Pacific Graphics 2016
E. Grinspun, B. Bickel, and Y. Dobashi
(Guest Editors)

Volume 35 (2016), Number 7

Tracing Field-Coherent Quad Layouts

Nico Pietroni[1†], Enrico Puppo[2], Giorgio Marcias[1], Roberto Scopigno[1] and Paolo Cignoni[1]

[1]ISTI, CNR, Italy [2]DIBRIS, University of Genova, Italy

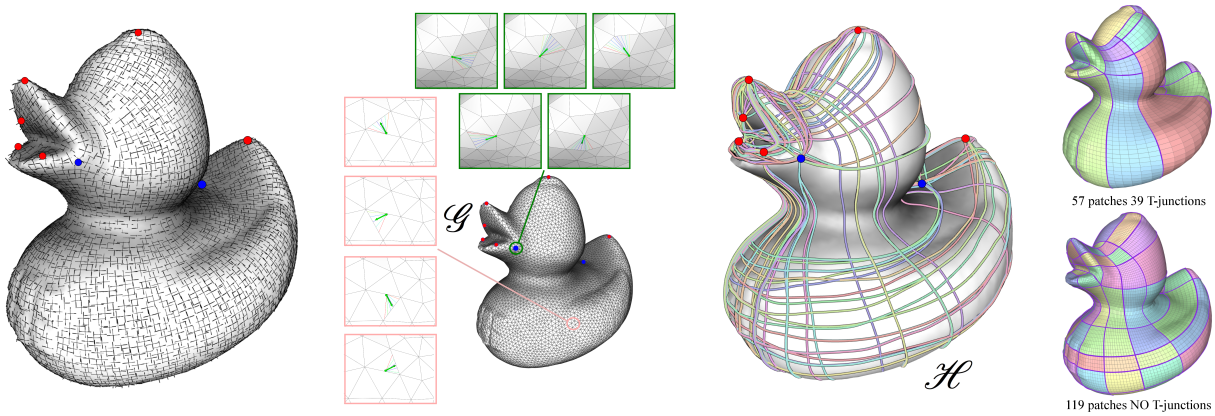


Figure 1: *Our input is a triangle mesh endowed with a cross field. We construct a graph $\mathscr{G}$ that allows us to trace field-coherent geodesic paths: graph $\mathscr{G}$ has multiple nodes at singularities of the field (upper zoom-in) and at auxiliary points along edges (lower zoom-in); each node is related to one of the cross field directions and a fan of field-coherent arcs emanate from it across a triangle. We use graph $\mathscr{G}$ to trace potential separatrices that connect singularities, forming another graph $\mathscr{H}$. A coarser or finer quad layout can be derived by solving a binary LP problem on $\mathscr{H}$. The solution can possibly involve t-junctions, while all final layouts may be used to produce a pure quad layout and quad meshes at arbitrary resolution.*

Abstract

Given a cross field over a triangulated surface we present a practical and robust method to compute a field aligned coarse quad layout over the surface. The method works directly on a triangle mesh without requiring any parametrization and it is based on a new technique for tracing field-coherent geodesic paths directly on a triangle mesh, and on a new relaxed formulation of a binary LP problem, which allows us to extract both conforming quad layouts and coarser layouts containing t-junctions. Our method is easy to implement, very robust, and, being directly based on the input cross field, it is able to generate better aligned layouts, even with complicated fields containing many singularities. We show results on a number of datasets and comparisons with state-of-the-art methods.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

Quad layouts provide an effective structure for parametrization and shape abstraction, furthermore they can be used for semi-regular quad mesh extraction or as a basis for subdivision surfaces [BLP*13]. In spite of several proposals in the literature, automatic methods are still far from providing high quality results on complicated shapes.

