familiarity with the basics of differential geometry are helpful, but not required.



Some exemplary surface partitions (quad layouts). Patch boundaries are visualized in black. Notice that each patch is four-sided, and patches are conforming (there are no T-junctions).

Background

The tutorial emerged in part from the overview and state-of-the-art discussion in the presenter's thesis on "Quad Layouts" that was awarded a Eurographics Best PhD Thesis Award. An accompanying extended state-of-the-art report is about to appear in Computer Graphics Forum.

Presenter

Marcel Campen is a postdoctoral researcher at the Courant Institute of Mathematical Sciences (New York University), working with Denis Zorin and Claudio Silva. He received his PhD in 2014 from RWTH Aachen University, where he worked with Leif Kobbelt. Marcel was awarded a EUROGRAPHICS Best PhD Thesis Award for his work on quad layouts.

Outline

The tutorial starts with an overview over the variety of areas where quad layouts and quad partitioning techniques have found application over the past decades. The piecewise representation of surfaces by means of tensor-product constructions (Splines, NURBS, Bézier patches) or semi-regular quad meshes (multiblock grids) and the piecewise representation of surface maps and parametrizations via rectangular charts are major use cases necessitating a conforming quad layout.

The essential distinction between quad layouts and quad meshes is discussed, and terminology and taxonomy is addressed to clarify the semantics of the many pertinent terms that are often used in an inconsistent manner in this context, such as mesh, T-mesh, layout, grid, network, multiblock, semi-regular, base mesh, base complex, partition.

The notion of a quad layout, or a conforming partition into quadrilateral patches, is formalized, its properties discussed, and common quality criteria are treated. For many use cases these involve geometric fidelity, structural simplicity, and some form of shape-aware orientation and alignment of the layout elements. The often conflicting nature of these objectives is exemplified, exposing the trade-off and balancing that is typically required when designing or generating a quad layout.

After this introduction, a detailed treatment of the various algorithmic options for the automatic or interactive generation and design of quad layouts follows. It is structured based on the following observation: The problem of quad layout generation (partitioning a surface into conforming quadrilateral patches) has discrete, combinatorial, and continuous degrees of freedom. These can loosely be attributed to the nodes, arcs, and patches of the layout, respectively. Many recent techniques take care of these degrees of freedom sequentially: first the nodes (in particular the extraordinary layout vertices) are determined, then the arcs, forming a suitable layout connectivity, are established, and finally the embedding of the layout, in particular of its patches, is optimized. For each step various approaches have been proposed. These can often be combined quite flexibly, though some techniques take care of multiple of these steps in an integrated manner, or require additional information from, or a tight coupling with, other steps.

In terms of node determination techniques, sampling-based strategies (uniform, non-uniform, isotropic, anisotropic) as well as specialized curvature-driven strategies (Gaussian curvature quantization, singularity extraction from cross fields and trivial connections) are treated.

The determination of a suitable connectivity is probably the most involved and most interesting aspect. There is a wide variety of techniques. Some are based on modifying a triangular layout (which can be obtained significantly easier due to simpliciality) into a quadrilateral layout, e.g. via subdivision, merging, or hybrid algorithms. Others formulate the problem directly as a binary programming problem in a straightforward manner (selecting arcs from a huge candidate set such that they form a conforming layout) and make use of clever heuristics to reduce the size of the problem to a tractable level. Another class of methods operates in a dual setting, based on graph duality, where crossing loops, that can be generated in a number of shape-aware manners, imply layouts that are automatically conforming and quad-only. Using integer programming techniques, a valid layout connectivity can furthermore be derived from parametric lattices. These options are introduced systematically with a focus on technical intricacies, and their respective advantages, disadvantages, and limitations are discussed.

In terms of approaches for the continuous, geometric optimization of the layout's embedding in the surface (determining the patch boundaries and node positions in all detail), isolated, arc-based techniques can be used, local or global patch parametrization methods can be employed, or mesh smoothing based methods can be made use of.

Besides these methods targeted at generic quad layouts, a number of methods focusing on restricted classes of layouts are available. These, for instance, derive quad partitions from shape skeletons, Reeb graphs, or volumetric maps, or yield more general non-conforming layouts (with T-junctions) that are of interest for certain use cases. An overview over these methods and their potential benefits (but also their limitations caused by the restriction) is given. Also, techniques that allow for efficient manual influence on a (semi-automatic) partitioning process, striking a balance between manual approaches that put all the burden on the user and fully automatic approaches that provide no or little control, are covered.

The tutorial concludes with a discussion of the future directions and open problems in this area of research – to indicate to developers the limits and boundaries of what is possible with the state-of-the-art today, and to outline to researchers where future efforts could be particularly impactful. Major issues are the strict guaranteeing of quality/optimality for concrete use cases and the generalization to higher dimensions, in particular for hexahedral layouts for solids.

The following is a manuscript of a state-of-the-art report on the tutorial's topic to appear in Computer Graphics Forum.

It serves as a comprehensive reference for all the techniques covered in the tutorial.

Partitioning Surfaces into Quadrilateral Patches: A Survey

Marcel Campen

New York University

Abstract

The efficient and practical representation and processing of geometrically or topologically complex shapes often demands a partitioning into simpler patches. Possibilities range from unstructured arrangements of arbitrarily shaped patches on the one end, to highly structured conforming networks of all-quadrilateral patches on the other end of the spectrum. Due to its regularity, this latter extreme of conforming partitions with quadrilateral patches, called quad layouts, is most beneficial in many application scenarios, for instance enabling the use of tensor-product representations based on NURBS or Bézier patches, grid-based multiresolution techniques, and discrete pixel-based map representations. However, this type of partition is also most complicated to create due to the strict inherent structural restrictions. Traditionally often performed manually in a tedious and demanding process, research in computer graphics and geometry processing has led to a number of computer-assisted, semi-automatic, as well as fully automatic approaches to address this problem more efficiently. This survey provides a detailed discussion of this range of methods, treats their strengths and weaknesses, and outlines open problems in this field of research.

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

1. Introduction

While primitive geometric objects, like spheres, cubes, cylinders, etc., can often be represented using simple mathematical expressions, objects with more complex geometry and in particular more complex topology are typically not amenable to such simple representation in a practical way. This led to the development of *piecewise* surface representations, where each piece of a partition of a complex surface can be represented in a simple way. Analogously, this applies to maps from or onto complex surfaces, which are handled more easily in a piecewise, chart-based manner.

Of particular interest are partitions of surfaces into patches which are four-sided and conforming (cf. Figure 3), sometimes called *quad layouts* (cf. Figure 1). This particular structure, for instance, enables the use of popular tensor-product representations based on NURBS or Bézier patches, efficient grid-based multiresolution techniques, or discrete pixel-based map representations.

It is important to make a distinction between a *quad layout* and what is often called a *quad mesh*. On a structural level, these two concepts are equivalent, but they follow different geometric objectives. Most importantly, the vertices of a quad mesh are often expected to sufficiently encode the geometric information via their spatial positions, implying the full surface geometry or an appropriate approximation thereof via interpolation or subdivision. A quad layout, on the other hand, is supposed to provide a partition of the surface, on top of which additional information can be encoded, for instance to represent the geometry (e.g. via splines) or a surface map (e.g. via rectangular chart parametrization). This conceptual difference necessitates different strategies for the generation of quad meshes (cf. the broad survey [BLP*13]) and quad layouts: while a quad mesh basically resamples a surface and its quality can be assessed locally, in the case of a quad layout the structural, topological aspect from a global point of view plays an important role.

This survey is concerned with methods that create such quad layouts for arbitrary surfaces. It is based in parts on the prelude and literature review in the thesis "Quad Layouts: Generation and Optimization of Conforming Quadrilateral Surface Partitions" [Cam14]. It includes extended treatments and updates on the most recent advances.

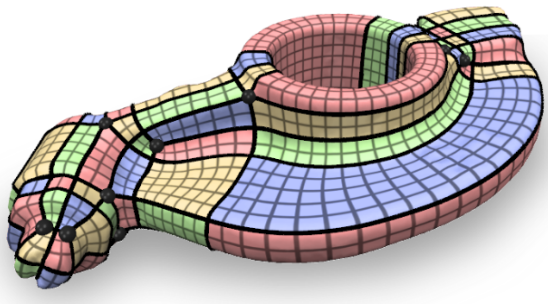


Figure 1: *A surface partitioned into quadrilateral patches (colored individually), with an iso-parameter grid visualization of rectangular parameterizations of the individual patches.*

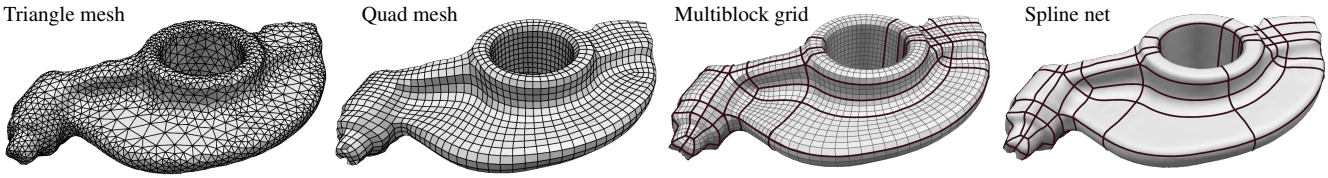


Figure 2: *Illustration of different kinds of* piecewise surface representations*. In the case of the triangle mesh the pieces are trilateral, in the other three cases quadrilateral.*

Outline

We begin by outlining the major reasons for the particular interest in conforming quad layouts in Section 2. In Section 3 the technical definitions, properties, and context needed for precise discussion of the topic are introduced. Then, methods for the automatic construction of quad layouts on given surfaces are surveyed: in Section 4 we focus on the structural, combinatorial aspect of the problem, in Section 5 on the geometrical aspect. Possibilities for manual, semi-automatic, and interactive quad layout design are discussed in Section 6, and methods targeting special classes of quad layouts are covered in Section 7. Finally, in Section 8 we outline some of the major open problems and interesting future directions in the area of quad layouts.

2. Background

We start by outlining some of the reasons for the particular interest in conforming quad layouts, as well as in their automatic generation, from technical and practical points of view.

2.1. Piecewise Surfaces

A very common and particularly simple instance of a piecewise surface representation is the triangle mesh (cf. Figure 2), typically representing a surface in a piecewise linear manner, though higher-order elements are common as well in simulation and analysis.

Analogously, a quad mesh represents and discretizes a surface using four-sided elements. Of particular interest are *semi-regular* quad meshes (also known as *multiblock grids* [SB96]), which contain an underlying coarse quadrilateral base structure (cf. Figure 2), i.e. which are a regular refinement of a quad layout. These provide advantages for various application cases, as detailed in a recent survey [BLP*13], where they are deemed "the most important class [of quad meshes] in terms of applications". For instance, they enable the application of efficient adaptive and multi-level solver schemes [BDL10, DHM09] in the context of quad-based Finite Element simulation and the application of degree adaptation techniques in the context of Isogeometric Analysis [HCB05, Bom12]. Their high level of structuredness is of benefit for applications like mesh compression [AG03] and Fourier or Wavelet based processing [AUGA08]. In the field of character animation designers are interested in quad meshes with good *edge-flow* [JLW10] (a concept closely related to a geometry-aware, simple base structure) as these tend to reduce artifacts and distortions during deformation.

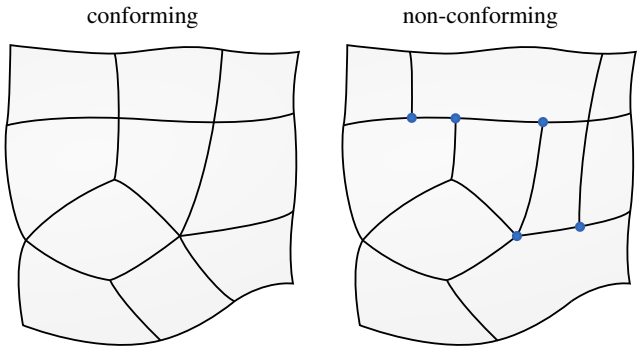


Figure 3: *Illustration of conforming and non-conforming partitioning of a domain into quadrilateral patches. A non-conforming layout may contain T-junctions (blue).*

Just like these semi-regular meshes or multiblock grids require an underlying conforming quad layout, the representation of surfaces using Bézier patches, NURBS, or other types of smooth surface constructions [Far02], often requires a quadrilateral partition to serve as a parametric domain specification.

Building such piecewise representations for an object's surface obviously involves two aspects: suitably partitioning the surface into pieces and finding a suitable representation for the geometry within each piece. For the case of triangle meshes, this is a well-researched, well-understood problem [AUGA08]. Also lot of research has been devoted to the automatic generation of quad meshes and significant advances have been made [BLP*13]. For the other cases, however, while methods for determining a suitable representation per piece are available (e.g. on the basis of grid fitting or interval assignment [TA93, Mit00, BVK08], spline fitting [EH96, KL96, MK95, MBVW95, AAB*88], or subdivision surface fitting [LLS01]), the generation of appropriate quad layouts of high quality in the first place proved to be a hard problem and has a long history of tedious manual efforts in practice. In this survey we discuss early automatic approaches to this problem and the significant advances that have been made in recent years.

It bears noting that there are further types of piecewise surface representation which do not rely on a quad layout, but on simpler-to-construct partitions with triangular or polygonal patches. Bézier triangles and certain types of box splines are examples of such representations. Their popularity in practice and support in software tools, however, is significantly lower compared to representations based on quadrilateral partitions. There are multiple reasons for this circumstance. One can argue from a historic point of view that in

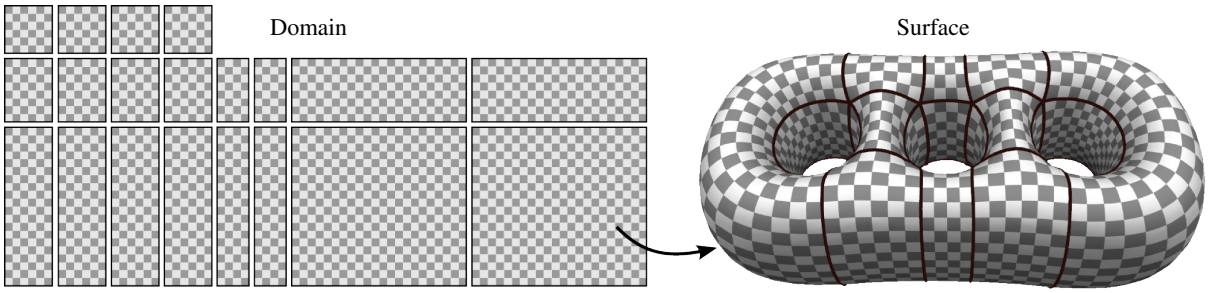


Figure 4: *Illustration of a* piecewise map *from a planar domain onto a surface using a quad layout based chart atlas.*

the early days of computer-aided geometric design in the automotive and aircraft industry tensor-product constructions already set the standard [Far02]. But there are also solid theoretical reasons for preferring quad structures over triangular structures: for instance, the circumstance that the boundaries of quad layout patches can very naturally be aligned to principal curvature directions [dC76], with aesthetic as well as practical advantages [BLP*13].

