Dual Strip Weaving: Interactive Design of Quad Layouts using Elastica Strips

Marcel Campen*
RWTH Aachen University

Leif Kobbelt*
RWTH Aachen University

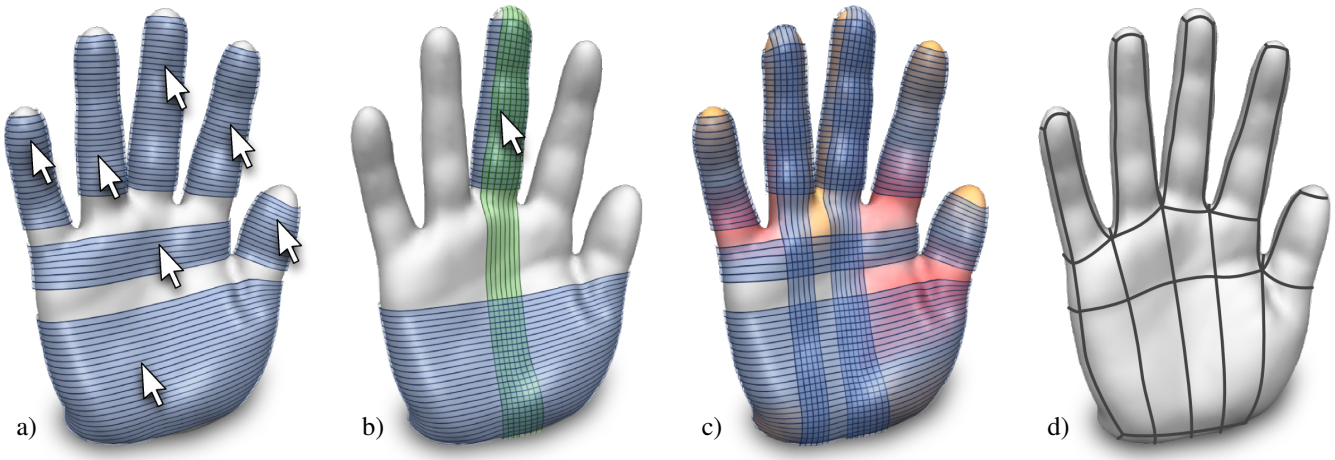


Figure 1: *Overview of our* Dual Strip Weaving *approach for the design of quadrilateral patch layouts. a) When hovering over the object, the user is immediately presented with the best elastica strip (visualized using a stripe pattern) at the current pointer position. It can be selected and fixed with a single click. b) Fixed strips (blue) constrain the design space; only compatible strips are offered next (green). c) Indicators based on color-coding and stripe patterns guide the user to regions where modifications are recommended for the benefit of layout quality. d) Finally, the implied quad layout structure is derived from a collection of strips. The accompanying video shows the entire process.*

Abstract

We introduce *Dual Strip Weaving*, a novel concept for the interactive design of quad layouts, i.e. partitionings of freeform surfaces into quadrilateral patch networks. In contrast to established tools for the design of quad layouts or subdivision base meshes, which are often based on creating individual vertices, edges, and quads, our method takes a more global perspective, operating on a higher level of abstraction: the atomic operation of our method is the creation of an entire cyclic strip, delineating a large number of quad patches at once. The global consistency-preserving nature of this approach reduces demands on the user's expertise by requiring less advance planning. Efficiency is achieved using a novel method at the heart of our system, which automatically proposes geometrically and topologically suitable strips to the user. Based on this we provide interaction tools to influence the design process to any desired degree and visual guides to support the user in this task.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling;

Keywords: quad mesh, base complex, segmentation, partition, subdivision, parameterization, edge flow

Links: DL PDF

*e-mail: (campen|kobbelt)@cs.rwth-aachen.de

1 Introduction

Since the early days of Computer-Aided Geometric Design, the partitioning of a surface into (preferably few) quadrilateral patches (or conversely: its composition thereof) has been an essential requisite and fundamental challenge. It is the tensor product nature of smooth surface representations like Bézier, B-Spline, or NURBS patches that established the need for such structures. Especially for reverse engineering purposes, the efficient creation of such partitionings has always been an important challenge, as outlined by [Li et al. 2006]. Quad meshes, which have seen an increase in popularity in recent years, sparked new interest in this problem. This is due to the fact that *semi-regular* quad meshes, which contain an underlying coarse quadrilateral base structure, i.e. which are a regular refinement of a *quad layout*, provide advantages for various application cases, as detailed in a recent survey [Bommes et al. 2013a].

A number of methods that tackle (variants of) this problem in an *automatic* manner have been proposed [Eck and Hoppe 1996; Boier-Martin et al. 2004; Dong et al. 2006; Daniels et al. 2009; Tarini et al. 2011; Bommes et al. 2011; Campen et al. 2012; Bommes et al. 2013b]. Suitability of these methods' results depends on the application context. An inherent issue is that a good quad layout generally is a compromise [Campen et al. 2012] that has to balance coarseness, patch rectangularity, feature and principal curvature alignment, and possibly further objectives. The precise formalization of the relative importances of these aspects and further requirements and side conditions in a certain application context is not an easy task, nor is automatically finding the optimal layout, assuming such formal description and quality measure was available.

The field of quad mesh generation is related but essentially different: in this field one typically aims for rather uniformly sized and shaped quads instead of a coarse network of potentially non-uniform patches whose dimensions are rather implied by the geometry and global structural interdependencies. Despite these differences, this field faces a very similar issue: a compromise be-

tween alignment, orientation, and element shape needs to be found [Bommes et al. 2009]. A solution which proved successful in practice was the inclusion of the user in the process – instead of aiming for full automation. Major 3D sculpting packages like Pixologic's ZBrush or Pilgway's 3D-Coat have recently been equipped with quad remeshing features which follow this paradigm and rely on high-level user interaction, e.g. regarding the specification of alignment and element sizing. This allows the user to tune the result to meet the given requirements – even if no formal description thereof, or no specialized optimization method therefor is available.

Unfortunately, while methods for *quad mesh generation* allow for user influence (regarding edge flow, irregular vertex configurations, element sizing, element anisotropy, etc.) [Bommes et al. 2009; Zhang et al. 2010; Kovacs et al. 2011], even though not always at interactive rates, existing methods targeted at the problem of *quad layout generation* (e.g. [Eck and Hoppe 1996; Boier-Martin et al. 2004; Tarini et al. 2011; Campen et al. 2012]) neither are intended for nor provide means for adequate user interaction.

1.1 Contribution

We introduce *Dual Strip Weaving*, a novel concept for the computer-aided interactive design of quad layout structures on surfaces. Our system features novel interaction tools and provides guides to effectively support the design process. It is based on a dual perspective: a quad layout's structure can be expressed by its dual graph, consisting of dual edge loops (cf. Figure 2). This view better exhibits the global connectivity-related interdependencies of quad-only layouts [Murdoch et al. 1997]. In our work we enrich these topological *dual loops* with a geometric dimension, turning them into *dual strips*. We furthermore show that the modeling of dual loops by means of geodesics, as done in previous work [Campen et al. 2012], is unsuited for the purpose of user-guided design and instead model our dual strips based on discrete, bending energy-minimizing *elastica* [Bruckstein et al. 2001]. This allows us to automatically propose geometrically and topologically suitable strips to the user, enabling an efficient workflow.

The contribution of our work can be summarized as follows:

- A fast method for the construction of *constrained elastica* on surfaces, without any restrictions on topology or homotopy.
- Novel support for boundaries and symmetries in the dual loops/strips framework.
- Interaction tools and metaphors for incremental layout design, supported by automatic background computations.
- Evaluation of intermediate design states and visualization of the assessment to guide the user.