A quad layout is totally defined by the network of separatrices connecting the singularities of a directional field [TPP*11]; a piecewise linear version of such network is defined on a quad mesh by chains of edges that emanate from irregular vertices; the quad mesh itself can be seen as the result of subdividing each patch of the lay-

† e-mail: nico.pietroni@isti.cnr.it

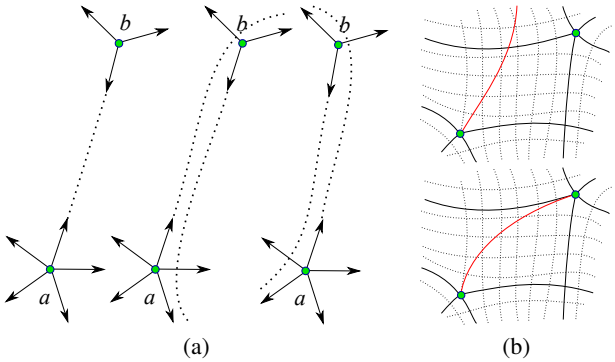


Figure 2: *(a) Instability of separatrices: a separatrix emanating from singularity A is connected to singularity B (left); an arbitrarily small drift may lead to totally different layouts (center and right). (b) Field coherency: a field-coherent path may* drift *from the cross field but it remains roughly parallel to it (top); a non-coherent path can* switch *to a different direction of the field (bottom). If non-coherent separatrices are selected, the resulting layout may contain non quadrilateral patches.*

out with a regular grid. Therefore, both the quad layout and the quad meshing problems can be solved by finding a proper network of separatrices.

In the quest for a solution to such problems, many difficulties arise from the intrinsic instability of directional fields on surfaces. Consider for instance the example depicted in Figure 2.a: a separatrix emanating from singularity A reaches another singularity B nearby; however, an arbitrarily small drift is sufficient for this separatrix to miss B and continue until reaching another singularity arbitrarily far from it. Since the arrangement of separatrices must be globally consistent, this apparently local fact has indeed global effects, eventually leading to a totally different network.

Further uncertainty comes from discretization. While singularities of a discrete field can be located robustly with a simple local analysis, tracing separatrices proves to be a much more challenging problem [MPZ14, RS14]. Moreover, an exact tracing of the underlying field does not necessarily provide a good solution: as demonstrated by [BLK11, TPP*11], just a few separatrices missing singularities in the short range and eventually wandering in long spirals may result in highly fragmented layouts.

A layout made of relatively few and large patches can be obtained by finding a consistent network made of as short as possible separatrices. Such separatrices must follow the underlying field, being allowed to drift from it just enough to hit singularities in the short range; moreover, they are allowed to intersect just orthogonally at corners of patches. These requirements restrict potential separatrices to the class of *field-coherent geodesic paths* that are exemplified in Figure 2.b and formally defined in Section 3.

An overview of the pipeline is shown in Figure 1. We take in input a triangle mesh, together with a discrete cross field computed with standard methods [HZ00, RVLL08, BZK09]. We designed a new robust, graph-based method for tracing geodesic paths directly on the triangle mesh, which are guaranteed to be field-coherent. Our method allows to test whether two intersecting paths are orthogonal or not, and whether two paths incident at the same singularity are compatible, on the sole basis of robust combinatorial tests. We compute a super-graph of potential separatrices $\mathcal{H}$, which is reduced to the optimal graph of separatrices by resolving a binary LP problem. With respect to methods based on global optimization of separatrices [RRP15] we do not need an initial globally smooth parametrization as input, and our formulation allows us to work also with an incomplete graph in input.

2. Related Work

In the following, we briefly review the contributions most related to our work. We will first overview the main methods for tracing aniostropic geodesics, then we describe and compare the methods for quad patch layout. A comprehensive survey on quad layout can be found in [BLP*13].