Also several subdivision surface schemes are able to operate on layouts with arbitrary polygonal patches, but quadrilateral patches are advantageous for the smoothness of the resulting surfaces [Rei95] for prominent schemes [CC78, DS78].

2.2. Piecewise Maps

Not only for the representation of a surface itself, also for the representation of maps from or onto the surface (i.e. parameterizations, texture maps, displacement maps, etc.) is a quad layout of great value. When dealing with surfaces of three-dimensional objects, one most often deals with 2-manifold surfaces (with or without boundaries), i.e. they are locally homeomorphic to a disk, thus amenable to (local) parameterization over $\mathbb{R}^2$. The correspondences provided by such maps for instance allow to apply simpler 2D operations to the surface, even though it is embedded in $\mathbb{R}^3$.

For such use cases, however, often a *global* parameterization is necessary, for which the whole surface needs to be homeomorphic to a disc. Hence, the surface needs to be cut. While a minimal *cut graph* [EHP02] is topologically sufficient, the resulting metric distortion in the map can be a problem for many applications.

Therefore, in practice often a chart atlas is used: the surface is cut (i.e. partitioned) into multiple charts which can be parameterized with low metric distortion. For various reasons (domain simplicity, transition simplicity, discretization and storage, continuity across chart borders [RNLL10, MZ12]) it is advantageous if these charts are quads and these quads are conforming, i.e. again one is interested in quad layouts. Figure 4 illustrates a quad layout based piecewise map from part of the plane onto the surface.

2.3. Layout Generation

A major source of motivation for investigation into the automation of the process of quad layout generation has come from the fact that the creation of quadrilateral partitions, for structured quad meshes, multiblock grids, spline networks, quadrilateral subdivision base meshes, etc., can be an extremely time-consuming task in practice [HCB05, LRL06].

From a technical point of view this process of *quad layouting*, i.e. the partitioning of a surface into conforming quadrilateral patches, involves determining the layout's combinatorial structure as well as its geometric embedding in the surface. The embedding describes the locations of the layout patches' corners and borders on the surface, and possibly parameterizations of the patches. Figure 5 shows an example quad layout.

In industrial workflows this layouting is often performed manually by skilled professionals in the animation and engineering sectors through the construction of nets of surface curves. An inherent issue, and major cause of the complexity of the task, is that a good quad layout generally is a compromise, balancing layout simplicity, patch rectangularity, feature and principal direction alignment, and possibly further objectives (cf. Section 3.4).

Ultimately, the hardness of the problem of automatic quad layout generation mainly stems from two facts:

1. the notion of quality of a quad layout is complex, application-dependent, sometimes fuzzy or hard to formalize,
2. the optimization problem for the generation of high quality quad layouts is of mixed nature: it has continuous, discrete, and combinatorial or topological degrees of freedom, and these have global interdependencies.

For instance, the following questions, concerning degrees of freedom of varying type, must be answered to find a solution to a quad layout generation task:

- **Discrete:** How many nodes (patch corners) to use?
- **Combinatorial:** Which nodes to connect in which ways?
- **Continuous:** How to embed the layout on the surface?

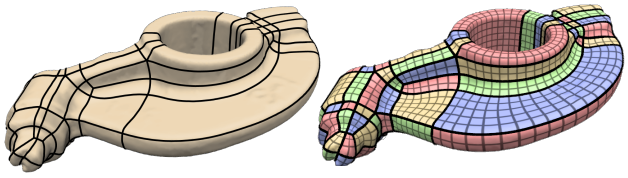


Figure 5: *A quad layout on a surface (left) and an iso-parameter line visualization of the parameterizations which embed its patches (right).*

Due to this heterogeneous nature of the quad layout generation problem, it is hard to tackle using standard optimization techniques ("black box solvers"). Hence, numerous special purpose strategies have been developed for the individual aspects of this problem. We survey those that tackle the combinatorial, structural aspects of a quad layout in Section 4, and those that tackle the geometrical embedding aspects in Section 5.

3. Foundations

We are going to deal with quad layouts on orientable two-dimensional manifolds $\mathcal{M}$ (with or without boundary $\partial\mathcal{M}$), henceforth called surfaces for simplicity.

3.1. Layout Graphs and Embeddings

The notion of a quad layout, a conforming quadrilateral surface partition, is formalized using the following definitions:

Definition 3.1 ***(Layout graph)*** A multigraph $G = (N, A)$ with nodes N and arcs A, which may contain multiple arcs between pairs of nodes as well as dangling arcs which are incident to only one node, defines the *graph* of a layout.

Definition 3.2 ***(Graph embedding)*** Each node h of G is associated with a map $f_h : 0 \to \mathcal{M}$ that assigns this node to a point on the surface. Furthermore, each arc a of G is associated with a continuous map $f_a : [0, l_a] \to \mathcal{M}$. The maps are such that $f_a(0) = f_h(0)$ and $f_a(l_a) = f_g(0)$ if h and g are the nodes incident to arc a, i.e. the curve formed by the embedded arc on the surface starts and ends at the points onto which the incident nodes are embedded. Furthermore, embedded arcs may only intersect at their endpoints and embedded dangling arcs end on $\partial\mathcal{M}$. The set of maps f then defines a *graph embedding* for G.

Figure 6 illustrates this. The embedded arcs partition the surface into regions (called *patches*) bounded by embedded arcs, embedded nodes, and possibly segments of $\partial\mathcal{M}$ – such patches bounded partially by $\partial\mathcal{M}$ are called *trimmed*. If all patches are homeomorphic to discs, the graph embedding formally is a *2-cell embedding (with boundary)*.

Definition 3.3 ***(Node valence)*** The valence val(h) of a node h is the number of incident arcs. Note that one arc can be incident to a node (and contribute to the valence) two times. An interior node of valence 4 is called *regular*, otherwise *irregular*; a node on $\partial\mathcal{M}$ is commonly considered regular if it is of valence 3 and two of the incident arcs are aligned with the boundary (for non-aligned boundary cases there is no generic notion of regularity).

Definition 3.4 ***(Patch valence)*** The valence val(p) of a patch p that is homeomorphic to a disc is the number of embedded nodes along the patch border. Note that one node can occur multiple times along a patch border. A non-trimmed patch of valence 4 is called *regular*, otherwise *irregular*.

If all patches are regular, a 2-cell embedding of a layout graph can easily be extended to a *layout embedding*:

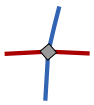
Definition 3.5 ***(Layout embedding)*** In addition to the node and arc maps, each regular patch p can be associated with a continuous map $f_p : [0, w_p] \times [0, h_p] \to \mathcal{M}$ (or a restriction thereof in case of a trimmed patch) such that this map agrees with the maps of the incident arcs, e.g. $f_p(x, 0) = f_a(x)$ or $f_p(x, 0) = f_a(l_a - x)$ (depending on the orientation). The set of maps f then defines a *layout embedding* for G.

Figure 7 illustrates the embedding of a patch via its associated map f_p.

Definition 3.6 ***(Quad layout)*** A layout graph G together with a layout embedding f in surface $\mathcal{M}$ where all patches are regular is called a *quad layout* $\mathcal{L}$ (cf. Figure 5 for an example).

3.2. Structural Properties

At regular nodes the four incident arcs can be partitioned into two pairs (depicted in red and blue on the right) of *opposite* arcs in the intuitive way. Based on this, separatrices can be defined:



Definition 3.7 ***(Separatrix)*** A chain of successively opposite arcs which starts at an irregular node and ends at a (not necessarily distinct) irregular node or $\partial\mathcal{M}$ is called (discrete) *separatrix*.

Note that we will later also deal with separatrices of cross fields and separatrices of parameterizations – concepts which are closely related but formally different.

Figure 8 illustrates the components of a quad layout.

Definition 3.8 ***(Layout minimality)*** If each arc of a quad layout is part of a separatrix, the layout is called *minimal*.

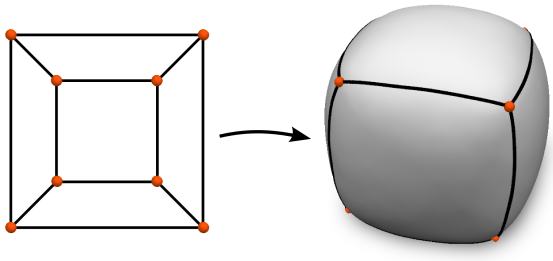


Figure 6: *Illustration of a layout graph (red nodes, black arcs) and an embedding of this graph on a surface.*

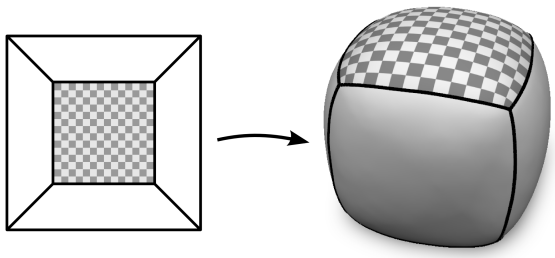


Figure 7: *Illustration of an individual patch embedding map.*

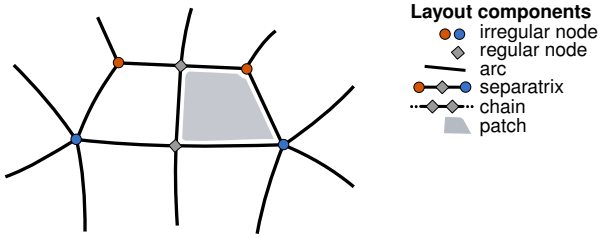


Figure 8: *Illustration of a layout's (primal) components.*

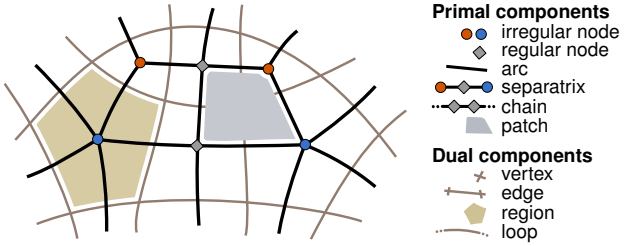


Figure 9: *Illustration of a layout's primal and dual components.*

In a minimal layout, there are no chains of arcs which are cyclic or start and end at $\partial\mathcal{M}$.

Definition 3.9 ***(Base complex)*** The largest minimal sub-layout of a layout is called its *base complex*.

Note that the base complex of a quad layout (or quad mesh) can equivalently be defined as being formed by *all* the separatrices of the layout.

3.3. Layout Duality

For the description and modification of the layout structure it can be of benefit to consider the dual layout:

Definition 3.10 ***(Dual)*** The combinatorial dual $\mathcal{D}$ of the cell complex specified by the quad layout $\mathcal{L}$ is called the *dual layout*.

$\mathcal{D}$ contains a *vertex* for each patch of $\mathcal{L}$, an *edge* for each arc of $\mathcal{L}$, and a *region* for each node of $\mathcal{L}$; we use the terms vertex, edge, and region for dual layouts in order to distinguish from nodes, arcs, and patches of primal layouts. Except for boundary cases, $\mathcal{D}$ is a 4-regular cell complex, i.e. every non-boundary dual vertex has four incident dual edges. Hence, at every non-boundary vertex there are two pairs of *opposite* edges. The set of all edges uniquely decomposes into a disjoint collection of cycles (and, when boundaries exist, boundary-to-boundary chains) of successively opposite edges. Note that these dual edge cycles (chains) correspond to – possibly non-simple, i.e. self-crossing – cyclic (or boundary-to-boundary) quad strips in $\mathcal{L}$. Geometrically, these dual edge cycles (chains) correspond to an arrangement of loops (or boundary-to-boundary curves) on the surface, cf. Figure 9; the loop intersections define the vertices and the loop segments between any two intersections define the edges of $\mathcal{D}$. The induced partition of $\mathcal{M}$ consequently defines the regions of $\mathcal{D}$ (cf. Figure 9). Note that a specific embedding for $\mathcal{D}$ is not inherent to the topological concept of duality.

3.4. Layout Quality

The *quality* of a quad layout is a measure that, at least to some extent, depends on the intended application. However, we can identify some generic aspects which are common to many application scenarios:

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

Geometric fidelity generally promotes mapping or representation quality, because the parametric distortion is a key factor in many applications, while structural simplicity gives preference to simpler surface representations, simpler mapping domains, or more flexibility for hierarchical structures (e.g. in the context of multi-block grids).

Unfortunately, both aspects tend to be opposing objectives. For instance, a genus 1 surface can always be covered with just a single conforming quad patch, but then the parametric distortion can be arbitrarily high, depending on the geometric shape of the surface. On the contrary, low distortion can be achieved at the expense of a large number of small patches (and possibly large number of irregular nodes [MZ13]), as in the case of quad meshing [BZK09, JTPS15].

The abstract quality of the layout hence is a function of complexity and distortion, and an appropriate trade-off needs to be made. How this relation looks like in detail is application-dependent.

As a consequence of this interplay of fidelity and simplicity objectives, methods for generating quad *meshes*, which typically rely on a target quad size and quad anisotropy (possibly varying over the surface) specified a priori [RLL*06, DBG*06, KNP07, HZM*08, BZK09, ZHLB10, KMZ11, BCE*13, MPZ14, LHJ*14, PPTSH14], are not well-suited for generating quad *layouts*; patch dimensions must rather be chosen automatically so as to arrive at a suitably balanced quality. Patches of widely varying sizes and aspect ratios, depending on the geometry of the surface and the global structure of the layout, can be optimal in this context.

One further aspect is quite commonly of relevance for a layout's quality:

- **Principal direction alignment:** the arcs and also the isoparametric curves of the embedded patches should follow directions of principal curvature on the surface with low deviation.

The importance of this aspect for prominent use cases of quad layouts is well known [LRL06, ACSD*03, CSAD04, CIE*16]. Depending on the application, it serves maximizing surface approximation quality [D'A00], minimizing normal noise and aliasing artifacts [BK01], optimizing element planarity [LXW*11], or achieving smooth curvature distribution (due to their tensor-product nature common spline surface representations are prone to ripples (curvature oscillations) if aligned badly [KBZ15]). Besides, principal direction alignment can also be of interest for aesthetic reasons. Due to their specific symmetries, quad layouts and quad meshes have the natural ability to align to the orthogonal principal directions. In fact, this is one of the major reasons for preferring them

over simplicial layouts [BLP*13]. Alignment to creases and boundaries may be seen as a special case of this principal direction alignment, but it can be advantageous to consider this in a dedicated, strict manner.