2 Related Work

A number of powerful methods for quad mesh generation have been introduced [Bommes et al. 2013a]. Several of them, especially those based on grid parameterizations [Kälberer et al. 2007], allow for user guidance, for instance regarding alignment [Bommes et al. 2009], connectivity [Myles et al. 2010], or element sizing and anisotropy [Zhang et al. 2010; Kovacs et al. 2011]. These possibilities made such methods amenable for practical application, as mentioned above.

For the generation of quad layouts, different methods, with a stronger focus on coarseness than on element uniformity, have been proposed [Eck and Hoppe 1996; Boier-Martin et al. 2004; Dong et al. 2006; Daniels et al. 2009]. More recent works in this area

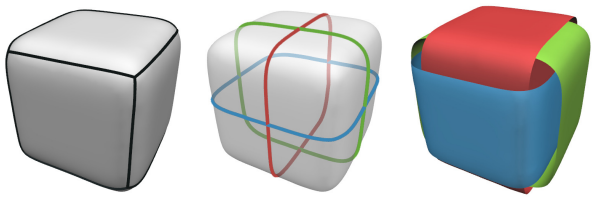


Figure 2: *Basic concepts: quad layout, dual loops, dual strips.*

[Campen et al. 2012; Tarini et al. 2011; Bommes et al. 2011] furthermore take alignment to features and principal directions as well as patch shape quality into account. User interaction is not provided for in these approaches.

Methods for quad layout construction which *do* involve the user, can be categorized based on the degree of user effort vs. automation. In some works a completely manual creation of the layout, i.e. drawing of the individual layout vertices and edges on a surface is described or assumed [Krishnamurthy and Levoy 1996; Bommes et al. 2008]. In other works some support is provided, e.g. once the vertices are positioned, edges between them can be proposed based on principal directions [Tong et al. 2006], or once vertices to be connected are indicated by the user, the corresponding edges can be realized as geodesics [Li et al. 2006].

Construction approaches for subdivision base meshes are also related, but the difference between quad layouts and such base meshes must not be missed: while, for instance, a torus can easily be represented by a quad layout with just one patch, a subdivision base mesh clearly needs to be finer. Furthermore, a small number of non-quad faces can be acceptable in subdivision base meshes, but not in quad-only layouts. Such meshes are often created by drawing individual vertices and edges, sometimes aided by operators enabling the creation of multiple related elements at once. Recently, a sketch-based system for a more efficient design of subdivision base meshes was described [Takayama et al. 2013]. It provides various helpful guides and automatisms to reduce the user's workload. Specifically for character models a skeleton-based approach can be of value, as shown by [Ji et al. 2010].

3 Concept

A design process with a user in the loop necessarily proceeds incrementally. This raises the question of what nature the increments should be, or in other words:

What should be the atomic operations provided to the user?

In established modeling tools this is often answered like: the user can create vertices, edges, and faces, and on a somewhat higher level multiple faces can be created at once using splitting, extrusion, and similar operators. In these cases, the user deals directly with the individual elements of the layout.

A drawback of this approach is its fine granularity: the operators are very local. Quad meshes and quad layouts, however, generally have a rather constrained global structure [Murdoch et al. 1997]. As this fact is not reflected in the local operators, it is up to the user to plan ahead such that in the end all the locally constructed elements meet up globally in a desirable manner. Large parts of the design might have to be redone when the set of created quads comes to form a non-quad region which cannot be quadrangulated without adverse side-effects (irregular vertices, excessive refinement, etc.). To cite [Takayama et al. 2013]: "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."

Inspired by the work of [Campen et al. 2012], who used global dual loops for automatic quad layout construction, we propose to use **the creation of an entire dual strip as *the* atomic operation** in our interactive design system. In contrast to the established local operators, this operator has a kind of built-in global consistency (no non-quad patches are ever generated), effectively reducing the burden of "planning ahead" for the user.

3.1 Dual Strips

A dual loop corresponds to a cyclic chain of quads in the primal layout – cf. [Campen et al. 2012] for foundations and detailed definitions. The union of all the quad patches of such chain we call *dual strip*, cf. Figure 2. We may say a dual loop is the spine of its dual strip. Note that a quad patch of the primal layout corresponds to an intersection region of two dual strips, crossing transversally. From this point of view our goal of partitioning a surface into quad patches is equivalent to **doubly-covering the surface with dual strips** where all strip intersections are transversal.

A practical analogon which illustrates this idea is basketry, i.e. the weaving, or more specifically plaiting, of baskets from strips of plant materials like bark, straw, or flax. The figure on the left shows an example (courtesy of Jonas Hasselrot). Notice how every quad is covered by two crossing strips. At the bottom corners of the basket it can be observed that irregular "vertices" (here of valence 3) can be formed using this technique, too. In this sense, the underlying conceptual idea of our system can be seen as weaving dual strips on a given surface until it is covered. Note, though, that our strips are of variable width and the layer-alternation of the woven strips, which serves stability in practice, is of no meaning in our case. This process of *Dual Strip Weaving* is computationally supported in our system, e.g. by proposing optimal routes or choosing appropriate widths for strips.

Note that building the layout based on dual loops or strips does not restrict the class of designable layouts in any way – for every quad-only layout there is an equivalent collection of dual loops/strips.

4 Interactive Workflow

We begin by describing the design workflow in order to provide a high-level understanding of the system and its core concepts. The accompanying video shows the system in action. As stated above, the creation of a dual strip is the fundamental operation in our system. Instead of having the user draw such strips by hand, the central idea of our system is to compute suitably optimized dual strips automatically and propose them to the user. Intuitive tools to select, edit, or model these strips are then made available to provide full flexibility, while still keeping the user workload low. Technical details of the involved algorithms for dual strip computation follow in Sections 5 and 6.

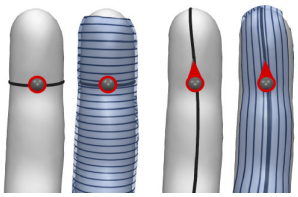
4.1 Tools and Metaphors

Simply **hovering** over the surface with the mouse pointer, the user is immediately presented with the best possible dual strip which runs through this point. This dual strip is constantly updated as the user moves the pointer. See Figure 1a.

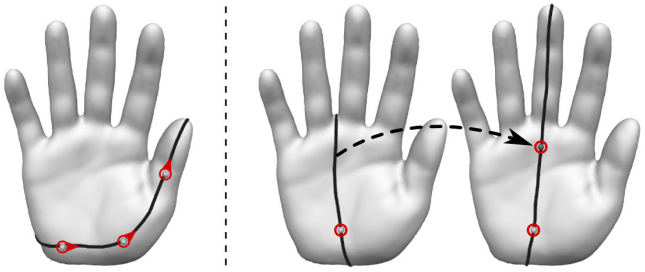
Using the **mouse wheel** the user can furthermore browse good alternative dual strips which run through the same point but take different routes over the surface, in order of descending quality.

These automatically proposed strips (computed and rated as described in Section 5), provide a rich fundament already sufficient for the construction of complete dual layouts. In order to provide full design flexibility to the user we furthermore enable the modeling and editing of dual strips using the following metaphors. In these manipulation modes it proved advantageous for clarity to display the strip in form of its representative spinal loop (the automatically chosen strip extent is not subject to user modification anyway; it serves purposes of "coverage" visualization).

By **clicking**, the user creates an anchor point and the best dual strip through this point is shown (the one which was already shown when the user hovered over this point). By **dragging** a directional anchor is created instead, producing a dual strip which runs through this point, interpolating the specified direction. The direction can also be altered interactively in order to explore the space of possible dual strips through the anchor point.