Tracing geodesic paths. Geodesic lines are shortest paths connecting pairs of points on a surface. Several exact and approximated methods have been proposed in the literature to compute geodesic lines on a mesh under the Euclidean metric [CWW13, KS98, LMS97, MMP87, SSK*05, Kan00]. Extensions of these methods to anisotropic metrics are investigated in [CHK13], where a new method is proposed based on a modified Dijkstra algorithm. The approach proposed in [ZZFJ14] uses an anisotropic geodesic to segment an input mesh into structural sound partitions. A similar approach to trace anisotropic paths proposed in [CBK12], which is specific for a metric induced by a cross field, also belongs to the class of Dijkstra methods. In order to consider a sufficiently large set of possible directions of propagation at each point, lines are traced by jumping from each vertex of the triangle mesh to the vertices of its k-ring. The distance induced by the anisotropic metric is calculated using a harmonic mapping of the k-ring of a vertex where field interpolated per edge is projected in parametric space. This approach is prone to numerical error due to the distortion introduced by the harmonic parametrization. We rely on a Dijkstra approach with auxiliary points along the edges (as in [LMS97]), which is modified to maintain geodesic paths always coherent to the underlying field.

Quad layout. Several methods has been proposed for field aligned quadrangulation based on a global parametrization [KNP07, BZK09, PTSZ11, BCE*13, CBK15], however most of them do not design coarse quad patch layouts. The idea of producing a coarse quad layout has been first investigated in [BMRJ04, CHCH06, DISC09]. The approach proposed in [MPKZ10] allows the insertion of t-junctions to adapt the patch layout to complex fields without deviating from the original field. This problem has been reconsidered in [BLK11, TPP*11] to align the quad layout to a given cross field. Both methods require that a field aligned quadrangulation is computed first, then they apply greedy techniques to disentangle its corresponding network of separatrices. A more sophisticated strategy that works directly on a triangle mesh equipped with a cross field has been proposed in [CBK12]. This method first extracts a set of field-aligned loops on the input surface, then it selects a minimal subset of loops that separate the cross-field singularities, by using a greedy strategy; the final quad layout is the dualization of

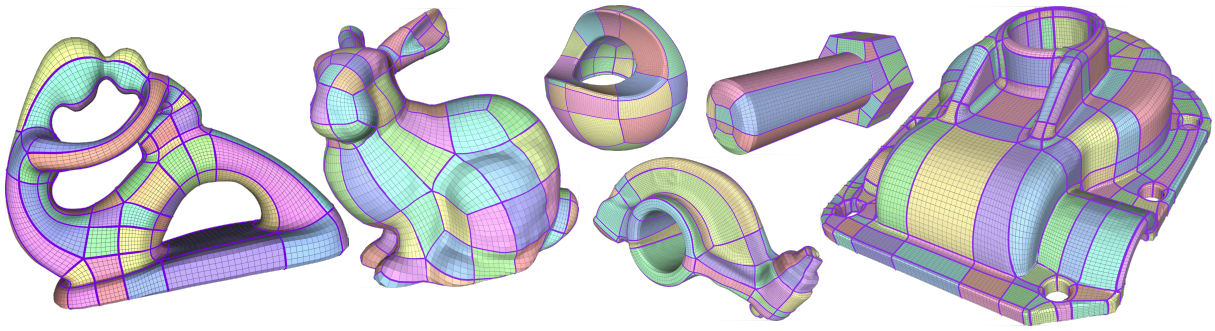


Figure 3: *Meshes with relatively simple fields that requires no t-junctions.*

this graph. Unfortunately, this method can get stuck on local minima, since the final layout is the result of a greedy process. Other methods [JLW10, CK14a, TPSHSH13, MTP*15] allow the user to interactively draw a quad patch layout using a simple graphical user interface.

The approaches proposed in [RRP15, ZZY16] are most related to our proposal because they share the idea of finding the best subset of separatrices that produces a quad layout by resolving a global 0-1 programming problem. However our method has several noticeable advantages:

Less dependence on input data: The approach of [RRP15] needs as input a globally smooth parametrization without foldovers. As shown in [MPZ14], neither [BCE*13] nor [Lip12] may guarantee the bijectivity of the produced parametrization, and this problem has been only partially solved in [DVPSH15].