3.5. Layout Conformity

In this survey we focus on the generation of *conforming* quad layouts (cf. Figure 3). This focus is chosen because, first of all, conformity is an essential requirement for many use cases and, secondly, even in many of the cases where non-conformity can be dealt with, one actually needs to construct a conforming layout in the first place, as explained in the following.

Examples of techniques that are able to deal with non-conformity, with so-called T-layouts or T-meshes, are T-splines and T-NURCCs [SZBN03] or dyadic T-mesh subdivision [KBZ15]. The option to have T-junctions can provide additional flexibility for modeling and editing purposes [SCF*04, BVK08]. It is very important, though, to notice that not every T-layout is suitable for these applications. In many cases it is required that the layouts, while being non-conforming, allow for a globally consistent assignment of (knot) intervals [SZBN03, TA93] to the layout's arcs (mapping each patch to a rectangle in parameter space), or equivalently, for a global layout embedding in form of a seamless surface parametrization (cf. Section 5.2). In a few instances subdivision and spline techniques have been generalized to non-consistent knot intervals [SFL*08], but this comes with drawbacks such as non-stationarity or impractically high polynomial degrees. Globally consistent (non-zero) knot intervals, however, exist if and only if the non-conforming layout actually has an underlying *conforming* layout of which it is a locally refined or coarsened version.

Existing approaches to automatic non-conforming quad layout construction often respect this implicitly (without explicit mention): they take as input a conforming quad layout or mesh, and obtain a T-layout as a subset thereof [MPKZ10, LRL06, SCF*04, EGKT08, GMSO14], a locally regular refinement thereof [WZXH12, SZBN03], or as a subset of the iso-parametric curves of a global seamless surface parametrization (cf. Section 4.2.4) [HWW*06, CBK15].

4. Layout Structure Determination

The combinatorial structure of a layout is determined by its layout graph. For its construction one needs to determine a suitable set of nodes and connect these by a suitable set of arcs such that the resulting graph is a valid quad layout graph (i.e. such that it can be embedded in a given surface forming all-regular patches). We survey techniques that strive to solve these problems in Sections 4.1 and 4.2, respectively.

A few methods do not strictly fit in either category (node or arc determination) as they generate both, nodes and arcs, in an inherently combined manner. The most prominent examples are techniques based on the Morse-Smale complex (cf. Section 4.1.1). This circumstance is pointed out whenever such a method is discussed in the following.

4.1. Nodes

The node configuration, i.e. the number of layout nodes and possibly their desired valences (and their (preliminary) positions on the surface), needs to be decided on early. Without these nodes at hand, we cannot talk about their connectivity, or the quad nature of patches. Of particular interest are the irregular nodes, those with a valence different from 4 – regular nodes can be created on demand afterwards by many of the methods that create arcs (cf. Section 4.2) for a given node configuration (typically implicitly, by crossing arcs) where necessary or beneficial.

The valences of these irregular nodes are further discrete degrees of freedom. They can already be decided on and fixed from the very beginning, or be left open. Some of the methods that create arcs for the nodes expect the desired valences as input, others cannot respect such demands but rather imply the valences by the arcs they create. Prescribing node valences is particularly reasonable when nodes are deduced from the surface's curvature (cf. Section 4.1.2).

4.1.1. Sampling-based

A very simple option for the determination of a set of nodes (not necessarily particularly well suited specifically for a quad layout) is to distribute a prescribed number of samples in some regular manner over the surface.

Isotropic. An initial set of samples can be distributed over the surface, for instance, in a random manner [Tur92], using error diffusion [AdVDI05], via poisson disk sampling [CJW*09], or by farthest point sampling [PC06].

To achieve high regularity and isotropy, Lloyd's relaxation can subsequently be applied [DFG99], reaching a state where the samples induce a centroidal Voronoi diagram [PC06].

Isotropically distributed nodes, i.e. uniformly sized patches, are, however, not necessarily beneficial for high layout quality. Variably and anisotropically sized patches typically allow for simpler, more parsimonious representations.

Anisotropic. Variants of some of the above approaches can take a prescribed sizing and anisotropy into account, for instance derived from the surface's shape operator in order to achieve some form of principal curvature direction dependence [LWSF10]. Note, however, that this is only a form of anisotropy determined a priori from local measures. Letting the global structure of the layout decide the patch dimensions, as some of the more sophisticated methods to be discussed in the following section allow, can be beneficial in order to yield simple, yet high-quality layouts.

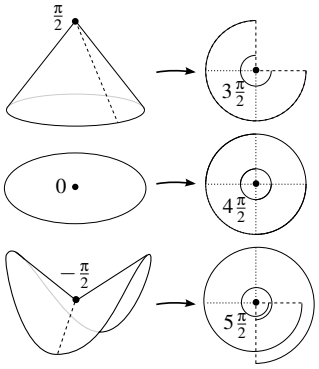
Quad-oriented. So far no attention has been paid to the fact that the samples are actually supposed to get connected to form nicely shaped quadrilateral patches. This aspect can be considered by making use of (an approximation of) the L_∞-norm, rather than the standard L_2-norm, in the Lloyd relaxation process [LL10].

Another option is the use of the critical points of Laplace-Beltrami eigenfunctions on the surface (or similar surface functions [HZM*08, ZHLB10, LHJ*14]) as nodes. These even come with a pure quad connectivity via Morse-Smale theory [DBG*06].

Error-driven. Using controlled triangle mesh simplification [GGK02] or clustering approaches [EH96, BMRJ04, PCK04], one can yield a sampling (often effectively a sub-sampling of the initial mesh vertices) that fulfils certain quality conditions. For instance, one can ensure that the surface regions in between samples are not far from being planar. This may be helpful if quad layouts with near-planar patches are sought. However, note that these approaches typically do not take into account the fact that ultimately a layout with *quadrilateral* patches is to be constructed. Strict quality guaranteed can thus not be given in general.

4.1.2. Gaussian Curvature-based

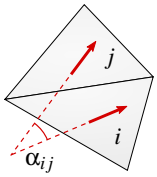
The two global layout quality criteria *geometric fidelity* and *structural simplicity* discussed in Section 3.4 imply some local criteria. For instance, a patch corner ideally has an angle of $\frac{\pi}{2}$. Thus, a node of valence k ideally lies at a point where the surface has Gaussian curvature $2\pi - k\frac{\pi}{2}$ because around such a point the (cut) surface unfolds to a sector with inner angle $k\frac{\pi}{2}$ [PS98] as depicted here. It is obvious that this ideal state typically cannot be achieved – unless the surface has isolated points s_i with a Gaussian curvature that is an integer multiple of $\frac{\pi}{2}$ while all other points of the surface have vanishing Gaussian curvature. Then one could place an irregular node n_i of valence $\text{val}(n_i) = k_i$ onto each point s_i with Gaussian curvature $2\pi - k_i\frac{\pi}{2}$ (regular nodes could be placed anywhere else as needed). This is only possible on an extremely restricted class of surfaces, namely those of unions of boxes (in particular polycubes, cf. Section 7.2).



On general surfaces with an arbitrary Gaussian curvature distribution, it proved very powerful to construct an *alternative* notion of Gaussian curvature that is as close as possible to the original curvature, subject to the condition that it is zero almost everywhere, except for some isolated points where it is some multiple of $\frac{\pi}{2}$. Then suitable nodes can be derived trivially as above. For this the meaning of "as close as possible" needs to be clarified. A appropriate measure for this can be defined in terms of the *metric connection* [CDS10]. A connection defines a notion of parallel transport on a surface and thus implies a notion of curvature. In particular, the *Levi-Civita* connection ∇_{LC} – the unique metric connection that transports tangent vectors without torsion – implies the usual Gaussian curvature.

The idea now is to construct a connection $\nabla_{\frac{\pi}{2}}$ which (A) is (in the least-squares sense) as close as possible to the original ∇_{LC} while (B) implying a curvature which is an integer multiple of $\frac{\pi}{2}$ everywhere (while it is 0 for all but a finite number of points s_i). Due to condition (B), we can then easily derive a set of irregular nodes and their valences from this connection. Due to condition (A) we can expect that this irregular node configuration, while not ideal, is in some sense as good as possible for the given surface geometry.

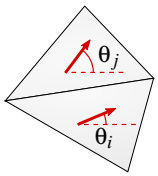
Connection Construction. In the discrete setting, i.e. on a triangle mesh, a metric connection can be expressed via *adjustment angles* α across the edges, which express the tangential rotation a tangent vector undergoes as it is transported from one face to a neighboring face across an edge. The discrete Levi-Civita connection is characterized by all these angles being zero. As we are interested in a connection as close as possible to Levi-Civita, our objective is $\|\alpha\|_2 \rightarrow \min$, subject to condition (B).



As detailed in [CDS10] condition (B) can be expressed via constraints for a set of basis face cycles which enforce that the adjustment angles exactly cancel the curvature of the Levi-Civita connection $\pm m_i\frac{\pi}{2}$. The factors m_i (which imply at which vertices the implied Gaussian curvature is which multiple of $\frac{\pi}{2}$) are integer variables of the optimization problem $\|\alpha\|_2 \rightarrow \min$, i.e. we end up with a mixed integer problem. Hence, a mixed integer solver is necessary for this optimization.

Cross Field Construction. An alternative formulation of this optimization problem, based on tangent direction fields, is possible. It is essentially equivalent but provides additional interesting insights and possibilities (in particular: consideration of principal direction alignment).

Instead of representing the per-edge adjustment angles as explicit variables, we can represent them implicitly as the angular differences between variable tangent directions in the incident faces, i.e. $\alpha_{ij} = \theta_i - \theta_j + \kappa_{ij}$, where θ are the angles of the tangent directions with respect to a local per-face reference system and the constant κ_{ij} aligns the reference systems.

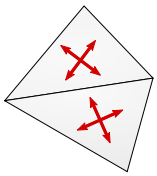


We could then perform an optimization with respect to the objective $\sum_{e_{ij}\in E}\left(\theta_i - \theta_j + \kappa_{ij}\right)^2 \rightarrow \min$, where E is the set of triangle mesh edges. Due to the relation between α and θ this is essentially equivalent to the connection based optimization problem – with one significant difference: constraints for a condition akin to (B) are inherent to this formulation. The resulting connection obviously transports vectors to themselves along any curve (just notice that by construction the field θ – and any constant rotation thereof – is parallel w.r.t. the connection given by α). Hence the implied curvature is always an integer multiple of 2π everywhere. The multiplicity, however, is fixed arbitrarily (formally and more precisely, it is implied by the indices of the field of chosen local reference systems) and we are actually interested in multiples of $\frac{\pi}{2}$, not 2π.

Both problems can be solved by adding integer variables m to the system which allow to add any multiple of $\frac{\pi}{2}$ to the terms: $\sum_{e_{ij}\in E}\left(\theta_i - \theta_j + \kappa_{ij} + m_{ij}\frac{\pi}{2}\right)^2 \rightarrow \min$.

The connection $\nabla_{\frac{\pi}{2}}$ resulting from this optimization then fulfills (A) and (B). In particular, a vector is not necessarily transported to itself along any curve by this connection but may be off by a multiple of $\frac{\pi}{2}$. A *cross* (a set of four directions invariant to rotations by $\frac{\pi}{2}$), however, is always transported to itself, so we are in fact

optimizing a cross field $\mathcal{C}$ for smoothness – a cross field which "by example" represents $\nabla_{\frac{\pi}{2}}$. Therefore it comes as no surprise that exactly the same optimization problem is used for the construction of smooth cross fields [LVRL06, PZ07, RVLL08, VCD*16] for purposes of field-guided quad meshing [BZK09].



Principal Direction Alignment. Various techniques have been proposed that allow taking some principal curvature alignment objective into account when generating a smooth cross field [VCD*16]. It can be beneficial to make use of these here in the construction of the cross field from which the layout nodes are derived. This influences the resulting node configuration, empirically increasing its suitability for a layout connectivity of high quality (cf. Section 3.4).

Optimization. The above optimization problems can be solved using mixed integer solvers [BZK12], as described in [BZK09]. But also alternative formulations that do not require a mixed integer solver have been proposed [KCPS13, DVPSH14]. In these methods the integer degrees of freedom are essentially expressed implicitly [PZ07] rather than explicitly [LVRL06]. Furthermore, taking the integrability of the cross field into account allows to influence the implied node configuration towards allowing for more rectangular patches (typically at the expense of a higher number of nodes, thus patches) [Nie12, DVPSH15]. It can also be helpful to abstract from noise or geometric detail in this context in order to avoid an excessive number of nodes [RVAL09, ECBK14].

Figure 10 shows the irregular nodes obtained for various models using the described method.

Validity. This connection or cross field based approach yields nodes together with a prescription of desired valences for these nodes. In general, a set of nodes with prescribed valences does not necessarily admit a pure quad layout that realizes these valences. It follows from Euler's polyhedron formula that, for any quad layout, $\sum_{n_i \in N}(1-\mathrm{val}(n_i)/4) = \chi$, where χ is the surface's Euler characteristic, and N the set of layout nodes.

We note that summands corresponding to regular nodes of valence 4 vanish. The sum over the irregular nodes' terms, however, needs to match the surface's Euler characteristic. This constraint needs to be considered in the optimization because otherwise the set of nodes does not admit a quad layout.

Fortunately, when determining the nodes and valences as described above, this condition is always fulfilled. It is a consequence of the Poincaré-Hopf theorem, relating a cross field's singularity indices (which imply the node valences) to the surface's Euler characteristic [Cam14, RVLL08, PZ07]. It must, however, be noted that this condition is a necessary, not a sufficient condition. However, only a single configuration fulfilling Euler's condition but not admitting a quad layout exists [JT73, MPZ14], and one can simply modify it should it occur.

4.2. Arcs

Once the desired nodes have been determined, their connectivity needs to be established, i.e. arcs need to be created. This settles the combinatorial degrees of freedom. It is important to note that this is not a purely combinatorial problem, it has a topological component. In particular, an arc cannot be identified by its two incident nodes alone; its path homotopy class is crucial in addition. Intuitively, an arc can wind around the handles of a surface (and the nodes on the surface) different numbers of times, ultimately implying different layouts. To be precise: the path homotopy class with respect to the surface punctured by the nodes (i.e. we can treat the nodes as forming small holes in the surface) is what matters [MPKZ10]. Note that two nodes can be connected by multiple, non-homotopic arcs, and a node can be connected to itself by an arc.