By clicking (or dragging) at further points on the surface, additional anchors (of positional or directional kind) can be placed. The best dual loop **interpolating** these anchors is then shown.



Another metaphor that can be used to conveniently edit the shape of a dual loop is **grabbing**. The user can grab the current loop at any point and drag it to another position, implicitly creating an additional anchor to be interpolated. This can be seen in analogy to modern route planning applications whose interfaces offer similar grab-and-drag tools to manipulate proposed routes.

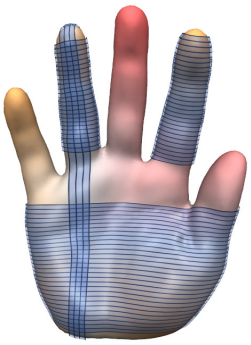
When the user begins creating a new dual strip, the existence of already created strips is taken into account. The best way for two strips to cross is orthogonally; at least they should be crossing transversally, not touching tangentially [Campen et al. 2012]. The system only proposes strips which respect this, favoring orthogonality, and which are furthermore not topologically equivalent to already existing ones, cf. Figure 1b.

4.2 Assessment & Feedback

The provided tools enable the flexible design of dual strip collections, which via dualization imply a primal quad layout structure (determination of its actual geometric embedding in detail is considered in Section 7). Especially when designing rather coarse layouts it can, however, be quite hard to visually judge the appropriateness and quality of the dual strip collection during design, because large patches that wrap around the underlying object are not visible in their entirety. We thus equip our system with visual indicators guiding the user in this regard.

The dual strips are visualized by a pattern of **quasi-parallel loops** in order to indicate their directional orientation ("edge flow"). Where two strips cross, these patterns overlap, forming a **grid**. This indicates that the corresponding region is *doubly-covered* as desired.

The user does not need to doubly-cover every part of the surface – uncovered regions only hint at sub-optimally shaped quads in the primal layout, but this can be a desired trade-off for layout simplicity. However, every region enclosed by loops at least needs to be disc-homeomorphic, otherwise no valid layout is implied. Regions with **non-disc topology** are hence highlighted in red to indicate that further strips are required to split this region.



When no region is red anymore, the set of dual strips is *topologically sufficient* and implies a valid primal quad layout. The addition of further strips is, however, often desired in order to refine the layout and achieve better geometric layout quality. Our system guides the user to those regions where this is particularly advisable. Regions which contain multiple pronounced **local extrema of Gaussian curvature** are highlighted in orange, indicating that it would be beneficial (though not mandatory) to add further strips there. This is motivated by the fact that such extrema are best represented by separate (irregular) vertices of the quad layout (thus separate dual regions) – having (some of) them lie in the interior of a quad patch would lead to low geometric patch quality.

In addition, we highlight in yellow those regions with a **mismatch between curvature and valence**: Ideally, the valence v of a region should be related to its total curvature K by $K = (4 - v)\frac{\pi}{2}$ for the sake of patch quality [Campen et al. 2012]. If the actual valence differs by more than 1, we indicate this situation for information. Note that this is mainly relevant for highly edited strips – the automatically proposed strips rarely lead to such situations due to the inherent favoring of orthogonality and alignment (cf. Section 5).

5 Elastica on Surfaces

The practicality and efficiency of the described interactive workflow depends on the geometric and structural quality of the automatically generated loops and strips. Given a surface M, a dual loop on it should ideally (cf. [Campen et al. 2012])

1) be aligned with principal directions of M and
2) have low geodesic curvature to facilitate a good quad shape.

Furthermore, for the sake of layout coarseness, it should rather

3) be short than wind around the whole object several times.

It should further respect, i.e. interpolate, the specified anchors of positional and directional kind (cf. Section 4). We generate dual strips in two steps: a dual loop is constructed to serve as spine (Section 5) and is then extended to an appropriate dual strip (Section 6).

5.1 Field-Guided Geodesic Loops

In their Dual Loops Meshing (DLM) method, [Campen et al. 2012] model dual loops as anisotropic geodesics with respect to a prescribed guiding cross field [Palacios and Zhang 2007; Bommes et al. 2009] which this method takes as input. This field is assumed to be aligned to stable principal directions and smooth otherwise, and in this way jointly promotes the alignment (1) and low geodesic curvature (2) of the constructed loops.

Unfortunately, this approach taken by DLM is not amenable to interactive user-guided design. The prescribed field already considerably restricts the space of representable layouts: dual loops constructed by DLM are essentially "bound" to the field. Hence, the user is not free to create loops as desired – geometrical as well as topological restrictions apply. In particular, the singularities of the field already completely define the number and approximate position of the result layout's vertices (up to local merging). But not only the number and configuration of vertices is fixed, also the connectivity of these vertices, i.e. the edges of the layout, is subject to restrictions induced by the fixed underlying field. In detail, only those loops along which the prescribed field has zero holonomy [Lai et al. 2010] can be created by the DLM approach.

For design purposes such broad restrictions are hardly communicable to the designer. In the following we hence present a field-less approach allowing for arbitrary loops. It is further able to take multiple user design constraints (the specified anchors) into account.

5.2 Elastica Loops

A suitable model for the (closed) curves $\ell : [0, L] \to M$ in arc-length parameterization we are looking for is the objective

$$c(\ell) = \int_0^L 1 + \alpha\, q_{\ell(t)}(\ell'(t)) + \beta\, \kappa_\ell(t)^2 dt \to \min \qquad (1)$$

subject to (positional and directional) constraints. Here the term 1 penalizes length, $q : TM \to \mathbb{R}$ penalizes deviation from principal directions (cf. Section 5.5), and the last term penalizes geodesic curvature $\kappa_\ell(t)$. A curve minimizing the bending energy $\int \kappa_\ell(t)^2 dt$ subject to positional and directional constraints is called *elastica* (or, depending on the context, *spline*), going back to Leonhard Euler and Jacob Bernoulli. Our functional in addition includes means to soft-constrain local direction via q, with the parameters α and β expressing the relative weighting of these objectives.

On surfaces, embedded elastica can be found using various methods of variational nature, like Active Contour and Snake models [Lee and Lee 2002; Bischoff et al. 2005] or the constrained spline optimization of [Hofer and Pottmann 2004]. These methods require an initialization and then strive to find a local optimum in the same homotopy class as the initial curve or loop[1]. To maximally support the user, we want to avoid the need for an initialization and would in particular like to find the optimum over all homotopy classes.

Hence, we design an algorithm, based on combinatorial optimization, which is able to find (approximations of) global optima of certain curve functionals, and this over all curve homotopy classes, i.e. without prescribed homotopy (cf. Figure 3). It is based on finding constrained minimum weight cycles in special graph structures.

To this end, all metric information must be modeled as graph edge weights. Unfortunately, in the case of a surface graph (e.g. a triangulation of the surface) only first-order differential quantities can be taken into account in such an approach: the weight of an edge has to be computed in advance from local information only, i.e. we can at most build the difference of its two end node positions and compute the weight from this vector's length and direction (and the node positions). Curvature, as in our objective (1), obviously is not accessible per-edge in this way.

5.3 Elastica Graph

If, however, we take a graph whose nodes are not a sample of the surface M, but of the tangent bundle TM, the nodes already contain first-order information, such that second-order properties,

[1] The spline method of [Panozzo et al. 2013] does not require initialization, but is essentially based on (pseudo-)geodesics, leading to similar homotopy class restrictions. In particular, it cannot construct (non-degenerate) loops from just one constraint point – the most relevant case here.