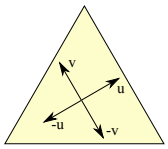
Simpler tracing: Designing a robust algorithm to trace paths in parametric space (as in [RRP15]) is a challenging task, especially in the presence of a complex parametric domain. Methods generating a globally smooth parametrization usually introduce discontinuities (jumps) and overlapping regions in parametric space, increasing the complexity of the tracing procedure. As opposite, we trace field-aligned separatrices directly on the triangle mesh by using a new graph-based strategy.

Better and more controllable result: The quad layouts of [RRP15, ZZY16] can be arbitrarily misaligned with respect to the original cross field. In fact, the tracing procedure inherits all the distortion of the input parametrization and may produce incontrollable artifacts in the final layout. This effect is shown in section 5.

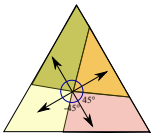
More flexible: Both methods in [RRP15, ZZY16] require to trace the complete set of candidate separatrices to guarantee a feasible solution. Instead, our system may produce a proper patch layout even starting from a subset of all possible separatrices. This feature is particularly useful in case of extremely complex fields.

3. Field-Coherent Geodesic Paths

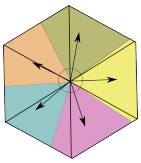
Let us consider a smooth manifold surface M. A cross field $\mathbf{X}$ is defined at each point of M by four pairwise orthogonal unit length vectors in the tangent plane, which we conventionally term u, v, $-u$ and $-v$ (see inset). Field $\mathbf{X}$ is said to be smooth if such directions vary smoothly while traversing M. Let ℓ be a simple open smooth line on M, which does not cross any singularity of $\mathbf{X}$, and let a and b be the endpoints of ℓ. Since $\mathbf{X}$ is smooth, if we select one of the four directions of $\mathbf{X}$ at a – say u – we can parallel transport u along ℓ up to b, mapping u to its corresponding direction of the cross field at each point along ℓ. We conventionally denote u_ℓ this parallel transport. Note that the parallel transport is not globally defined but dependent on ℓ: for instance, if we consider an orbit around a singularity and we place a and b along this orbit, we may get different transports of u at b, depending on which half of the orbit we follow.



Let p be a non-singular point of $\mathbf{X}$. We partition the tangent plane at p into four non-overlapping sectors, obtained by tracing bisectors between consecutive directions of the cross at p. Each sector is the portion of space that spans directions between $-\frac{\pi}{4}$ and $\frac{\pi}{4}$ from the respective direction of the cross.



If p is a singularity of $\mathbf{X}$, we generalize this partition by tracing bisectors between consecutive directions of separatrices emanating from p, so that each bisector is associated to a *port* corresponding to the outgoing tangent direction of a given separatrix at p, as shown in the inset. An arbitrary vector w in the tangent plane at p will belong to one sector and it will be associated to the corresponding direction.



Now let ℓ be a smooth open line as above, let t_a be the tangent of ℓ at its endpoint a, and let u be the direction of $\mathbf{X}(a)$ whose sector contains t_a. We say that ℓ is *coherent* to $\mathbf{X}$ if and only if the tangent t_p of ℓ at every point p lies in the sector of transported direction $u_\ell(p)$. This means that ℓ may *drift* from the integral lines of $\mathbf{X}$, but it does not *switch* to a different direction of $\mathbf{X}$ (see Figure 2.b).

We introduce a non-Riemannian metric on M to penalize non-coherent lines by increasing their length proportionally to their amount of drift. Let p be a point of M, let w be a vector in its tangent plane and let u be the direction of $\mathbf{X}(p)$ that contains w in its sector. We define norm

$$\|w\|_{\mathbf{X}} = |w|(1+\alpha\frac{\text{Angle}(u,w)}{\pi/4}) \qquad (1)$$

where $|w|$ is the Euclidean norm of w, Angle measures the unsigned

angle between a pair of vectors, and α is a parameter that tunes the amount of penalty for the drift.