The number of arcs to choose from is thus infinite. A large variety of methods have been proposed to make this choice in such a way that the resulting layout consists of only quadrilateral patches. We discuss them in this section. We remark that some of them (inherently) introduce additional nodes in the process. While some introduce irregular nodes, others are able to restrict to additional regular nodes which do not negatively affect the geometric layout quality.

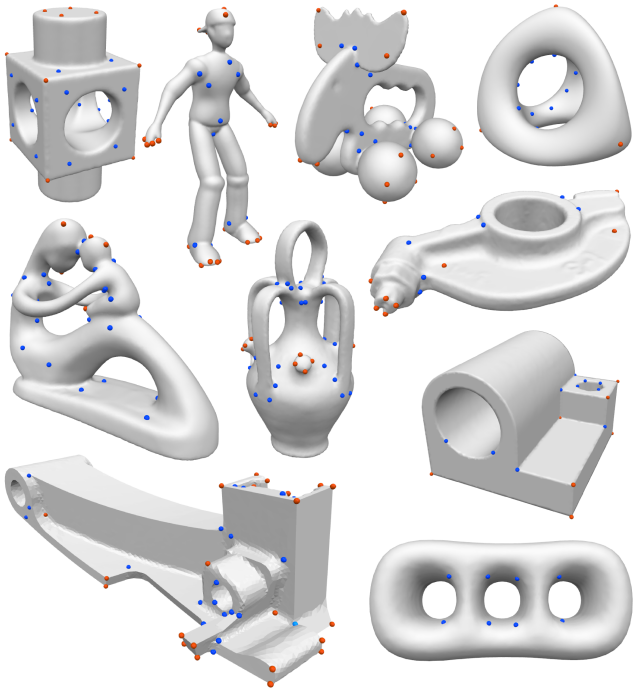


Figure 10: *Irregular nodes computed using a principal direction guided cross field optimization (cf. Section 4.1.2). Valence 3 nodes in red, valence 5 nodes in blue, valence 6 nodes in cyan. Note how most valence 3 nodes lie in regions with positive Gaussian curvature and valence 5+ nodes in regions with negative Gaussian curvature, but where appropriate or necessary, e.g. in order to fulfill the Poincaré-Hopf theorem, nodes can also arise in flat regions.*

4.2.1. Triangulation-based

One might imagine simple strategies of adding arcs between nodes one by one in an incremental, greedy manner (e.g. using advancing front techniques), or based on some notion of node proximity (defined, e.g., by a Voronoi diagram). Achieving a final state in which every single implied patch has valence 4, i.e. is a quad, however, is hard.

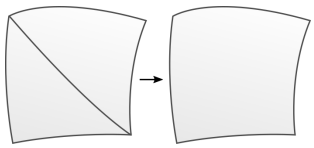
If we were to drop this structural requirement, i.e. aim for arbitrary polygonal patches, the problem becomes easy. Interestingly, achieving a layout in which every single implied patch has valence 3, i.e. is a triangle, is not harder than this unconstrained problem: once a polygonal layout is available, each n-gon can easily be split into $n-2$ triangles by locally inserting $n-3$ additional arcs – effectively because triangles are simplices. Splitting into quads by additional arcs, by contrast, only works if each polygon is a $2n$-gon (a condition hard to impose on a Voronoi diagram or any incremental arc generation approach).

Based on this insight, a number of methods have been developed that initially create a triangle (or polygonal) layout, which is then turned into a quad layout by various kinds of modifications performed in a post-process. These are discussed in this subsection.

An initial triangle layout can, for instance, be generated using intrinsic or restricted Delaunay tessellation [AdVDI05, PC06, YLL*09] (or, if available, simply be taken from the mesh used to obtain the nodes, cf. Section 4.1.1). Polygonal layouts typically result from clustering approaches [CSAD04, PCK04, BMRJ04]

When using the Delaunay tessellation approach, it can be advisable to employ (an approximation of) the L_∞-norm in the computation. This has been demonstrated to yield triangulations whose triangles often form well-shaped quads in pairs [LL10]. This "preconditioning" of shape and arrangement of the triangles can lead to a quad layout of higher quality when applying the post-processing techniques described in the following. Still, control over the geometric fidelity and alignment is limited due to the two-step nature of all of these approaches.

Pairing. Two adjacent triangles can be turned into a quadrilateral by removing the separating arc from the mesh, thereby *pairing* the two triangles to become a quad. This pairing can be modeled using the dual graph of the triangle mesh.



The dual graph (V,E) consists of a set V of dual vertices, one for each triangle, and a set E of dual edges, one for each arc. Note that each dual edge identifies a pair of triangles. A subset E' of the edges thus describes a set of triangle pairs. To allow for a merging of triangle pairs to quads, it is important that no triangle appears in more than one pair. This gives rise to the following definition.

A *matching* of the dual graph is a subset $E' \subseteq E$ such that each vertex of V has at most one incident edge in E'. A matching is *perfect* (or *complete*) if each vertex of V has exactly one incident edge in E'; in this case $2|E'| = |V|$. A perfect matching thus describes a triangle pairing such that a pure quad layout is obtained. An imperfect matching, by contrast, leaves one or more triangles unpaired, resulting in a mixed triangle-quad layout.

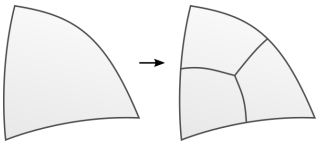
A variety of strategies have been proposed for the construction of matchings for given triangulations, as detailed in the following. Because we are dealing with manifold triangle meshes, note that we can reasonably assume $|E| = O(|V|)$ for the dual graph in the following.

- **Greedy Matching.** Add edges from E to E' (initialized with $\varnothing$) one by one, preserving the matching property (at most one incident edge per vertex). The ordering can be driven by some quality measure, a weight $w(e)$ assigned to each dual edge $e \in E$, assessing the shape properties of the quad resulting from each individual pair [BF98, LL94, JSK91, Hei83]. Assuming the weight can be computed in $O(1)$ time per edge, this approach has time complexity $O(n \log n)$.
- **Maximum-Cardinality Matching.** Ignoring geometric quality of the result and instead aiming for a layout with as many quads (as few triangles) as possible, one can aim for a matching that is as complete as possible, maximizing $|E'|$. Such a matching can be found in time $O(n^{1.5})$ [Blu90].
- **Max-Min-Weight Matching.** The dual graph typically admits many matchings of maximum cardinality. Among those, it is reasonable to aim for the best one – in terms of some quality measure. One way of defining this is via maximality of the minimum weight of edges in E', i.e. $E' = \arg\max_{F \in \mathcal{F}} \min_{e \in F} w(e)$, where $\mathcal{F}$ is the set of all maximum-cardinality matchings [EH96]. This setup assumes that a high weight indicates a preferable pair, leading to a high quality quad. A max-min-weight matching can be found in time $O(n^3)$ [Law76].
- **Minimum-Cost Perfect Matching.** Instead of considering the weight of only the worst pair in the matching, one can consider the cost $c(F) = \sum_{e \in F} w(e)$, i.e. the sum of weights of all edges in the matching F, and select one with minimum cost, i.e. $E' = \arg\min_{F \in \mathcal{F}} c(F)$ [RLS*12]. This setup assumes that a low weight indicates a preferable pair. Under the assumption that a perfect matching exists, this minimum cost matching can be found in time $O(n^2)$, using an improved variant [Gab90, Kol09] of the blossom algorithm [Edm65].

  This assumption holds in many cases: every triangle mesh without boundary has a dual perfect matching [EKK*11]. For meshes with boundary, however, a perfect matching does not always exist (in particular for meshes with an odd number of triangles). Remacle et al. [RLS*12] describe a simple strategy to effectively make unpaired triangles appear along the boundary only, where they can be eliminated by simple local modifications (flips and splits). There is no proof of guaranteed success though.

Refinement. Instead of treating a triangle as a half-quad (to be paired with another triangle to form a full quad), it can be treated to be consisting of multiple quads. The application of a suitable mesh refinement operator can materialize these individual quads, making them replace the original triangles.

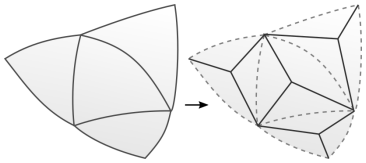
- **Catmull-Clark.** Inserting an additional node in the center of each arc and each patch, and then inserting additional arcs connecting each arc center node to the adjacent patch center nodes leads to a pure quad layout. Effectively, each triangle is split into three quads by this operation (the refinement operator used in the Catmull-Clark subdivision scheme [CC78]). This technique has been used for the purpose of quad layout construction [DSC09, PPT*11].



  This refinement operator can be applied to arbitrary polygon meshes as well: an n-gon is then split into n quads [PCK04].

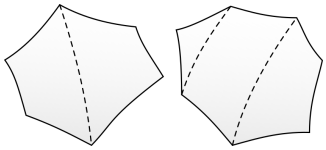
  It must be noted that the quality of the resulting quad layout in terms of regularity is low. The center node inserted into an n-gon has valence n in the layout. In the case of a triangle mesh, this leads to numerous irregular nodes of valence 3, and in the case of polygon meshes potentially to high-valence nodes.

- **Arc-Centric.** Inserting an additional node in the center of each patch, and then inserting additional arcs connecting each such patch center node to the patch's corner nodes, splits every triangle into three sub-triangles. Now removing the original arcs, pairs these sub-triangles to form quads (one per original arc) [VZ01]. Note that this creates a quad mesh with half as many quads as Catmull-Clark refinement does. Special care must, however, be taken at boundaries, where sub-triangles have no partner. One can, for instance, split boundary arcs to solve this issue in terms of connectivity.



  Note that this refinement operator can be applied to arbitrary polygon meshes as well.

- **Polygon Split.** When dealing with polygonal initial layouts, another strategy that inserts only arcs but no additional nodes, thus creates layouts with a lower number of patches than the above refinement operators, can be used [BMRJ04]: each $2n$-gon is split into $n-1$ quads by inserting $n-2$ arcs. Each $(2n+1)$-gon, however, can only be split into $n-1$ quads and 1 triangle (by inserting $n-1$ arcs). For general polygonal layouts, it does thus create a mixed triangle-quad layout only.



Hybrid Pairing and Refinement. Performing a complete pairing (by means of a dual perfect matching) is relatively expensive and might leave limited control over quality. One can instead perform a partial pairing by some greedy approach that is stopped when some quality criterion is about to be violated (or when no further pairing is possible). The resulting mixed triangle-quad layout can then be turned into a pure quad layout by one of the refinement operators. This partial pre-pairing can have the advantage of achieving a simpler result (compared to applying refinement to the triangle layout immediately) and higher regularity (because refinement of quads does not introduce additional irregular vertices) [VZ01].

Higher quality can be achieved by applying the refinement not to partially paired triangle meshes, but to *quad-dominant* layouts created with particular attention to surface geometry and directional alignment [RLL*06, JTPS15, MK04, LKH08].

Decimation. Once a pure quad layout has been obtained using pairing or refinement techniques, one might want to try to adjust the complexity, node density, or regularity of the quad layout using generic quad mesh decimation techniques [DSSC08, DSC09, TPC*10]. Driving this decimation in a way that the resulting layout is geometrically faithful and irregular vertices end up in plausible locations is challenging. In this context Panozzo et al. [PPT*11] use special kinds of pre-computed sizing fields (called *fitmaps*) to influence the decimation process.

4.2.2. Binary Optimization

While a partition into triangles can easily be obtained in various ways (cf. Section 4.2.1), directly creating a partition where all patches are quadrilateral is not possible in similarly simple, local, greedy manners. For pure quad layouts, more global approaches are necessary to take this condition into account.

The problem that needs to be solved can be formalized as follows: *select a subset of the set of all possible arcs, such that each patch implied by this subset is quadrilateral.*

The hardness of this problem mainly comes from two facts: the set of all possible arcs is infinite in general (cf. Section 4.2), and the question of whether the implied patches are quadrilateral is not a purely combinatorial but a geometric question, considering that intersections of arcs need to be taken into account (either be prevented, or be considered as implicitly forming additional regular nodes).

This hard problem has recently been tackled by pre-filtering the set of all possible arcs such that firstly it becomes relatively small and, in particular, finite, and secondly arcs intersect only in specific ways, such that intersections can be taken into account more easily. The simplified problem can then be formulated as a global binary optimization problem and approached using standard solvers [RRP15, ZZY16, PPM*16, RP17]. An objective based on the arcs' geometric quality (e.g. in terms of simplicity or alignment to desired directions) can be used to find a good layout among the valid ones.

This pre-filtering approach comes with a risk, though: there is no guarantee that the selected subset actually admits a quad layout; and if it does, it might be arbitrarily far away from the optimum in terms of quality.

Pre-Filtering To restrict the possible intersection patterns, one can restrict the arcs' tangents based on a global *seamless* parametrization of the surface which has *cones* at the nodes (with the cone angles corresponding to the desired node valences) [MZ12, BZK09, TACSD06], cf. Figure 13 for an example of such parametrizations.

The separatrices of such a parametrization (i.e. the u- and v-isocurves emanating from the nodes) are guaranteed to be finite

if the parametrization's charts' transitions have *rational* translational components [CBK15]. They furthermore intersect only orthogonally away from the nodes, and the patches formed are all quadrilateral (cf. Section 4.2.4). Directly using these separatrices as arcs, however, is problematic: the implied quad layout can be highly complex because the separatrices can be arbitrarily long and form arbitrarily many intersections [BLK11, TPP*11].

It can however be shown that the structural properties are preserved if one allows the arcs' tangential direction to deviate from the parametrization's isocurve directions by less than $\pm 45^\circ$ [CBK12]. The arcs do then no longer intersect orthogonally, but those aligned to the local u-isocurve ($\pm 45^\circ$) do still *transversally* intersect those aligned to the local v-isocurve. With proper precautions (in particular: a single arc per *port* of each node), this is sufficient to guarantee that no non-quad patches are formed [RRP15].

This can be exploited by considering as candidate arcs only curves that fulfil this $< \pm 45^\circ$ criterion with respect to a given global seamless parametrization, or, to simplify implementation, a quantized [CBK15] version thereof (effectively a quad mesh) [ZZY16, RP17]. The property of all patches being quadrilateral can then be expressed using relatively simple binary conditions that prevent multiple intersecting candidate arcs that are locally aligned to the same parametric direction from being chosen for the layout together [RRP15].