Figure 3: *Our optimization algorithm for elastica on surfaces has* free homotopy*: depending on the orientation of the directional constraint (red dot with arrow), loops from differing homotopy classes are obtained because the optimum is searched over all classes.*

i.e. curvature, can be computed per edge: given two adjacent nodes with position and tangential direction each, we are able to evaluate $\int \kappa(t)^2 dt$ for an imagined curve interpolating this Hermite data.

We are free to chose any sampling of TM and a connectivity to construct a graph approximation of the tangent bundle. A natural and intuitively accessible way is to take some digraph g approximating M and then form its *derivative* g' (also called *line graph* or *adjoint*) [Beineke 1968]: a node of this graph is a *directed* edge of g and can hence readily be identified with a point $p \in M$ (the edge's midpoint) and a direction $d \in T_pM$ (the directed edge's direction). Two g'-nodes are connected iff the corresponding directed g-edges are adjacent, forming a directed path of length two (cf. Figure 4). For the special case of planar regular grid graphs such line graph or related product graph constructions have successfully been applied for curvature-regularized image segmentation purposes [Schoenemann and Cremers 2007; Schoenemann et al. 2011].

As underlying graph g we choose an extended neighborhood graph built from a triangulation $\mathcal{T} = (V, E, F)$ of M, where two nodes are connected (by two directed edges, realized as geodesics on M) iff they have graph distance $\leq k$ in $\mathcal{T}$. We found $k = 4$ to sufficiently increase the angular resolution such that the resulting elastica are visually smooth. In this case, the cardinality of the node sets is related as follows: $|V_g| = |V|$, with an average valence of 60, and $|V_{g'}| \approx 60|V|$, with an average valence of 60. Note that g'-edges connecting g-edges which form a large geodesic angle can be omitted (e.g. the red and purple ones in Figure 4) – the corresponding curves have large geodesic curvature, thus are expendable in light of the bending energy minimization goal. We use a generous limit of 30°, reducing the average valence of g'-nodes to 10.

Curved Edges The question remains how a g'-edge $e' = (e_0, e_1)$ should be interpreted geometrically, i.e. which curve's geodesic curvature should be measured to obtain edge weights for e'. Let's first assume a planar configuration. One option is to compose a circular arc of maximal radius and a straight line segment, connecting the midpoints of e_0 and e_1 (cf. Figure 4). For this unique curve we calculate $\int \kappa^2 = 2\gamma \tan\frac{\gamma}{2} / \min(|e_0|, |e_1|)$, where γ is the angle between g-edges e_0 and e_1 at their common node, $|\cdot|$ the length of an edge. As for its approximation

$$\frac{\gamma^2}{\min(|e_0|, |e_1|)} =: \kappa^2(e')$$

(using Taylor expansion $\tan(\gamma) = \gamma + O(\gamma^3)$) favorable convergence properties have been proved (i.e. using graph refinement, continuous elastica can be approached) [Bruckstein et al. 2001], we opt for this choice. To generalize to non-planar configurations we only need to measure γ as *geodesic* angle on M, i.e. in a tangent plane at the common node of e_0 and e_1. This effectively unrolls the configuration to the tangent plane, masking out normal curvature, such that only the geodesic curvature is measured as intended.

The length of a g'-edge e' realized as described above evaluates to

$$|e'| = \frac{\gamma \min(|e_0|, |e_1|)}{2 \tan\frac{\gamma}{2}} + \frac{||e_0| - |e_1||}{2}.$$

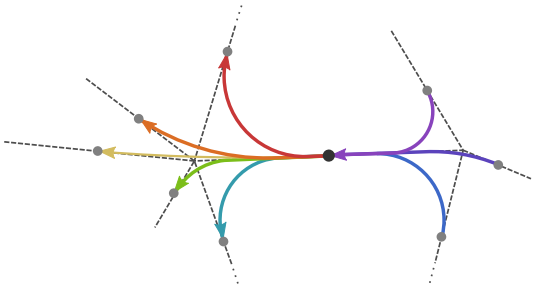


Figure 4: *Illustration of one directed node's (black) incoming and outgoing curved edges in the elastica graph g'. The underlying surface graph g is shown dashed. The g'-nodes lie at g-edge centers.*

The function q from (1), promoting principal direction alignment, we evaluate for e' using the trapezoid rule, sampling at the incident g'-nodes n_0, n_1 (midpoints of g-edges e_0 and e_1), leading to

$$q(e') := \frac{1}{2}\left(q_{n_0}(e_0) + q_{n_1}(e_1)\right)|e'|.$$

Using these definitions we can build a discrete version of (1) for the closed curve formed by a cyclic chain E of g'-edges:

$$c(E) = \sum_{e' \in E} |e'| + \alpha\, q(e') + \beta\, \kappa^2(e') =: \sum_{e' \in E} w(e') \qquad (2)$$

5.4 Constructing Discrete Elastica

Minimizers of (2) are minimum weight cycles in the elastica graph g'. The global optimum could be found using a variant of the Floyd-Warshall algorithm. However, we are not interested in this unconstrained optimizer, but would like the loop to interpolate the specified anchors to account for user influence. Therefore, we design an algorithm based on nested minimum weight path problems which can very efficiently be solved using Dijkstra's algorithm.

In detail, we would like to take into account an arbitrary number of positional and directional constraints to be fulfilled by a loop ℓ. Let $\Phi = (\phi_0, \ldots, \phi_{n-1})$ with $\phi_i = (p_i, d_i) \in TM$ be a list of $n > 0$ constraint points that shall be passed by ℓ in order, with tangent parallel to d_i where $d_i \neq 0$ ($d_i = 0$ signifies a purely positional constraint). Let v_i be the g-node closest to p_i and ξ_i the set containing just the outgoing g-edge of v_i closest to parallel with d_i if $d_i \neq 0$, and the set of all outgoing g-edges of v_i otherwise.

For each two subsequent constraints ϕ_i, ϕ_{i+1} (all indices taken mod n), we compute intermediate elastica curves according to (2): for each pair $(a, b) \in \xi_i \times \xi_{i+1}$ (remember: a, b are edges in g and nodes in g') we compute the shortest path (taking the g'-edge weights w into account) from a to b in g' and record its cost as $\bar{w}(a, b)$. Note that if $|\xi_i| = 1$ or $|\xi_{i+1}| = 1$, the shortest paths for all pairs can be computed with just one run of Dijkstra's algorithm[2]. Only if both constraints have unspecified direction, $\min(|\xi_i|, |\xi_{i+1}|)$ runs are necessary. Being independent, all runs can conveniently be performed in parallel.

Then we form a metagraph with metanodes $\cup_{0 \leq i < n} \xi_i$, and directed metaedges $\cup_{0 \leq i < n} (\xi_i \times \xi_{i+1})$, weighted by $\bar{w}$ (cf. Figure 5). In this metagraph we find the minimum-weight cycle. Due to the special structure of the metagraph, we can do this more efficiently than by using the Floyd-Warshall algorithm. If at least one ξ is a singleton, the cycle is found using a run of Dijkstra's algorithm on the metagraph from this one element back to itself. Otherwise, this is done for each element of the smallest ξ and the minimum taken. In any

[2] A variant that does not return the empty path if $a = b$ must be used.