In the following, we are interested in finding field-coherent paths of minimal length w.r.t. $\|\cdot\|_{\mathbf{X}}$ (i.e., anisotropic geodesics) that connect pairs of singularities of $\mathbf{X}$. Note that, since we restrict our search to field-coherent lines only, we are in fact looking for potential separatrices of a slightly perturbed cross field that retains the same singularities of $\mathbf{X}$.

3.1. Discrete setting

We represent a discrete cross field $\mathbf{X}$ on a triangle mesh T by storing one cross per triangle. The parallel transport of crosses between adjacent triangles is obtained by flattening the triangles to the same plane and associating each direction of a cross with the direction of the other cross that makes the smallest angle with it. This transport from triangle to triangle is defined by a simple 2D signed rotation; a vertex p is a singularity of $\mathbf{X}$ if and only if the sum of all rotations obtained by orbiting about p is different from zero.

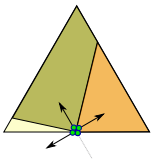
Smooth lines over M will be approximated with polylines over T, having their joints only at intersections with edges of the mesh. For the sake of simplicity, we will consider only lines that do not cross vertices of T, while they may have their endpoints at singular vertices. Given such a polyline ℓ, the transport u_ℓ of a field direction along it can be defined as in the continuous case, by using the triangle-by-triangle parallel transport defined above. Similarly, we can define a polyline ℓ to be coherent with $\mathbf{X}$ if the direction of segments of ℓ remains within the same sector of the transport u_ℓ at all triangles traversed by ℓ.

3.2. The geodesic graph

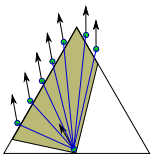
In order to efficiently explore the space of field-coherent paths that connect singularities we build a graph $\mathcal{H}$ that represents discrete field-coherent geodesic paths traversing mesh T. The nodes of $\mathcal{H}$ are the vertices of T that correspond to singularities of $\mathbf{X}$; the arcs of $\mathcal{H}$ are field-coherent polylines that approximate geodesic paths between singular vertices. Such polylines have their joints at auxiliary points placed along the edges of mesh T. Auxiliary points provide enough degrees of freedom to propagate geodesics over the mesh independently of the underlying tessellation, allowing us to diffuse the sectors of the cross field coherently through adjacent triangles, as shown in Figure 4.

The arcs of graph $\mathcal{H}$ are in fact built as chains of directed straight-line arcs of an intermediate graph $\mathcal{G}$, which is built as described in the following. We start by placing auxiliary points along the edges of T, controlling their density with an angular parameter ϕ: for each edge e, we tune its sampling step so that the angle subtended by two consecutive points on e and the opposite vertex is smaller than the given threshold ϕ. In practice, angle ϕ sets the sampling rate of space in terms of angular directions.

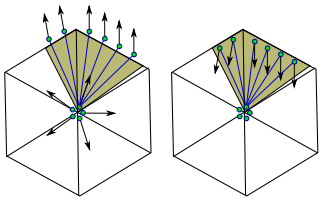
Intermediate graph $\mathcal{G}$ has its nodes placed at singular vertices and at auxiliary points; while its arcs are straight-line segments crossing each triangle and joining either pairs of auxiliary points, or singular vertices with auxiliary points. Since we want to represent only field-coherent paths, each auxiliary point s lying on edge e generates eight potential nodes of $\mathcal{G}$, i.e., one node per triangle incident at e and per direction of the cross in the triangle.



The nodes corresponding to a single auxiliary point and a single triangle are depicted in the inset by bullets, with arrows representing their associated directions in the cross. Each node has a corresponding sector inside the triangle, which represents the portion of triangle that can be traversed by field-coherent paths outgoing from the node in its associated direction. Note that there are at most three nodes per point per triangle, as at least one sector is always completely occluded by the edge. Therefore, each auxiliary point will actually correspond to six nodes of $\mathcal{G}$.