The candidate arc set defined in this way is still infinite, though. Additional criteria are necessary to reduce it to a finite size. For instance, one can use an arc length cut-off, a threshold on a spiral pitch measure [RRP15], a cardinality threshold [PPM*16], or regional restrictions [ZZY16, RP17]. Figure 11 shows an example candidate set on a simple surface. Depending on the choice of parameters, the candidate set might, however, be too small in the sense of not admitting any quad layout at all – unless one includes the arcs of some initial (possibly low quality) quad layout in the set, effectively enabling a fallback to this solution in the worst case [RP17] (a property shared with simplification strategies [TPP*11, BLK11] that start from initial layouts, cf. Section 4.2.4). A very generous choice, on the other hand, can lead to an overly large, intractable optimization problem.

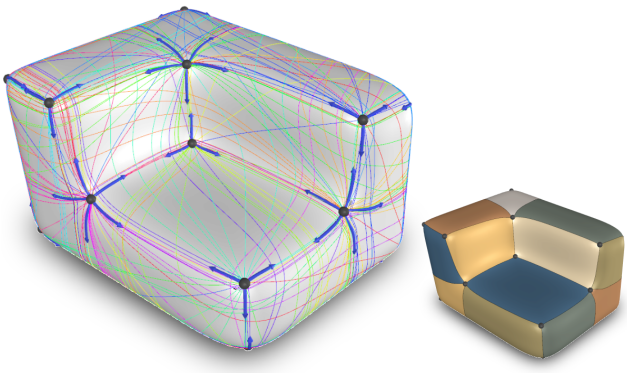


Figure 11: *Candidate arcs/separatrices for a given node configuration (black dots, their ports depicted using blue arrows) [RRP15]. Chosing a proper subset of these based on a system of binary conditions and local quality ratings can yield the quad layout on the bottom left. (Image courtesy of F. Razafindrazaka)*

A potential difficulty with this approach is the hardness of constructing a global (quantized) seamless parametrization in the first place [MPZ14, CBK15]. One can consider using a *cross field* (which also provides two orthogonal directions everywhere on the surface) instead [PPM*16, KLF15], as it is significantly easier to construct for a given set of nodes (i.e. field singularities) [VCD*16]. However, not for all cross fields do the separatrices (the field's integral curves emanating from singularities) form a quad layout, and the resulting candidate arcs may not admit any quad layout either, no matter how the thresholds are chosen. This can, for instance, be due to the existence of *attractors* in the field or due to globally unfit field holonomy [MPZ14, KNP07]. With a suitable formulation one can, however, at least obtain partial or non-conforming layouts in such problematic cases, that can be post-processed into quad layouts by refinement operations [PPM*16].

4.2.3. Dual Loops

As discussed in Sections 4.2.1 and 4.2.2 directly constructing a quad layout – without having to resort to post-processing based on pairing or refinement strategies – requires a global consideration of the problem; incremental, greedy construction seems intractable. However, it was observed [CBK12] that an incremental construction is possible in the dual setting: instead of the primal arcs of the layout, one considers its dual edges. As discussed in Section 3.3 these dual edges form loops [MBBM97].

The key observation in this dual setting is that *any* arrangement of loops that A) have only simple intersections and B) partition the surface into regions of disk topology, is dual to a pure quad layout [CBK12], Figure 12 shows an example. An intersection is not simple if the loops are in tangential contact or if more than two loops intersect at the same point. Such arrangements can easily be constructed incrementally: simply add loops until all non-disk regions are split into disks, while avoiding non-simple intersections.

The major challenge, though, is to perform this in such a way that the resulting quad layout is not only structurally pure, but geometrically faithful as well. Two methods that take layout simplicity, arc straightness, and alignment to principal directions and feature curves into account in an incremental dual loop construction process have been proposed in recent years. One takes a set of nodes and valences as input and constructs suitable arcs, the other more flexibly constructs nodes and arcs simultaneously. We describe them in the following.

In general, the dual loops should be short, straight (geodesically), and aligned to directions of principal curvature [CBK12]. These three objectives are not compatible in general, i.e. they must be balanced. The relative weighting between straightness and principal direction alignment can reasonably be driven by a measure of local surface shape anisotropy. For instance, in regions where the surface is cylindrical and thus has a very pronounced directional nature, aligning to either one of the principal directions is more important than in nearly flat or spherical regions, where straightness should be the commanding principle.

The relative weighting of the shortness objective versus these di-

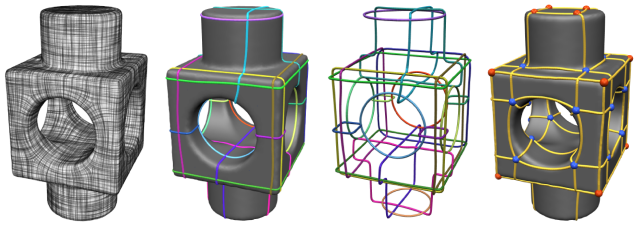
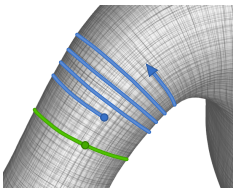


Figure 12: *Dual loops approach. Left: principal direction aligned cross field. Middle: dual loops arrangement (on the surface and separate for clarity). Right: primal pure quad layout.*

rectional objectives is a free parameter that, to some extent, allows to choose between simpler and more geometrically faithful layouts.

Fixed Nodes. In analogy to the description in Section 4.2.2, for a given configuration of nodes and valences, a cross field with singularities of corresponding indices at these nodes can be constructed (cf. Figure 12 left). Alternatively, a global seamless parametrization with cones at the nodes, as in the primal, global binary optimization based approach, can be used. This cross field (or parametrization) is optimized based on straightness (low Dirichlet energy) and principal curvature direction alignment objectives [BZK09, VCD*16].

Now consider a closed *integral curve* of the cross field (or isocurve of the parametrization): it forms a loop that is perfectly in line with the combined straightness and alignment objective represented by the cross field – but entirely neglects the shortness objective; these curves can be arbitrarily long and wind around the surface many times (blue in the inset). By contrast, a closed *geodesic* curve forms a loop that exclusively considers the shortness objective.



The idea now is to combine these two concepts and construct *anisotropic* geodesic loops (green in the inset), where the underlying anisotropic metric is implied by the cross field, and the chosen strength of the anisotropy effectively balances the shortness objective and the directional objective. Dual loops which are optimal in a discrete anisotropic geodesic sense can efficiently be found using a dynamic programming approach, akin to Dijkstra's shortest path algorithm [CBK12]. Using additional constraints, improper loop intersections can be ruled out.

One can then add such optimal loops one by one. As one wants to use the predetermined nodes for the implied primal layout, the nodes should come to lie in separate regions of the dual layout, i.e. each pair of nodes should be separated by loops. Guided by this principle, loops can be added with the goal of separating yet unseparated nodes [CBK12]. This is not guaranteed to be always possible, so some nodes might get merged in the final layout. If using a cross field and not a parametrization to guide the loops, globally unfit field holonomy is theoretically able to prevent finding enough loops to partition the surface into only disk-like regions, similar to the primal case (cf. Section 4.2.2).

Once enough dual loops have been constructed, the actual primal quad layout can be generated by dualizing the embedded graph defined by the arrangement of loops. Figure 12 shows and example of the dual and primal layout.

Free Nodes Alternatively, one can construct the dual layout more flexibly by not relying on predetermined nodes. This is particularly important to enable user interaction, guidance, and additional design constraints in the process [CK14a].

Without prescribed nodes, one cannot make use of the cross field or parametrization (whose singularities or cones correspond to prescribed nodes) as before. Without the directional information provided by them, we cannot model the set of desirable dual loops as anisotropic geodesics. We need to model the objectives of straightness and of directional alignment (which are otherwise taken care of jointly by the cross field alignment) separately.

For this purpose *elastica* curves are required. In contrast to geodesics, their optimization additionally takes second-order curve properties into account, which is required to be able to consider straightness. By constructing the so-called derivative graph of a surface triangulation, optimal discrete elastica loops can still be found using an efficient Dijkstra-type algorithm [CK14a], and positional and directional constraints can be taken into account for interactive layout design purposes.

One of the advantages of this approach is that the restrictions due to a predetermined cross field, that could present an obstacle to obtaining a good layout, are dropped.

4.2.4. Integer Grid Maps

A valid set of arcs, or, more precisely, separatrices (i.e. chains of arcs between irregular nodes, cf. Section 3), can be derived from a global parametrization's isocurves emanating from the nodes. If the parametrization is *seamless* and has a *cone* of angle $k\frac{\pi}{2}$ at each node of valence k [MZ12], there are k such curves emanating from each valence k node, they intersect only orthogonally (implying additional regular nodes), and partition the surface into quadrilateral regions. However, only if the seamless parametrization is *quantized* [CBK15] (or, without loss of generality, integer rounded [BZK09, KNP07]), do these curves form a finite quad layout. Figure 13 shows an example.

Such a parametrization is also known as *integer grid map* [BCE*13], because it maps a regular axis aligned grid in parameter space onto a quad mesh on the surface. Many recent quad meshing methods (cf. Section 3.4) are based on this principle. The quad layout formed by the parametrization's separatrix curves actually is the *base complex* (cf. Section 3) of this implied quad mesh.

This base complex can, however, be arbitrarily fine; the separatrix curves can wind around the underlying object many times and form numerous intersections, implying additional (regular) nodes and thus additional patches (cf. Figure 13 left and Figure 14). While additional constraints can be added to the parametrization optimization problem to yield a simpler base complex [MPKZ10] (cf. Figure 13 right), choosing an appropriate and valid set of constraints is challenging, and their effectiveness limited.

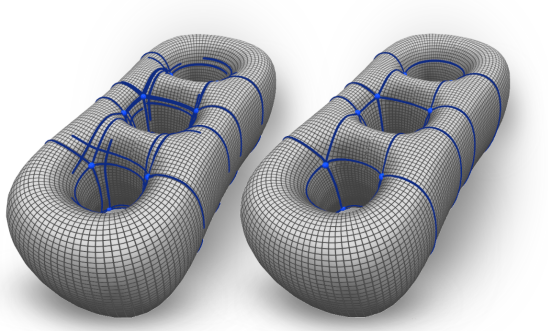


Figure 13: *Global seamless parametrization with cones at nodes (blue dots), visualized using a fine grid of isocurves. The parametrizations on the left and right differ slightly. On the left, the separatrices (shown only partially) form a very complex layout with small patches. On the right, the separatrices are short and form a very simple quad layout.*

Extremal Integer Grid Maps

One powerful way to strictly simplify the base complex, though, is to make the underlying integer grid simpler in the first place. As the base complex is a subset, thus either equally or less complex, this bounds the complexity of the resulting quad layout. This is achieved by performing a stronger quantization of the parametrization, resulting in a coarser implied quad mesh.

This, however, comes with a complication: the robustness of many integer grid map generation methods decreases with increasing coarseness. The parametrization tends to become invalid (non-injective, containing degeneracies or inversions) more easily. Countermeasures such as stiffening [BZK09] are often not powerful enough to prevent this in this case either.

Recent advances in this field have, however, led to robust rounding or quantization techniques [BCE*13, CBK15]. With these, the quantization strength (*target edge length*) can be increased arbitrarily, even to extreme values beyond the size of the model. Due to the robust procedures, still a valid result can be obtained; the involved correctness constraints implicitly (anisotropically) reduce the size of the individual quad elements such that they fit the prescribed nodes. In this way, very coarse quad meshes, with possibly even coarser base complexes, can be obtained.

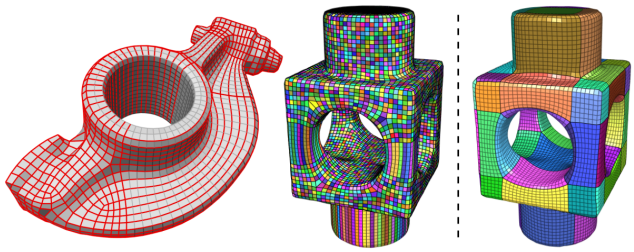


Figure 14: *Quad meshes [BZK09] and their base complexes. On the left the arcs of the base complex are highlighted (red). In the middle the base complex patches are visualized by individual colors – on the right a very similar quad mesh of this object is depicted (same resolution, same number and type of irregular nodes) with a much simpler base complex is depicted. The chances of directly yielding a mesh with such a simple, nicely structured base complex from an unconstrained integer grid map construction are very low.*

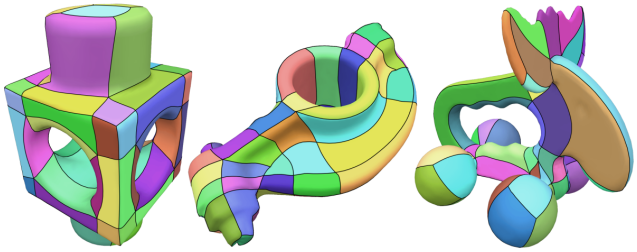


Figure 15: *Quad layouts obtained from an extremal integer grid map, optimized for as-large-as-possible patches (given prescribed irregular nodes) [BCE*13]. (Image courtesy of D. Bommes)*

It can be observed, though, that for large prescribed target edge lengths, the optimization of the parametrization seems to be dominated by the aim of fulfilling the involved validity constraints, and geometric quality occasionally suffers, resulting, e.g., in badly shaped patches and badly aligned arcs.

Figure 15 shows some example layouts obtained in this way.

Greedy Simplification

The recent robust methods that are able to construct *extremal* integer grid maps [CBK15, BCE*13] are not particularly simple and consist of multiple non-trivial components. Somewhat simpler, non-robust variants [KNP07, BZK09], however, can only reliably be used to generate relatively fine quad meshes, with a high number of additional regular nodes, i.e. long separatrices that form many intersections.

A possible approach is to take such an initial layout which is too fine, too complex, and subsequently attempt to simplify it by means of connectivity editing operations applied to the separatices, aiming for shorter separatrices that form fewer intersections. This can be seen as a local variant of the global binary optimization approach (cf. Section 4.2.2), with the difference that a valid starting layout is known which is then incrementally modified. This avoids the risk of not finding any solution (in case pre-filtering was to aggressive) at the cost of possibly not getting very far with the local modifications.

Using sets of operators that preserve the quad structure, one attempts to reduce the total length of separatrices, thus the complexity of the implied layout. Such operators can be built from separatrix removal and re-insertion steps [TPP*11], or by focusing on eliminating certain helical configurations, which were identified as most relevant for base complex simplicity [BLK11]. Using greedy strategies the underlying huge search spaces are then explored in the search of modified quad meshes with a simpler base complex, implying simpler quad layouts.

5. Layout Embedding Optimization

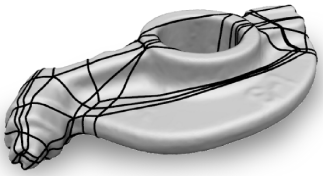
Once the layout's structure in terms of nodes and connecting arcs has been determined, the layout's embedding in the surface needs to be decided. This involves answering the questions of where exactly the nodes should be positioned and which routes over the surface the arcs should take. The embedded nodes and arcs then delineate the quad patches – whose interior embedding (in form of parametrizations over rectangles, cf. Section 3) one might want to compute and optimize as well.