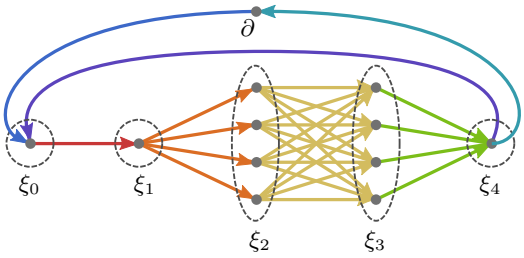


Figure 5: *Metagraph construction for multi-constraint elastica. Example with 2 positional (ξ_2, ξ_3) and 3 positional+directional (ξ_0, ξ_1, ξ_4) constraints. The boundary node ∂ allows for the construction of* dual curves *in addition to dual loops.*

case, for fixed n and k, the complexity of the complete elastica construction algorithm is $O(|V| \log |V|)$, V being the set of vertices of M's triangulation.

Concatenation of the intermediate elastica which correspond to the metaedges of the minimum-weight cycle yields the optimal loop interpolating Φ: in case that $d_i \neq 0$ at each constraint, the metagraph is a single cycle and the optimal loop trivially composed of all the pairwise elastica curves; otherwise, if there are constraints with unspecified direction, this algorithm finds the minimum over all (combinations of) possible directions. In the case of one positional constraint only, the $|\xi_0|$ runs of Dijkstra's algorithm on the metagraph yield $|\xi_0|$ loops (all crossing the anchor point in different directions), which can be offered as alternatives to the user.

It is worth noting that there are quad layouts which contain dual loops crossing themselves. Also such loops can readily be obtained using the described algorithm without any additional measures.

5.5 Principal Direction Alignment

The functional $q : TM \to \mathbb{R}$ is used to penalize deviation of loop directions from principal directions. We compute (unit) principal direction vectors $d_{\min}$, $d_{\max}$ and principal curvatures $\kappa_{\min}$, $\kappa_{\max}$ from the eigenvectors and eigenvalues of the shape operator [Cohen-Steiner and Morvan 2003]. The alignment of a unit tangent vector t with a principal direction can naturally be measured using the inner product $|\langle t, d\rangle|$ on M, the deviation using the reciprocal. We measure deviation to *either* principal direction using

$$\mathrm{dev}_p(t) := \max\left(|\langle t, d_{\min}\rangle_p|, |\langle t, d_{\max}\rangle_p|\right)^{-1} - 1,$$

which is zero in case of perfect alignment.

We weight this deviation with the local shape anisotropy factor $(|\kappa_{\min}| - |\kappa_{\max}|)^2$ as in [Knöppel et al. 2013] and obtain

$$q_p(t) := (|\kappa_{\min}| - |\kappa_{\max}|)^2 \mathrm{dev}_p(t),$$

which suitably vanishes in umbilic regions with ill-defined principal directions.

5.6 Feature Curves

It is usually desirable to align patch boundaries of a quad layout to sharp feature curves on the surface. To enable this, dual loops should not cross such features with small crossing angles, but ideally orthogonally. We replace $\mathrm{dev}(p, t)$ defined above by $\mathrm{dev}(p, t) := |\langle t, d_{\max}\rangle_p|^{-1} - 1$ on feature curves to achieve this. Now orthogonally crossing loops are favored, and the penalty increases to infinity as the crossing angle goes to zero.

5.7 Boundaries

So far we considered dual *loops*, i.e. closed curves. On surfaces with boundary ∂M, we also need to deal with non-closed curves which end at ∂M. For this, we add one additional node ∂ representing all boundaries to the above metagraph construction. Edges (a, ∂) and (∂, b) are added for each $a \in \xi_{n-1}$ and each $b \in \xi_0$, weighted by the cost of the shortest path between a/b and any g'-node in E_∂, where E_∂ is the set of all g-edges incident to a boundary vertex of M's triangulation $\mathcal{T}$. Where distinction is necessary, we call the resulting elastica *dual curves* instead of dual loops.

The user can choose whether the final quad layout should be aligned to the boundary, or whether non-aligned (trimmed) patches are acceptable. In the first case, dual curves should meet the boundary orthogonally. This is achieved by treating the boundary as a feature curve, using the weighting described in Section 5.6.

5.8 Symmetries

In case M has a global (exact or approximate, extrinsic or intrinsic) symmetry, it is likely that the user wishes to design an accordingly symmetric layout. Global symmetry can be of reflectional or rotational kind. In the first case, M consists of two, in the latter of two or more symmetric parts M_i. We provide the convenient option to perform the design on only one part, M_0, automatically transferring the layout to the other parts. We make no assumptions about the symmetry transformations ψ_i mapping M_0 onto M_i except for continuity; they can be specified by the user or be determined using (semi-)automatic methods [Mitra et al. 2013]. Note that the dual curves on M_0 must meet certain constraints to merge into continuous smooth loops on M. We achieve this as follows (cf. Figure 6):

Rotational In case of rotational symmetry, the boundary ∂M_0 has two symmetric parts which are identified by the map ψ_1. In g' we realize this identification by means of virtual edges, connecting respective g'-nodes adjacent to ∂M_0. A loop found in g' containing one or more of these virtual edges then represents one or more dual curves on M_0 whose ends lie on positions on ∂M_0 which are identified by ψ_1 in a pairwise manner (cf. Figure 6d). Hence, mapped to all parts via ψ, these curves join seamlessly (cf. Figure 6e).

Reflectional In case of reflectional symmetry we can make the following observation: a symmetric loop on M must either cross ∂M_0 orthogonally, or it must cross another (not necessarily distinct) loop on ∂M_0 – an analogous property was recently discussed for symmetric cross fields [Panozzo et al. 2012]. The orthogonal case can be handled just like an aligned boundary as described above. For the other case we must ensure the existence of both crossing curves at once. Conceptually, instead of letting a curve end

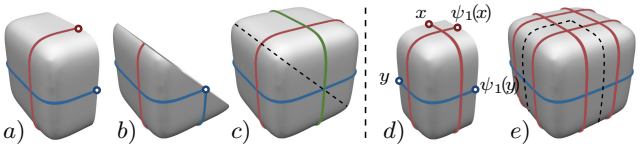
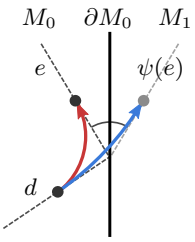


Figure 6: *Symmetry: (a) shows one representative half of the reflectionally symmetric object (c), dual curves end orthogonally on its boundary and thus form smooth loops when mapped to both halves. In (b) one half for another reflectional symmetry is shown, the blue curve is reflected on the boundary and thus forms two smooth crossing loops on (c). (d) shows one of four parts of a rotational symmetry, left and right halves of its boundary are identified, thus technically the two red curves are one loop, as is the blue curve. Mapped to all four parts, continuous loops are formed (e).*

at ∂M_0, we reflect it at this boundary and let it also form the second curve. Technically, weights of g'-edges between g-edges incident to ∂M_0 are computed differently: the red g'-edge (d, e) in the figure on the right is assigned the weight computed for the blue curve $(d, \psi(e))$ formed with the reflection of the g-edge e. In this way, curves resulting from the Dijkstra approach are either closed and contained in M_0, meet ∂M_0 orthogonally and end there, or meet ∂M_0 non-orthogonally and are reflected. Mapped to both halves of M via ψ, these curves form symmetric loops (or curves), smoothly crossing the symmetry's stationary line.



6 From Loop to Strip

While (infinitesimal) dual loops are sufficient to define the layout's topological structure, the lack of a geometric dimension may overly stress the user's imaginative powers. Hence, we appropriately expand dual loops, constructed as described in the previous section, to dual strips in order to better visualize the state and to guide the process of interactive design as described in Section 4.

Formally, in analogy to a dual loop (cf. Section 5.2), a dual strip is a continuous map $s(u, v) : [-l, r] \times [0, L] \rightarrow M$ of a rectangle (or a subset thereof in case of surface boundaries) onto the surface, with $s(\cdot, 0) = s(\cdot, L)$ in case of a closed dual strip. Its dual loop/curve (spine) is its zero-u-isocurve $s(0, \cdot)$. An alternative intuitive view of a dual strip is as a continuum of adjacent dual loops – which represent the one-parameter family of u-isocurves of s. The stripe pattern used to illustrate the strips (cf. Figure 1) simply represents a regular sampling of this continuum.

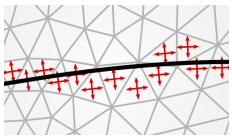
Conceptually, we build a dual strip by taking a dual loop and broadening it by extending the map up to appropriate bounds l and r, i.e. up to the point where the shape of M causes too large distortions in s. As the acceptable degree of distortion depends on the user's intent, the boundaries of the dual strips are to be seen as fuzzy indicators rather than definite patch borders. Because, a priori, the bounds l and r are unknown we compute an unbounded, global, distortion-minimizing map $S : M \rightarrow \mathbb{R}^2$ (cf. Section 6.1) from which all individual strip maps s can be extracted. This common map S is recomputed whenever the user inserts a new strip.

Note that one could follow the simpler strategy of expanding strips in a local, incremental manner – until some stopping criterion is met – e.g. based on a geodesic wavefront emanating from the loop [Schmidt 2013]. Besides the difficulty of defining a suitable local stopping criterion, this would result in u-isocurves with a constant spacing, implied by the constant gradient magnitude of a geodesic distance field. Depending on the model's shape this leads to isocurves with high geodesic curvature, thus to unnatural stripe patterns and strip boundaries (e.g. on the arc of model FIVE or the arms of model FERTILITY in Figure 7). The global map based approach, by contrast, yields stripes which much better resemble the edge flow of a globally coherent quad mesh by its very nature.

6.1 Global Parameterization

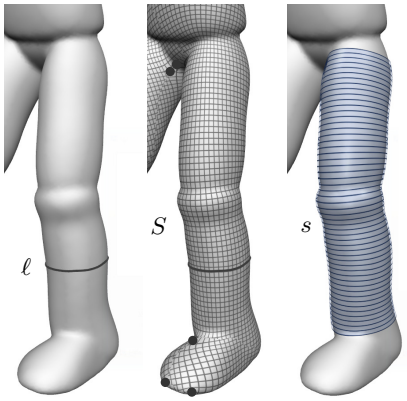
We require the global map S to be aligned with all dual loops, i.e. the loops should be isocurves of S. This implies that, in general, S needs to have singularities. Such parameterization with alignment and singularities can automatically be computed using an instance of cross field based parameterization [Kälberer et al. 2007].

For the generation of a suitable globally smooth cross field we make use of the representation vector concept [Palacios and Zhang 2007], which allows for smoothness optimization via simple sparse linear system solves [Knöppel et al. 2013; Diamanti et al.] – in contrast to the popular angle-based formulation [Bommes et al. 2009] without any integer constraints. In each triangle of the triangulation $\mathcal{T}$ of M which is crossed by a dual loop we set a unit representation vector which represents a cross aligned with the loop. These representation vectors are then harmonically interpolated over $\mathcal{T}$, a cross for each triangle recovered from the result, and the field singularities extracted, analogous to the recent description as a special case (4-RoSy) in [Diamanti et al.].



Then we compute a global, chart-based parameterization which is optimized for isometry and alignment of isocurves to the constructed cross field. This can be done efficiently, again using a single sparse linear system solve, in a least-squares manner as described in [Kälberer et al. 2007; Bommes et al. 2009] (note that in contrast to these works we do not have to perform any integer rounding). To achieve exact alignment to the dual loops we add (linear) alignment constraints to the optimization problem, forcing all edge intersections of a loop to lie on a common isocurve as described in [Campen and Kobbelt 2014]. Note that due to the isometry objective a globally constant parametric sampling is sufficient to generate stripe pattern strip visualizations of uniform density.

By construction, each dual loop is an isocurve[3] of S. To both sides of a loop ℓ we then conquer the continuum of adjacent isocurves, effectively extracting the strip map s as illustrated on the right. We stop when a singularity in S is reached (or, in the extreme case of a singularity-free field, the entire surface is conquered). The singularities quite naturally indicate appropriate bounds for the strips because they represent portions $(\pm k\,\pi/2)$ of total Gaussian curvature, in particular extrema like corners – which should not come to lie in the interior of quad patches for the benefit of patch developability (e.g. enabling low-distortion patch parameterization). Encountered degeneracies or fold-overs, which can sometimes appear in S, can gracefully be dealt with by simply stopping there, too. As these appear adjacent to singularities on principle, the impact is minuscule. Note that the field singularity constellation can also be exploited to easily perform the region coloring related to Gaussian curvature extrema (cf. Section 4.2).



Note that the cross field has no direct influence on the dual loop construction (in contrast to the DLM approach [Campen et al. 2012]); it merely serves the strip map creation for visualization purposes – and helps to ensure loop and strip transversality as described next.

6.2 Strip Compatibility

Whenever a strip has been created by the user, it should constrain the design space appropriately, as already mentioned in Section 4. Within the area covered by the strip we thus remove (i.e. disable)

[3] Formally, the chart-based parameterization S has transition functions across chart boundaries implied by the cross field. The term "isocurve" is implicitly meant to take these transitions into account.

all those nodes from the elastica graph whose associated direction forms an angle $\leq 45^\circ$ with the isocurves of the strip map s. Now, when the user hovers over the area covered by the strip, 1) no strips which are topologically equivalent are proposed, and 2) the underlying dual loops do only cross transversally, as required for validity.

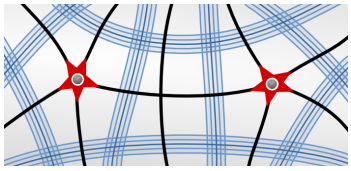
7 Primalization

The collection of dual strips defined by the user uniquely determines the structure of the primal quad layout. For the determination of the layout's detailed geometry, i.e. its actual embedding in M, we then have multiple (automatic, manual, assisted) options.

In the final stage of their method, [Tarini et al. 2011] apply a technique to optimize the embedding of a quad layout. It is driven by a quality metric based on parameterization of the layout's patches. A similar strategy is described by [Campen et al. 2012]. Another approach that pays particular attention to principal direction alignment is presented by [Campen and Kobbelt 2014]. These approaches proceed in a fully automatic manner.

Alternatively, in order to give control to the user also in this stage, a manual strategy can be followed: the user positions the layout's vertices in the regions inbetween the dual loops and draws layout edges dual to the loops in order to connect them. Note that no hard topological decisions need to be made in this context, the question is only where a vertex should be positioned within a region and where within a dual loop corridor a patch boundary should lie. Nevertheless, this can be a tedious task.

We provide assistance by automatically proposing good vertex positions and edge routes – which may serve as starting point for adjustments through the user. Each region (enclosed by dual loops) of valence $\neq 4$ contains at least one cross field singularity. We initially position a region's primal vertex onto this singularity if there is just one, otherwise into the geodesic center of the region. We then compute elastica curves within the corridors between the dual loops and use them as routes for the connecting edges. This is illustrated here schematically. The manipulation tools from Section 4.1 can be applied for these curves, too. The use of elastica is motivated by the fact that the objectives stated for dual loops in Section 5 often hold for primal edges, too. Note that in regions of valence 4 two curves cross, implying a natural position for the corresponding regular vertex. This latter approach was used for the examples shown in the following.



8 Results

Figure 7 shows layouts designed by non-expert users with the described system. The accompanying video gives further impressions of the design process.

Parameters We normalized all input models (to bounding box diagonal 1), such that model-independent parameters can be used. We used $\alpha = 1$ and $\beta = 0.25$ in (2) for all examples.