All of the node determination methods (cf. Section 4.1) and most of the arc determination methods (cf. Section 4.2) already provide an embedding of the nodes and arcs. While this could be taken as is, it is typically reasonable to consider it preliminary and strive for optimization. We discuss methods that were proposed for this purpose in the following.

5.1. Arc Straightening

Embedded arcs should ideally be as straight as possible, i.e. they should be geodesic curves on the surface – at least when considering the situation just locally. This allows the two neighboring patches to be mapped to rectangular domains with low distortion along this arc.

Using simple straightening approaches the initial (possibly jagged, badly routed) arcs can be adjusted towards, or completely turned into, geodesic curves [PCK04, LRL06]. But while being geodesic is optimal locally for an individual arc, globally, where the interplay of arcs and patches is important, it can be problematic. This can be seen in the inset where geodesically embedded arcs lead to near-degenerate patches. Compare this to the layout embedding depicted in Figure 5. Obviously, to achieve a layout of overall high quality, it can be advantageous to embed arcs non-geodesically – depending on the global situation.



5.2. Global Parametrization Optimizaton

To perform a global optimization of the node, arc, and patch embeddings in an integrated manner, one makes use of a global (chart atlas based) parametrization that represents a complete layout embedding (cf. Section 3), consisting of node, arc, and patch embeddings. This global parametrization is to be optimized with respect to some distortion measure, for instance aiming for isometry or conformality.

Unfortunately, this leads to a non-linear, non-convex optimization problem. If one keeps the nodes fixed, one arrives at a simple, convex problem for certain objectives, such as harmonicity [TACSD06, BVK08] or cross field alignment [CK14b]. Such a formulation can thus be used to efficiently optimize the arc and patch embeddings (cf. Figure 16 top) – while the nodes remain in their initial positions.

Full optimization including free node positions has been tackled by *alternating* strategies. One possibility is to keep a subset of arcs and nodes fixed while efficiently optimizing the embedding of the others, and repeating this a sufficient number of times while rotating the choice of fixed and free arcs and nodes [TPP*11]. Another option is to alternatingly optimize arc as well as patch embeddings while keeping the nodes fixed (via simple linear system solves, as mentioned above), and improve the node positions for fixed patch embeddings (using a gradient descent strategy) [CK14b] (cf. Figure 16 bottom). The latter method is independent of the quality of the initial arc embeddings.

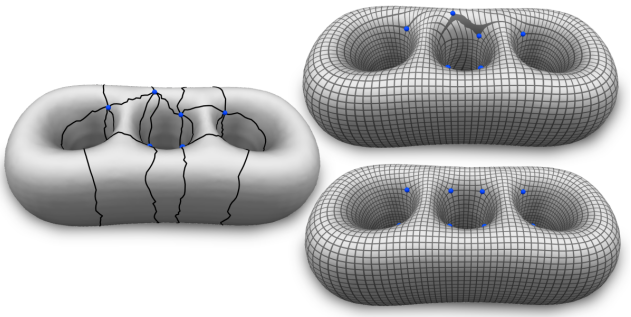


Figure 16: *Left: initial nodes and arcs. Top right: global parametrization, implying optimized arc and patch embeddings – but nodes remain unaltered. Bottom right: global parametrization after additionally optimizing node embeddings, implying patch embeddings with lower distortion, thus overall higher quality.*

5.3. Global Mesh Optimization

A simpler, but less robust, variant of optimization via a global parametrization is optimization via a quad mesh. Effectively, one could consider this a discretized version of the approach described in Section 5.2: based on an initial parametrization of the patches (e.g. by means of Tutte's embedding), create a regular grid of quads per patch, which together form a quad mesh aligned with the initial arcs [CBK12] (cf. Figure 17 left and middle). The chosen resolution of this mesh affects the efficiency and robustness of the following optimization.

Instead of iteratively optimizing the parametrization, which due to its singularities and chart-atlas nature is relatively involved, one can then apply quad mesh smoothing techniques [ZBX05], together with reprojection to avoid quad mesh vertices from moving off the surface. The final layout can be extracted as the base complex of the optimized quad mesh in the end (cf. Figure 17 right).

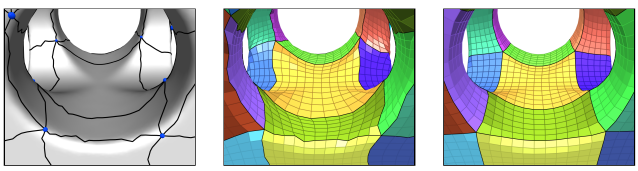


Figure 17: *Discrete, mesh based quad layout embedding optimization via constrained quad mesh smoothing.*

The simplicity of this approach comes with a price though: the reprojection of the vertices, and possibly the final layout, onto the surface is not unconditionally robust; the quad mesh resolution in relation to the surface feature size needs to be sufficiently fine.

6. Manual Influence

Due to the complexity and hardness of the quad layout generation problem, the results of current fully automatic methods are not always qualitatively suitable for all application scenarios. Guidance provided by additional user input can be leveraged to influence the results, to shape or *design* the layout according to specific requirements.

While an entirely manual layout construction [BVK08, KL96, MBVW95, AAB*88] is certainly an option, the involved global structural conditions and geometric interdependencies make this a challenging, tedious, and time consuming approach, even for experienced users. To cite Takayama et al. [TPSHSH13] in this regard: "it is often quite challenging even for professional artists to manually design a perfect quad mesh on the first try. Since the quality [...] is a global property, the correction of a single mistake might require regeneration of the entire mesh.". They hence proposed a system which provides various helpful guides and automatisms to reduce the user's workload – for the case of subdivision base mesh design. Similarly, semi-automatic, assisted approaches were proposed for the creation of quad layouts [TACSD06, TDN*12, CK14a, JLW10, ULP*15]. The user (directly or indirectly, e.g. via dual loops or a skeleton) places nodes and delineates the connecting arcs.

This user-controlled approach to the problem can be advantageous because layout design decisions might depend on the intended use of the layout and cannot always be derived automatically from geometry alone. An additional possibility to manually influence the resulting quad layout is to edit the result of an automatic method in a post-process using suitable structure-preserving operators [PZKW11, PBJW14].

7. Restricted Layout Classes

The methods discussed in the previous sections target general quad layouts. Effects of discretization and parameter choices aside, *any* quad layout could theoretically arise from many of these methods.

A number of methods that create quad layouts from restricted classes have been described in the literature as well. This is not necessarily always due to applications' demand for these specific types of layouts. In this sense, while the general methods could theoretically yield the qualitatively optimal layout, the restriction might unnecessarily limit the quality that can be achieved. On the other hand, this restriction may help to prevent bad or even worst-cases, which the general methods cannot strictly rule out, from arising. Furthermore, manual control (cf. Section 6) over the resulting layout can be easier for restricted classes of layouts. This, for instance, is exploited in skeleton-driven techniques, as described in the following.

7.1. Skeletal Layouts

Given a proper (homotopy equivalent) skeleton for a 3D model, designed manually or created automatically [TDS*16], structured quad layouts for the surface can be derived. Essentially, a partition of the surface into topological cylinders and punctured spheres can be associated with the skeleton: each bone corresponds to a topological cylinder, and each joint to a topological sphere, punctured such that the cylinders of the incident bones can be attached. For automatic skeletonization methods, this partition and correspondence is often a byproduct of the process.

Each cylindrical region can always be partitioned into a cyclic chain of k (arbitrary, but commonly chosen to be 4) quads (effectively by a map of the region to an open k-gonal prism). Then one is only left with defining compatible canonical quad partitions for n-times punctured spheres, such that the combination of these local quad layout templates for bones and joints form a global quad layout [JLW10, ULP*15].

7.2. Polycube Layouts

A number of methods deal with so-called polycube maps [THCM04], which involve a map of a surface onto a polycube surface, which consists of orthogonal polygon facets. As each orthogonal polygon, thus the entire polycube surface, can easily be subdivided into conforming quadrilateral patches, this allows to define a quad layout for the surface via the inverse map.

While these polycube maps are of interest for volumetric mapping and meshing purposes [GSZ11, HJS*14, FXBH16], the restriction to the specific sub-class of quad layouts that can result from such a map does not necessarily provide particular benefits for the surface case from an application perspective, and particularly not in terms of computational simplicity or robustness. A potential benefit could be the general absence of self-intersecting dual loops from polycube surface layouts, e.g. for volumetric meshing purposes [MH02, KBLK13].

8. Future Directions

Finally, we briefly discuss some of the major open problems and interesting future directions. While many of the presented methods have their own unanswered detail questions, e.g. regarding specific guarantees and properties, we focus here on the overarching aspects.

8.1. Optimality

Quad layout generation approaches treated so far in the literature are typically based on heuristics and simplified, abstracted objectives, not a true application-specific quality measure. This could be due to the fact that the actual measure can be quite hard to formalize, too hard to optimize for, or simply too application-specific, i.e. not generic enough for a research publication targeting a general audience.

As such, all the automatic methods proposed can rarely give strict guarantees concerning the quality and suitability of the result for any specific application scenario with hard requirements.

It would certainly be valuable to have methods available that are able to create a layout with specific guarantees, such as, for instance, a bound on the surface approximation error when the layout is used as a base domain for a particular spline-based surface representation, or a bound on the distortion when the layout is used as domain for a piecewise surface map. As finding the globally optimal layout satisfying certain constraints is likely to be intractable in a general case, and directly yielding any (non-optimal) layout fulfilling certain hard conditions is perhaps not within reach either, it would also be of benefit to have methods that are able to efficiently incrementally modify a quad layout, improving it towards a specific goal, ideally being able to guarantee that certain conditions will ultimately be met – albeit in a sub-optimal (non-parsimonious) manner.

8.2. Sub-Problem Integration

The decomposition of the quad layout generation problem into subproblems which are solved in sequence, is a popular and efficient strategy employed by many of the recent methods. It, for instance, allows the discrete, the combinatorial, and the continuous aspects of the problem to be tackled separately, by specialized optimization techniques.

On the other hand, higher quality could potentially be achieved if the entire problem would be solved at once, in an integrated manner. For instance, consider that the quality of a node configuration can only vaguely be judged without knowledge of also the arc connectivity and the patch embedding. As a simpler, but also more limited, alternative to an integrated optimization, one could consider making, in some form, retroactive adjustments to the node configuration during the arc construction, or adjustments to the nodes or arcs during the patch embedding optimization, so as to arrive at a layout of higher quality.

8.3. Hexahedral Layouts of 3D Solids

Just like quadrilateral layouts enable tensor product constructions in 2D (on surfaces of objects), *hexahedral layouts* enable this in 3D (in objects' interior volumes). In such partitions, each part (or *cell*) is topologically cubical. Some applications require such a three-dimensional (piecewise) representation of a solid's interior volume – for instance for simulations that cannot be expressed on the surface alone. A generalization of the discussed quad layout generation techniques to this hex layout case would be of high value.

Quite unfortunately, while the continuous embedding optimization techniques (cf. Section 5) could be adapted based on three-dimensional field-guided parameterization [NRP11], the most powerful node determination techniques (cf. Section 4.1) and nearly all connectivity determination techniques (cf. Section 4.2) do *not* generalize from 2D to 3D, as detailed in the following. As an alternative to hex layout construction from scratch, one could proceed incrementally, first constructing a surface quad layout and then extending it to a volume hex layout [Eri14], but only a restricted class of quad layouts is suitable and achieving high geometric layout quality poses a major challenge.

Cross Fields. The node determination technique based on cross fields (cf. Section 4.1.2) does not extend to 3D. While in 2D virtually any node configuration that can result from this method is valid, for the 3D analog based on *3D frame fields* [HTWB11] additional global constraints need to be met such that a valid singularity network (implying nodes and arcs) can be deduced [VCD*16]. Thus, so far manual singularity specification [NRP11] has been used, or attempts been made to repair invalid configurations in a post-process [LLX*12, JHW*14], though, to cite Jiang et al. [JHW*14], "it remains an open problem to give a sufficient condition".

Dual Loops. The concept of dual loops (cf. Section 4.2.3) extends to the three-dimensional case in the form of dual sheets [MBBM97]. The efficient generation of candidate dual sheets, however, is an unsolved problem. While minimal embedded loops can be generated using efficient Dijkstra-type techniques, minimal embedded surfaces (as natural model for good dual sheets) are not amenable to such efficient generation [Gra10]. A particular challenge is that dual sheets of unknown topology (arbitrary genus, arbitrary number of boundaries) must be dealt with, while dual loops are always simple curves.

Seamless Parametrization. A number of techniques discussed relies on a seamless surface parametrization (cf. Sections 4.2.2, 4.2.4, 5.2). For the surface case, a robust solution to the problem of obtaining such a parametrization is known [MPZ14]. Many components thereof (e.g. the field tracing or the motorcycle graph) have not yet been extended to 3D.

Pairing. Just like a triangulation can easily be obtained for a surface, a tetrahedralization can easily be obtained for a solid's volume. As we have seen, triangle layouts can often be converted into quad layouts (though not necessarily of high quality) by means of pairing (cf. Section 4.2.1). One can try to extend this and combine multiple tetrahedra to hexahedra. This, however, only works to some extent [SRUL16], yielding hexahedra dominant layouts.

Refinement. While Catmull-Clark refinement (cf. Section 4.2.1) turns any polygonal layout into a quad layout, polyhedral layouts are turned into hexahedral layouts by an analogous refinement only if all polyhedra are trivalent (i.e. three edges meet at each corner). This is the case for tetrahedra. Unfortunately, this one construction that actually does extend to 3D is typically the worst in terms of quality already in 2D; note that most nodes and arcs of the resulting hexahedral layout are irregular rather than regular.

Morse-Smale Complex. While in 2D a pure quad layout can be derived from scalar fields via Morse-Smale theory (cf. Section 4.1.1), the existence of multiple types of saddle-points in 3D scalar fields typically leads to Morse-Smale complexes with cells that are not hexahedral [LHJ*14].

Layout Simplification. We considered the simplification of fine quad layouts (or meshes) to coarser layouts (cf. Section 4.2.4). Recently, a similar technique has been proposed for hexahedral meshes [GDC15]. The robust generation of pure hexahedral meshes to begin with remains an issue [JHW*14, FXBH16] to be investigated.

9. Summary

The problem of partitioning surfaces into quadrilateral patches is not a simple, and not an easy problem. It comes with discrete, combinatorial, topological, and continuous degrees of freedom, global structural conditions and interdependencies, as well as a variety of non-trivial qualitative requirements. A multitude of techniques have been used and proposed to construct and optimize a quad layout's nodes, arcs, patches, and their embedding in the underlying surface.