Timings We run our design system on a commodity PC. To get an impression of the system's performance, let's consider the practical scenario of an input mesh with 30,000 faces for example. The elastica loop construction is output sensitive: good dual loops with low cost are found within a few milliseconds; when multiple anchors induce very high cost, construction time can increase to around 1 or 2 seconds. The subsequent generation of the cross field takes around 200 ms, the parameterization and strip extraction around 400 ms. For optimal responsiveness, we display a loop immediately after its computation, while the field and strip generation is performed asynchronously in a background thread. In total, this allows to provide an interactive interface – only expensive loops restrain fluency to some extent. The user interaction time per model of Figure 7 for the design of all strips was between 10 seconds and 5 minutes.

8.1 Limitations & Future Work

In our system, optimal dual loops are constructed and then extended to dual strips. It could be of benefit if the strip geometry, in particular its width, could also already be considered in the loop optimization: wide strips could be favored over very narrow ones to the benefit of layout coarseness. However, it is unclear how this problem could be tackled algorithmically, and currently it seems unlikely that such integrated optimization could be performed at interactive rates, as desired in our system.

While we ensure that each loop crosses all other loops transversally, favoring orthogonality, a loop can be brought to cross *itself* with arbitrary angles; there is no mechanism in the loop optimization algorithm that could provide control over self-crossings and prevent small angles there. This is not a noticeable hurdle in practice because the principal direction alignment term in (1) favors orthogonality also in this situation, but depending on the anchors set by the user, self-crossing with bad angles cannot be ruled out. However, it is easy to automatically detect such a loop and highlight it accordingly, asking for adjustment.

Our current implementation offers a fully interactive workflow on models with several tens of thousands of triangles. In practice sometimes models with hundreds of thousands or millions of triangles are to be dealt with. The geometric fidelity of these complex models, however, is not relevant for the layout in many use cases – at least not for its topological structure. This opens the door for an efficient employment of simplified proxies. The layout design could be performed on a decimated model version and the result mapped back to the detailed original. This could even be done in a transparent manner, i.e. the elastica construction and strip expansion is performed on a coarse model in the background but the (interpolated) result displayed to the user on the original model. The final embedding optimization would be done on the detailed model again. We leave in-depth investigation of suitable simplification and mapping strategies (e.g. along the lines of [Lee et al. 1998]) and the analysis of the resulting interaction quality to future research.

While, as a sideline, this proxy strategy offers a way to abstract from small-scale geometric detail, it could also be valuable to investigate ways to offer scale control for the layout design process independent of the model resolution. For this the shape operator could be computed with a correspondingly large integration radius, the curvature of elastica be measured based on analogously smoothed tangent planes, and scale-aware methods for cross field and parameterization construction [Ray et al. 2009; Ebke et al. 2014] be employed for the strip expansion on the desired level of detail.

9 Conclusion

We proposed a novel, alternative concept for the design of quad layouts, i.e. partitionings of surfaces into nets of quadrilateral patches, as required in various fields of geometry representation and processing. In contrast to established systems it builds upon global operators based on dual loops and strips. The core technical component of our approach is a novel combinatorial algorithm for the constrained optimization of elastica loops embedded in surfaces. We further described how our system can support the designer in the presence of symmetries, boundaries, and feature curves.

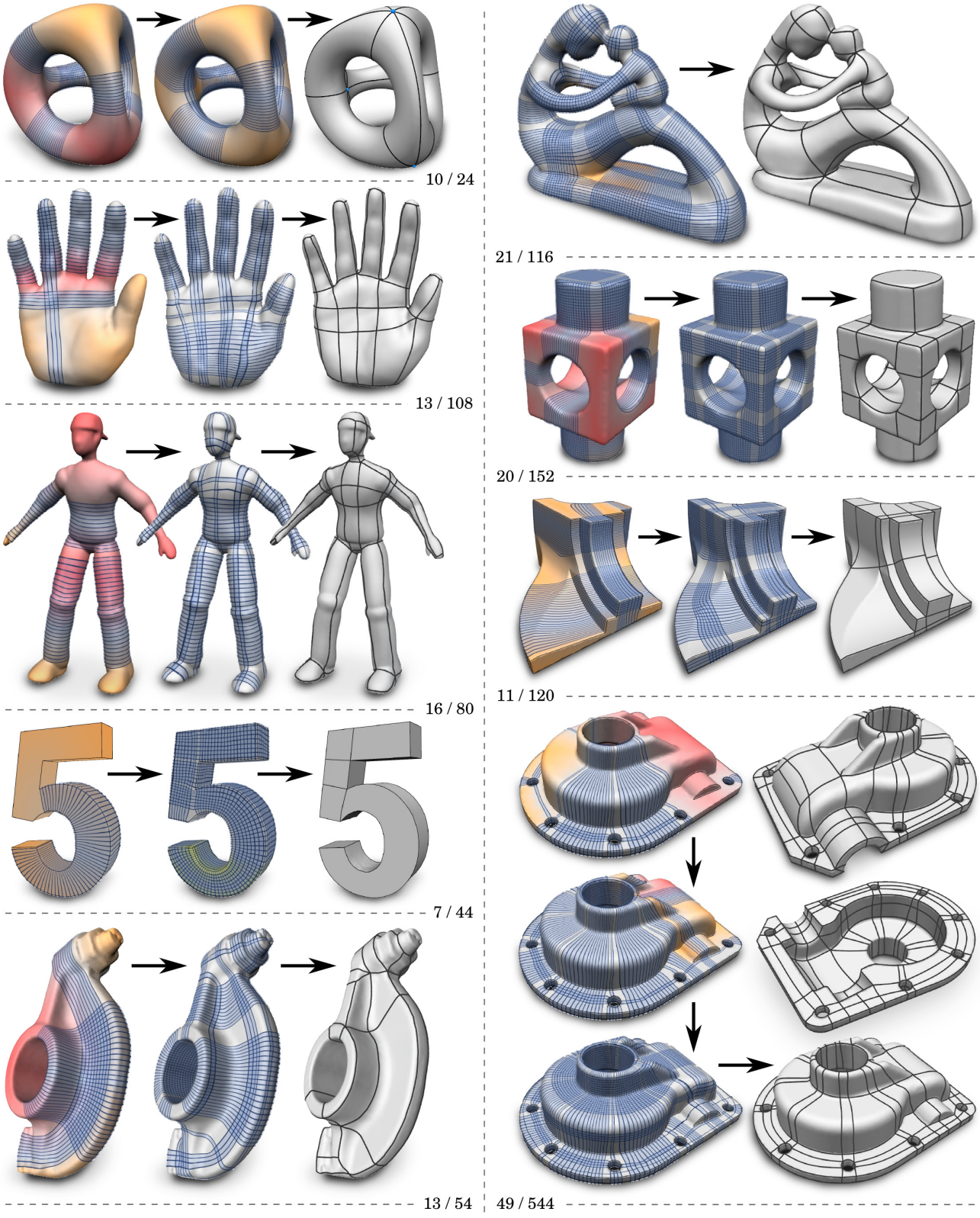


Figure 7: *Quad layouts designed with our method. Shown are intermediate design stages with dual strips and region coloring, followed by the final quad layout. Next to each layout the number of dual strips and the (typically much larger) number of primal layout edges is specified.*

Acknowledgements

This research project was funded by the European Research Council (ERC Advanced Grant "ACROSS", grant agreement 340884) and the German Research Foundation (DFG Cluster of Excellence UMIC, grant DFG EXC 89). Models have been taken from the AIM@SHAPE repository. The GUY model was initially created using Cosmic Blobs® by Dassault Systèmes Solidworks Corp.