For the nodes, there are approaches based on sampling, which are relatively simple, but often not particularly suited for quad patches, even less for conforming and parsimonious quad layouts. Approaches based on the surface's Gaussian curvature have proven powerful in this context and are employed by many recent methods.

For the construction of arcs, defining the layout's connectivity, the simplest methods are based on direct conversion of initial triangular layouts. This allows for robust implementations, but the results are typically not of high quality in terms of alignment and regularity. Direct, primal construction of quadrilateral connectivity involves a global combinatorial optimization problem, and the major challenge is the reduction of the infinite search space to a finite yet feasible subspace. Dual approaches allow for efficient connectivity construction in an incremental manner. The control over the layout's quality is less direct in this setting though, due to the indirect, dual perspective. Extremal or simplified integer grid maps allow to derive connectivity information from a quantized global parametrization. The main challenge in this context is the robust construction of an injective and seamless global parametrization.

For the optimization of patch embeddings, local approaches are simple but limited in their effectiveness. State-of-the-art methods employ a global parametrization, representing all the nodes', arcs', and patches' embeddings in the surface, which can iteratively be optimized while taking their interdependencies into account.

Directions for future work include the investigation of methods that enable the creation of layouts particularly suited for specific applications, ideally being able to provide strict guarantees, e.g. regarding error bounds. The automatic construction of 3D hexahedral layouts likewise is of significant interest, and the generalization of the discussed quad layouting techniques to the next dimension poses a multitude of open problems and interesting challenges.

References

[AAB*88] ANDERSSON E., ANDERSSON R., BOMAN M., ELMROTH T., DAHLBERG B., JOHANSSON B.: Automatic construction of surfaces with prescribed shape. *Computer-Aided Design 20*, 6 (1988), 317–324. 2, 15

[ACSD*03] ALLIEZ P., COHEN-STEINER D., DEVILLERS O., LÉVY B., DESBRUN M.: Anisotropic polygonal remeshing. *ACM Transactions on Graphics 22*, 3 (2003), 485–493. 5

[AdVDI05] ALLIEZ P., DE VERDIÈRE E. C., DEVILLERS O., ISENBURG M.: Centroidal voronoi diagrams for isotropic surface remeshing. *Graph. Models 67*, 3 (2005), 204–231. 6, 9

[AG03] ALLIEZ P., GOTSMAN C.: Recent advances in compression of 3D meshes. In *Proceedings of the Symposium on Multiresolution in Geometric Modeling* (2003), pp. 3–26. 2

[AUGA08] ALLIEZ P., UCELLI G., GOTSMAN C., ATTENE M.: Recent advances in remeshing of surfaces. In *Shape Analysis and Structuring, Mathematics and Visualization* (2008), Springer, pp. 53–82. 2

[BCE*13] BOMMES D., CAMPEN M., EBKE H.-C., ALLIEZ P., KOBBELT L.: Integer-grid maps for reliable quad meshing. *ACM Transactions on Graphics 32*, 4 (2013), 98:1–98:12. 5, 12, 13

[BDL10] BRAKHAGE K.-H., DAHMEN W., LAMBY P.: A unified approach to the modeling of airplane wings and numerical grid generation using B-spline representations. *Notes on Numerical Fluid Mechanics and Multidisciplinary Design 109* (2010), 239–263. 2

[BF98] BOROUCHAKI H., FREY P. J.: Adaptive triangular-quadrilateral mesh generation. *International Journal for Numerical Methods in Engineering 41*, 5 (1998), 915–934. 9

[BK01] BOTSCH M., KOBBELT L.: Resampling feature regions in polygonal meshes for surface anti-aliasing. *Computer Graphics Forum 20*, 3 (2001), 402–410. 5

[BLK11] BOMMES D., LEMPFER T., KOBBELT L.: Global structure optimization of quadrilateral meshes. *Computer Graphics Forum 30*, 2 (2011), 375–384. 11, 13

[BLP*13] BOMMES D., LÉVY B., PIETRONI N., PUPPO E., SILVA C., TARINI M., ZORIN D.: Quad-mesh generation and processing: A survey. *Computer Graphics Forum 32*, 6 (2013), 51–76. 1, 2, 3, 6

[Blu90] BLUM N.: A new approach to maximum matching in general graphs. In *Automata, Languages and Programming: Proc. 17th Int. Colloquium Warwick University* (1990), Springer Berlin Heidelberg, pp. 586–597. 9

[BMRJ04] BOIER-MARTIN I. M., RUSHMEIER H. E., JIN J.: Parameterization of triangle meshes over quadrilateral domains. In *Proc. SGP '04* (2004), pp. 197–208. 7, 9, 10

[Bom12] BOMMES D.: *Quadrilateral Surface Mesh Generation for Animation and Simulation*. PhD thesis, RWTH Aachen University, 2012. 2

[BVK08] BOMMES D., VOSSEMER T., KOBBELT L.: Quadrangular parameterization for reverse engineering. *Mathematical Methods for Curves and Surfaces* (2008), 55–69. 2, 6, 14, 15

[BZK09] BOMMES D., ZIMMER H., KOBBELT L.: Mixed-integer quadrangulation. *ACM Transactions on Graphics 28*, 3 (2009), 77:1–77:10. 5, 8, 10, 12, 13

[BZK12] BOMMES D., ZIMMER H., KOBBELT L.: Practical mixed-integer optimization for geometry processing. In *Proc. Curves and Surfaces* (2012), pp. 193–206. 8

[Cam14] CAMPEN M.: *Quad Layouts: Generation and Optimization of Conforming Quadrilateral Surface Partitions*. Shaker Verlag, 2014. 1, 8

[CBK12] CAMPEN M., BOMMES D., KOBBELT L.: Dual Loops Meshing: Quality Quad Layouts on Manifolds. *ACM Transactions on Graphics 31*, 4 (2012), 110:1–110:11. 11, 12, 14

[CBK15] CAMPEN M., BOMMES D., KOBBELT L.: Quantized global parametrization. *ACM Transactions on Graphics 34*, 6 (2015). 6, 11, 12, 13

[CC78] CATMULL E., CLARK J.: Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design 10*, 6 (1978), 350–355. 3, 10

[CDS10] CRANE K., DESBRUN M., SCHRÖDER P.: Trivial connections on discrete surfaces. *Computer Graphics Forum 29*, 5 (2010), 1525–1533. 7

[CIE*16] CAMPEN M., IBING M., EBKE H.-C., ZORIN D., KOBBELT L.: Scale-Invariant Directional Alignment of Surface Parametrizations. *Computer Graphics Forum 35*, 5 (2016). 5

[CJW*09] CLINE D., JESCHKE S., WHITE K., RAZDAN A., WONKA P.: Dart throwing on surfaces. In *Proceedings of the Twentieth Eurographics Conference on Rendering* (2009), pp. 1217–1226. 6

[CK14a] CAMPEN M., KOBBELT L.: Dual strip weaving: Interactive design of quad layouts using elastica strips. *ACM Transactions on Graphics 33*, 6 (2014), 183:1–183:10. 12, 15

[CK14b] CAMPEN M., KOBBELT L.: Quad layout embedding via aligned parameterization. *Computer Graphics Forum 33*, 8 (2014), 69–81. 14

[CSAD04] COHEN-STEINER D., ALLIEZ P., DESBRUN M.: Variational shape approximation. *ACM Transactions on Graphics 23*, 3 (2004), 905–914. 5, 9

[D'A00] D'AZEVEDO E. F.: Are bilinear quadrilaterals better than linear triangles? *J. Sci. Comput. 22*, 1 (2000), 198–217. 5

[DBG*06] DONG S., BREMER P.-T., GARLAND M., PASCUCCI V., HART J. C.: Spectral surface quadrangulation. *ACM Transactions on Graphics 25*, 3 (2006), 1057–1066. 5, 6

[dC76] DO CARMO M. P.: *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, NJ, 1976. 3

[DFG99] DU Q., FABER V., GUNZBURGER M.: Centroidal voronoi tessellations: Applications and algorithms. *SIAM Rev. 41*, 4 (1999), 637–676. 6

[DHM09] DAHMEN W., HOVHANNISYAN N., MÜLLER S.: *Adaptive Multiscale Methods for Flow Problems: Recent Developments, IGPM Report #293*. Tech. rep., RWTH Aachen, 2009. 2

[DS78] DOO D., SABIN M.: Behavior of recursive division surfaces near extraordinary points. *Computer-Aided Design 10*, 6 (1978), 356–360. 3

[DSC09] DANIELS J., SILVA C. T., COHEN E.: Semi-regular quadrilateral-only remeshing from simplified base domains. *Computer Graphics Forum 28*, 5 (2009), 1427–1435. 10

[DSSC08] DANIELS J., SILVA C. T., SHEPHERD J., COHEN E.: Quadrilateral mesh simplification. *ACM Transactions on Graphics 27*, 5 (2008), 148. 10

[DVPSH14] DIAMANTI O., VAXMAN A., PANOZZO D., SORKINE-HORNUNG O.: Designing *n*-PolyVector fields with complex polynomials. *Computer Graphics Forum 33*, 5 (2014). 8

[DVPSH15] DIAMANTI O., VAXMAN A., PANOZZO D., SORKINE-HORNUNG O.: Integrable PolyVector fields. *ACM Transactions on Graphics 34*, 4 (2015). 8

[ECBK14] EBKE H.-C., CAMPEN M., BOMMES D., KOBBELT L.: Level-of-detail quad meshing. *ACM Transactions on Graphics 33*, 6 (2014), 184:1–184:11. 8

[Edm65] EDMONDS J.: Maximum matching and a polyhedron with 0,1-vertices. *J. of Res. the Nat. Bureau of Standards 69B* (1965), 125–130. 9

[EGKT08] EPPSTEIN D., GOODRICH M. T., KIM E., TAMSTORF R.: Motorcycle Graphs: Canonical Quad Mesh Partitioning. *Computer Graphics Forum 27*, 5 (2008), 1477–1486. 6

[EH96] ECK M., HOPPE H.: Automatic reconstruction of B-spline surfaces of arbitrary topological type. In *Proc. SIGGRAPH 96* (1996), pp. 325–334. 2, 7, 9

[EHP02] ERICKSON J., HAR-PELED S.: Optimally cutting a surface into a disk. In *Proc. Symp. on Comp. Geometry* (2002), pp. 244–253. 3

[EKK*11] ESPERET L., KARDOŠ F., KING A. D., KRÁL' D., NORINE S.: Exponentially many perfect matchings in cubic graphs. *Advances in Mathematics 227*, 4 (2011), 1646–1664. 9

[Eri14] ERICKSON J.: Efficiently hex-meshing things with topology. *Discrete & Computational Geometry 52*, 3 (2014), 427–449. 16

[Far02] FARIN G.: *Curves and Surfaces for CAGD. A Practical Guide*. Morgan-Kaufmann, 2002. 2, 3

[FXBH16] FANG X., XU W., BAO H., HUANG J.: All-hex meshing using closed-form induced polycube. *ACM Transactions on Graphics 35*, 4 (2016), 124:1–124:9. 15, 16

[Gab90] GABOW H. N.: Data structures for weighted matching and nearest common ancestors with linking. In *Proc. ACM-SIAM Symposium on Discrete Algorithms* (1990), SODA '90, pp. 434–443. 9

[GDC15] GAO X., DENG Z., CHEN G.: Hexahedral mesh re-parameterization from aligned base-complex. *ACM Transactions on Graphics 34*, 4 (2015), 142:1–142:10. 16

[GGK02] GOTSMAN C., GUMHOLD S., KOBBELT L.: Simplification and compression of 3d meshes. In *Tutorials on Multiresolution in Geometric Modelling: Summer School Lecture Notes* (2002), Springer Berlin Heidelberg, pp. 319–361. 7

[GMSO14] GUNPINAR E., MORIGUCHI M., SUZUKI H., OHTAKE Y.: Feature-aware partitions from the motorcycle graph. *Computer-Aided Design 47* (2014), 85–95. 6

[Gra10] GRADY L.: Minimal surfaces extend shortest path segmentation methods to 3d. *IEEE Trans. Pattern Anal. Mach. Intell. 32*, 2 (2010), 321–334. 16

[GSZ11] GREGSON J., SHEFFER A., ZHANG E.: All-hex mesh generation via volumetric polycube deformation. *Computer Graphics Forum 30*, 5 (2011), 1407–1416. 15

[HCB05] HUGHES T. J. R., COTTRELL J. A., BAZILEVS Y.: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechaniccs and Engineering 194* (2005), 4135–4195. 2, 3

[Hei83] HEIGHWAY E.: A mesh generator for automatically subdividing irregular polygons into quadrilaterals. *IEEE Transactions on Magnetics 19*, 6 (1983), 2535–2538. 9

[HJS*14] HUANG J., JIANG T., SHI Z., TONG Y., BAO H., DESBRUN M.: ℓ1-based construction of polycube maps from complex shapes. *ACM Transactions on Graphics 33*, 3 (2014), 25:1–25:11. 15

[HTWB11] HUANG J., TONG Y., WEI H., BAO H.: Boundary aligned smooth 3D cross-frame field. *ACM Transactions on Graphics 30*, 6 (2011), 143:1–143:8. 16

[HWW*06] HE Y., WANG K., WANG H., GU X., QIN H.: Manifold T-spline. In *Proc. Geometric Modeling and Processing* (2006), pp. 409–422. 6

[HZM*08] HUANG J., ZHANG M., MA J., LIU X., KOBBELT L., BAO H.: Spectral quadrangulation with orientation and alignment control. *ACM Transactions on Graphics 27*, 5 (2008), 147. 5, 6

[JHW*14] JIANG T., HUANG J., WANG Y., TONG Y., BAO H.: Frame field singularity correction for automatic hexahedralization. *Transactions on Visualization and Computer Graphics, 20*, 8 (2014). 16

[JLW10] JI Z., LIU L., WANG Y.: B-mesh: A modeling system for base meshes of 3d articulated shapes. In *Proc. Pacific Graphics '10* (2010), pp. 2169–2178. 2, 15

[JSK91] JOHNSTON B. P., SULLIVAN J. M., KWASNIK A.: Automatic conversion of triangular finite element meshes to quadrilateral elements. *International Journal for Numerical Methods in Engineering 31*, 1 (1991), 67–84. 9

[JT73] JUCOVIČ E., TRENKLER M.: A theorem on the structure of cell-decompositions of orientable 2-manifolds. *Mathematika 20* (1973), 63–82. 8

[JTPS15] JAKOB W., TARINI M., PANOZZO D., SORKINE-HORNUNG O.: Instant field-aligned meshes. *ACM Transactions on Graphics 34*, 6 (2015), 189. 5, 10

[KBLK13] KREMER M., BOMMES D., LIM I., KOBBELT L.: Advanced automatic hexahedral mesh generation from surface quad meshes. In *Proceedings of the 22nd International Meshing Roundtable* (2013), pp. 147–164. 15

[KBZ15] KOVACS D., BISCEGLIO J., ZORIN D.: Dyadic T-mesh subdivision. *ACM Transactions on Graphics 34*, 4 (2015). 5, 6

[KCPS13] KNÖPPEL F., CRANE K., PINKALL U., SCHRÖDER P.: Globally optimal direction fields. *ACM Transactions on Graphics 32*, 4 (2013), 59. 8

[KL96] KRISHNAMURTHY V., LEVOY M.: Fitting smooth surfaces to dense polygon meshes. In *Proc. SIGGRAPH 96* (1996), pp. 313–324. 2, 15

[KLF15] KOWALSKI N., LEDOUX F., FREY P.: Automatic domain partitioning for quadrilateral meshing with line constraints. *Engineering with Computers 31*, 3 (2015), 405–421. 11

[KMZ11] KOVACS D., MYLES A., ZORIN D.: Anisotropic quadrangulation. *Computer Aided Geometric Design 28*, 8 (2011), 449–462. 5

[KNP07] KÄLBERER F., NIESER M., POLTHIER K.: Quadcover – surface parameterization using branched coverings. *Computer Graphics Forum 26*, 3 (2007), 375–384. 5, 11, 12, 13

[Kol09] KOLMOGOROV V.: Blossom v: a new implementation of a minimum cost perfect matching algorithm. *Mathematical Programming Computation 1*, 1 (2009), 43–67. 9

[Law76] LAWLER E.: *Combinatorial Optimization: Networks and Matroids*. Saunders College Publishing, Fort Worth, 1976. 9

[LHJ*14] LING R., HUANG J., JÜTTLER B., SUN F., BAO H., WANG W.: Spectral quadrangulation with feature curve alignment and element size control. *ACM Transactions on Graphics 34*, 1 (2014). 5, 6, 16

[LKH08] LAI Y.-K., KOBBELT L., HU S.-M.: An incremental approach to feature aligned quad dominant remeshing. In *Proc. Symp. Solid and Physical Modeling 2008* (2008), pp. 137–145. 10

[LL94] LEE C., LO S.: A new scheme for the generation of a graded quadrilateral mesh. *Computers & Structures 52*, 5 (1994), 847 – 857. 9

[LL10] LÉVY B., LIU Y.: L_p centroidal voronoi tessellation and its applications. *ACM Transactions on Graphics 29*, 4 (2010). 6, 9

[LLS01] LITKE N., LEVIN A., SCHRÖDER P.: Fitting subdivision surfaces. In *IEEE Visualization 2001* (2001), pp. 319–324. 2

[LLX*12] LI Y., LIU Y., XU W., WANG W., GUO B.: All-hex meshing using singularity-restricted field. *ACM Transactions on Graphics 31*, 6 (2012), 177:1–177:11. 16

[LRL06] LI W.-C., RAY N., LÉVY B.: Automatic and interactive mesh to t-spline conversion. In *Proc. SGP '06* (2006), pp. 191–200. 3, 5, 6, 14

[LVRL06] LI W. C., VALLET B., RAY N., LÉVY B.: Representing Higher-Order Singularities in Vector Fields on Piecewise Linear Surfaces. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 1315–1322. 8

[LWSF10] LI H., WEI L.-Y., SANDER P. V., FU C.-W.: Anisotropic blue noise sampling. *ACM Transactions on Graphics 29*, 6 (2010), 167:1–167:12. 6

[LXW*11] LIU Y., XU W., WANG J., ZHU L., GUO B., CHEN F., WANG G.: General planar quadrilateral mesh design using conjugate direction field. *ACM Transactions on Graphics 30*, 6 (2011). 5

[MBBM97] MURDOCH P., BENZLEY S., BLACKER T., MITCHELL S. A.: The spatial twist continuum: a connectivity based method for representing all-hexahedral finite element meshes. *Finite Elem. Anal. Des. 28* (1997), 137–149. 11, 16

[MBVW95] MILROY M. J., BRADLEY C., VICKERS G. W., WEIR D. J.: G1 continuity of B-spline surface patches in reverse engineering. *Computer-Aided Design 27*, 6 (1995), 471–478. 2, 15

[MH02] MÜLLER-HANNEMANN M.: Quadrilateral surface meshes without self-intersecting dual cycles for hexahedral mesh generation. *Comput. Geom. 22*, 1-3 (2002), 75–97. 15

[Mit00] MITCHELL S. A.: High fidelity interval assignment. *Int. J. Comput. Geometry Appl. 10*, 4 (2000), 399–415. 2

[MK95] MA W., KRUTH J.-P.: Parametrization of randomly measured points for least squares fitting of B-spline curves and surfaces. *Computer-Aided Design 27* (1995), 663–675. 2

[MK04] MARINOV M., KOBBELT L.: Direct anisotropic quad-dominant remeshing. In *Proc. Pacific Graphics '04* (2004), pp. 207–216. 10

[MPKZ10] MYLES A., PIETRONI N., KOVACS D., ZORIN D.: Feature-aligned T-meshes. *ACM Transactions on Graphics 29*, 4 (2010), 117:1–117:11. 6, 8, 12

[MPZ14] MYLES A., PIETRONI N., ZORIN D.: Robust field-aligned global parametrization. *ACM Transactions on Graphics 33*, 4 (2014). 5, 8, 11, 16

[MZ12] MYLES A., ZORIN D.: Global parametrization by incremental flattening. *ACM Transactions on Graphics 31*, 4 (2012). 3, 10, 12

[MZ13] MYLES A., ZORIN D.: Controlled-distortion constrained global parametrization. *ACM Transactions on Graphics 32*, 4 (2013). 5

[Nie12] NIESER M.: *Parameterization and Tiling of Polyhedral Surfaces*. PhD thesis, Freie Universität Berlin, 2012. 8

[NRP11] NIESER M., REITEBUCH U., POLTHIER K.: Cubecover – parameterization of 3d volumes. *Computer Graphics Forum 30*, 5 (2011), 1397–1406. 16

[PBJW14] PENG C.-H., BARTON M., JIANG C., WONKA P.: Exploring quadrangulations. *ACM Transactions on Graphics 33*, 1 (2014). 15

[PC06] PEYRÉ G., COHEN L. D.: Geodesic remeshing using front propagation. *Int. J. Comput. Vision 69*, 1 (2006), 145–156. 6, 9

[PCK04] PURNOMO B., COHEN J. D., KUMAR S.: Seamless texture atlases. In *Proc. SGP '04* (2004), pp. 65–74. 7, 9, 10, 14

[PPM*16] PIETRONI N., PUPPO E., MARCIAS G., SCOPIGNO R., CIGNONI P.: Tracing field-coherent quad layouts. *Computer Graphics Forum 35*, 7 (2016). 10, 11

[PPT*11] PANOZZO D., PUPPO E., TARINI M., PIETRONI N., CIGNONI P.: Automatic construction of quad-based subdivision surfaces using fitmaps. *IEEE Trans. Vis. Comput. Graph. 17*, 10 (2011). 10

[PPTSH14] PANOZZO D., PUPPO E., TARINI M., SORKINE-HORNUNG O.: Frame fields: Anisotropic and non-orthogonal cross fields. *ACM Transactions on Graphics 33*, 4 (2014), 134. 5

[PS98] POLTHIER K., SCHMIES M.: Straightest geodesics on polyhedral surfaces. In *Vis. and Math. '97*. Springer, 1998, pp. 135–150. 7

[PZ07] PALACIOS J., ZHANG E.: Rotational symmetry field design on surfaces. *ACM Transactions on Graphics 26*, 3 (2007). 8

[PZKW11] PENG C.-H., ZHANG E., KOBAYASHI Y., WONKA P.: Connectivity editing for quadrilateral meshes. *ACM Transactions on Graphics 30*, 6 (2011), 141. 15

[Rei95] REIF U.: A unified approach to subdivision algorithms near extraordinary points. *Comput. Aided. Geom. Des. 12* (1995). 3

[RLL*06] RAY N., LI W. C., LÉVY B., SHEFFER A., ALLIEZ P.: Periodic global parameterization. *ACM Transactions on Graphics 25*, 4 (2006), 1460–1485. 5, 10

[RLS*12] REMACLE J.-F., LAMBRECHTS J., SENY B., MARCHANDISE E., JOHNEN A., GEUZAINET C.: Blossom-quad: A non-uniform quadrilateral mesh generator using a minimum-cost perfect-matching algorithm. *Int. J. for Numerical Methods in Engineering 89*, 9 (2012). 9

[RNLL10] RAY N., NIVOLIERS V., LEFEBVRE S., LÉVY B.: Invisible seams. In *Proc. Eurographics Symposium on Rendering* (2010), pp. 1489–1496. 3

[RP17] RAZAFINDRAZAKA F. H., POLTHIER K.: Optimal base complexes for quadrilateral meshes. *Computer Aided Geometric Design* (2017). 10, 11

[RRP15] RAZAFINDRAZAKA F. H., REITEBUCH U., POLTHIER K.: Perfect matching quad layouts for manifold meshes. *Computer Graphics Forum 34*, 5 (2015), 219–228. 10, 11

[RVAL09] RAY N., VALLET B., ALONSO L., LÉVY B.: Geometry-aware direction field processing. *ACM Transactions on Graphics 29*, 1 (2009). 8

[RVLL08] RAY N., VALLET B., LI W. C., LÉVY B.: N-symmetry direction field design. *ACM Transactions on Graphics 27*, 2 (2008), 10:1–10:13. 8

[SB96] SPEKREIJSE S., BOERSTOEL J. W.: Multiblock grid generation. part 2: Multiblock aspects. In *VKI Lecture Series 1996-06*. von Karman Institute for Fluid Dynamics, 1996, pp. 1–39. 2

[SCF*04] SEDERBERG T. W., CARDON D. L., FINNIGAN G. T., NORTH N. S., ZHENG J., LYCHE T.: T-spline simplification and local refinement. *ACM Transactions on Graphics 23*, 3 (2004). 6

[SFL*08] SEDERBERG T. W., FINNIGAN G. T., LI X., LIN H., IPSON H.: Watertight trimmed NURBS. *ACM Transactions on Graphics 27*, 3 (2008). 6

[SRUL16] SOKOLOV D., RAY N., UNTEREINER L., LÉVY B.: Hexahedral-dominant meshing. *ACM Transactions on Graphics 35*, 5 (2016), 157:1–157:23. 16

[SZBN03] SEDERBERG T. W., ZHENG J., BAKENOV A., NASRI A.: T-splines and T-NURCCs. *ACM Transactions on Graphics 22*, 3 (2003), 477–484. 6

[TA93] TAM T. K. H., ARMSTRONG C. G.: Finite element mesh control by integer programming. *Int. J. Numerical Methods in Engineering 36* (1993), 2581–2605. 2, 6

[TACSD06] TONG Y., ALLIEZ P., COHEN-STEINER D., DESBRUN M.: Designing quadrangulations with discrete harmonic forms. In *Proc. SGP '06* (2006), pp. 201–210. 10, 14, 15

[TDN*12] TIERNY J., DANIELS J., NONATO L. G., PASCUCCI V., SILVA C.: Interactive quadrangulation with reeb atlases and connectivity textures. *IEEE TVCG 18* (2012), 1650–1663. 15

[TDS*16] TAGLIASACCHI A., DELAMÉ T., SPAGNUOLO M., AMENTA N., TELEA A.: 3D skeletons: A state-of-the-art report. *Computer Graphics Forum 35*, 2 (2016), 573–597. 15

[THCM04] TARINI M., HORMANN K., CIGNONI P., MONTANI C.: Polycube-maps. *ACM Transactions on Graphics 23*, 3 (2004). 15

[TPC*10] TARINI M., PIETRONI N., CIGNONI P., PANOZZO D., PUPPO E.: Practical quad mesh simplification. *Computer Graphics Forum 29*, 2 (2010), 407–418. 10

[TPP*11] TARINI M., PUPPO E., PANOZZO D., PIETRONI N., CIGNONI P.: Simple quad domains for field aligned mesh parametrization. *ACM Transactions on Graphics 30*, 6 (2011). 11, 13, 14

[TPSHSH13] TAKAYAMA K., PANOZZO D., SORKINE-HORNUNG A., SORKINE-HORNUNG O.: Sketch-based generation and editing of quad meshes. *ACM Transactions on Graphics 32*, 4 (2013), 97:1–97:8. 15

[Tur92] TURK G.: Re-tiling polygonal surfaces. In *SIGGRAPH '92* (1992), pp. 55–64. 6

[ULP*15] USAI F., LIVESU M., PUPPO E., TARINI M., SCATENI R.: Extraction of the quad layout of a triangle mesh guided by its curve skeleton. *ACM Transactions on Graphics 35*, 1 (2015), 6. 15

[VCD*16] VAXMAN A., CAMPEN M., DIAMANTI O., PANOZZO D., BOMMES D., HILDEBRANDT K., BEN-CHEN M.: Directional field synthesis, design, and processing. *Computer Graphics Forum 35*, 2 (2016). 8, 11, 12, 16

[VZ01] VELHO L., ZORIN D.: 4-8 subdivision. *Comput. Aided Geom. Des. 18*, 5 (2001), 397–427. 10

[WZXH12] WANG W., ZHANG Y., XU G., HUGHES T.: Converting an unstructured quadrilateral/hexahedral mesh to a rational t-spline. *Computational Mechanics 50*, 1 (2012), 65–84. 6

[YLL*09] YAN D.-M., LÉVY B., LIU Y., SUN F., WANG W.: Isotropic remeshing with fast and exact computation of restricted voronoi diagram. In *Proc. SGP '09* (2009), pp. 1445–1454. 9

[ZBX05] ZHANG Y., BAJAJ C., XU G.: Surface smoothing and quality improvement of quadrilateral/hexahedral meshes with geometric flow. In *Proc. 14th Int. Meshing Roundtable* (2005), pp. 449–468. 14

[ZHLB10] ZHANG M., HUANG J., LIU X., BAO H.: A wave-based anisotropic quadrangulation method. *ACM Transactions on Graphics 29*, 4 (2010), 118:1–118:8. 5, 6

[ZZY16] ZHANG S., ZHANG H., YONG J.-H.: Automatic quad patch layout extraction for quadrilateral meshes. *Computer-Aided Design and Applications 13*, 3 (2016), 409–416. 10, 11