References

BEINEKE, L. W. 1968. Derived graphs of digraphs. In *Beiträge zur Graphentheorie*. Teubner, Leipzig, 17–33.

BISCHOFF, S., WEYAND, T., AND KOBBELT, L. 2005. Snakes on triangle meshes. In *Bildverarbeitung für die Medizin*, 208–212.

BOIER-MARTIN, I. M., RUSHMEIER, H. E., AND JIN, J. 2004. Parameterization of triangle meshes over quadrilateral domains. In *Proc. SGP '04*, 197–208.

BOMMES, D., VOSSEMER, T., AND KOBBELT, L. 2008. Quadrangular parameterization for reverse engineering. *Mathematical Methods for Curves and Surfaces*, 55–69.

BOMMES, D., ZIMMER, H., AND KOBBELT, L. 2009. Mixed-integer quadrangulation. In *Proc. SIGGRAPH 2009*, 1–10.

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

BOMMES, D., LÉVY, B., PIETRONI, N., PUPPO, E., SILVA, C., TARINI, M., AND ZORIN, D. 2013. Quad-mesh generation and processing: A survey. *Computer Graphics Forum 32*, 6, 51–76.

BOMMES, D., CAMPEN, M., EBKE, H.-C., ALLIEZ, P., AND KOBBELT, L. 2013. Integer-grid maps for reliable quad meshing. In *Proc. SIGGRAPH 2013*, 98:1–98:12.

BRUCKSTEIN, A., NETRAVALI, A., AND RICHARDSON, T. 2001. Epi-convergence of discrete elastica. *Appl.Analysis 79*, 137–171.

CAMPEN, M., AND KOBBELT, L. 2014. Quad layout embedding via aligned parameterization. *Computer Graphics Forum*.

CAMPEN, M., BOMMES, D., AND KOBBELT, L. 2012. Dual loops meshing: quality quad layouts on manifolds. In *Proc. SIGGRAPH 2012*, 110:1–110:11.

COHEN-STEINER, D., AND MORVAN, J.-M. 2003. Restricted delaunay triangulations and normal cycle. In *Proc. Symp. Comp. Geom.*, SCG '03, 312–321.

DANIELS, J., SILVA, C. T., AND COHEN, E. 2009. Semi-regular quadrilateral-only remeshing from simplified base domains. *Comput. Graph. Forum 28*, 5, 1427–1435.

DIAMANTI, O., VAXMAN, A., PANOZZO, D., AND SORKINE-HORNUNG, O. Designing N-PolyVector fields with complex polynomials. *Computer Graphics Forum 33*, 5, 1–11.

DONG, S., BREMER, P.-T., GARLAND, M., PASCUCCI, V., AND HART, J. C. 2006. Spectral surface quadrangulation. In *Proc. SIGGRAPH 2006*, 1057–1066.

EBKE, H.-C., CAMPEN, M., BOMMES, D., AND KOBBELT, L. 2014. Level-of-detail quad meshing. In *Proc. SIGGRAPH Asia 2014*.

ECK, M., AND HOPPE, H. 1996. Automatic reconstruction of B-spline surfaces of arbitrary topological type. In *Proc. SIGGRAPH 96*, 325–334.

HOFER, M., AND POTTMANN, H. 2004. Energy-minimizing splines in manifolds. *ACM Trans. Graph. 23*, 3, 284–293.

JI, Z., LIU, L., AND WANG, Y. 2010. B-mesh: A modeling system for base meshes of 3d articulated shapes. In *Proc. Pacific Graphics '10*, 2169–2178.

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

KNÖPPEL, F., CRANE, K., PINKALL, U., AND SCHRÖDER, P. 2013. Globally optimal direction fields. *ACM Trans. Gra. 32*, 4.

KOVACS, D., MYLES, A., AND ZORIN, D. 2011. Anisotropic quadrangulation. *Comp. Aided Geom. Design 28*, 8, 449–462.

KRISHNAMURTHY, V., AND LEVOY, M. 1996. Fitting smooth surfaces to dense polygon meshes. In *Proc. SIGGRAPH 96*, 313–324.

LAI, Y.-K., JIN, M., XIE, X., HE, Y., PALACIOS, J., ZHANG, E., HU, S.-M., AND GU, X. 2010. Metric-driven rosy field design and remeshing. *IEEE Trans. Vis. Comput. Graph. 16*, 1, 95–108.

LEE, Y., AND LEE, S. 2002. Geometric snakes for triangular meshes. *Comput. Graph. Forum 21*, 3, 229–238.

LEE, A. W. F., SWELDENS, W., SCHRÖDER, P., COWSAR, L., AND DOBKIN, D. 1998. MAPS: Multiresolution adaptive parameterization of surfaces. In *Proc. SIGGRAPH '98*, 95–104.

LI, W.-C., RAY, N., AND LÉVY, B. 2006. Automatic and interactive mesh to t-spline conversion. In *Proc. SGP '06*, 191–200.

MITRA, N. J., PAULY, M., WAND, M., AND CEYLAN, D. 2013. Symmetry in 3d geometry: Extraction and applications. *Comput. Graph. Forum 32*, 6, 1–23.

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

MYLES, A., PIETRONI, N., KOVACS, D., AND ZORIN, D. 2010. Feature-aligned T-meshes. In *Proc. SIGGRAPH 2010*, 117:1–117:11.

PALACIOS, J., AND ZHANG, E. 2007. Rotational symmetry field design on surfaces. In *Proc. SIGGRAPH 2007*, 55:1–55:10.

PANOZZO, D., LIPMAN, Y., PUPPO, E., AND ZORIN, D. 2012. Fields on symmetric surfaces. In *Proc. SIGGRAPH 2012*, 111.

PANOZZO, D., BARAN, I., DIAMANTI, O., AND SORKINE-HORNUNG, O. 2013. Weighted averages on surfaces. *ACM Trans. Graph. 32*, 4, 60:1–60:12.

RAY, N., VALLET, B., ALONSO, L., AND LEVY, B. 2009. Geometry-aware direction field processing. *ACM Trans. Graph. 29*, 1, 1:1–1:11.

SCHMIDT, R. 2013. Stroke parameterization. *Comp. Graph. Forum 32*, 2, 255–263.

SCHOENEMANN, T., AND CREMERS, D. 2007. Introducing curvature into globally optimal image segmentation: Minimum ratio cycles on product graphs. In *Proc. ICCV 2007*, 1–6.

SCHOENEMANN, T., MASNOU, S., AND CREMERS, D. 2011. The elastic ratio: Introducing curvature into ratio-based image segmentation. *IEEE Trans. Img. Proc. 20*, 9, 2565–2581.

TAKAYAMA, K., PANOZZO, D., SORKINE-HORNUNG, A., AND SORKINE-HORNUNG, O. 2013. Sketch-based generation and editing of quad meshes. In *Proc. SIGGRAPH 2013*, 97:1–97:8.

TARINI, M., PUPPO, E., PANOZZO, D., PIETRONI, N., AND CIGNONI, P. 2011. Simple quad domains for field aligned mesh parametrization. *Proc. SIGGRAPH Asia 2011*, 142:1–142:12.

TONG, Y., ALLIEZ, P., COHEN-STEINER, D., AND DESBRUN, M. 2006. Designing quadrangulations with discrete harmonic forms. In *Proc. SGP '06*, 201–210.

ZHANG, M., HUANG, J., LIU, X., AND BAO, H. 2010. A wave-based anisotropic quadrangulation method. In *Proc. SIGGRAPH 2010*, 118:1–118:8.