Each node is connected to directionally coherent nodes on the adjacent faces, which belong to its sector. For instance, let us consider a triangle t and a node s_t^u placed at point s lying on an edge of t and pointing at direction u. Then, node s_t^u is connected with an outgoing arc to all nodes that fall within the sector of u, belong to neighboring triangles, and bear a coherent direction (i.e., the transport of u to each neighboring triangle). See the inset for an example.



Singular vertices of $\mathbf{X}$ generate special nodes. Since the cross field is undefined at a singular vertex p, we generate twice as many nodes as the number of separatrices incident at p (typically 3 or 5); we orient each such node according to the tangent direction of its corresponding separatrix at p and we assign it a sector consequently. Details on how we derive the directions of separatrices at singularities can be found in the Additional material. In order to find the correspondence between sectors and adjacent nodes, we conformally map the star of p to 2D space. For each direction there is one *source* node for outgoing arcs and one *sink* node for incoming arcs. Outgoing arcs are generated in the same way as arcs from auxiliary points. Arcs incoming at sinks are generated similarly, by connecting to each sink n_v at vertex v the nodes that belong to the sector of n_v, to a triangle incident at v and that contain v in their sector (see inset).

By construction, each arc of $\mathcal{G}$ is contained in a triangle, it is directed, and it is tagged with one of the four directions of $\mathbf{X}$ in the triangle. Moreover each arc of $\mathcal{G}$ connects two nodes whose directions are coherent by parallel transport, hence, any directed chain of arcs of $\mathcal{G}$ is a field-coherent path; in fact, $\mathcal{G}$ provides a discrete approximation of all such paths on T. Finally, the length/weight of each arc of $\mathcal{G}$ is computed according to Equation 1.

The number of arcs of $\mathcal{G}$ in each triangle t depends on the sampling density of auxiliary points, which varies with parameter ϕ; this number is quadratic in the number of auxiliary points on t; in practice, it is easily bounded by a small constant, hence the overall complexity of $\mathcal{G}$ is linear in the size of T.

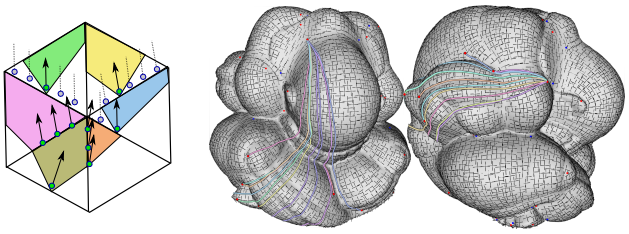


Figure 4: *Left: The Dijkstra search proceeds by spanning sectors of the cross fields across the mesh. Right: the set of Field-Coherent Geodesic Paths diffusing from two singularities along given directions on the Pensatore model.*

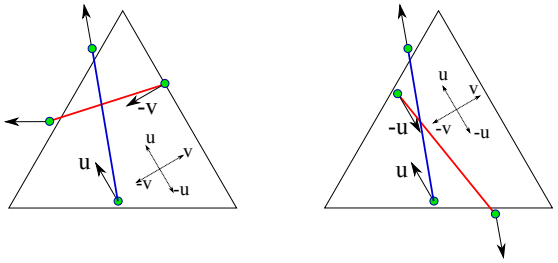


Figure 5: *Orthogonal (left) and tangential (right) intersections can be seen simply looking at the cross field direction associated to arcs.*

3.3. Tracing field-coherent geodesic paths

In the proposed setup, a field-coherent geodesic path between two nodes is obtained with a simple Dijkstra shortest path computation on graph $\mathcal{G}$. A single search launched from a source node n_p spreads field-coherent geodesic paths over the surface, approximating the continuous settings (see Figure 4). This is sufficient to find all field-coherent geodesic paths from n_p to all sink nodes of $\mathcal{G}$. Running Dijkstra searches from all source nodes of $\mathcal{G}$ until reaching all sinks provides us with a complete graph of potential separatrices. For meshes with border, path expansion stops as soon as the border is reached, and such paths are discarded. We encode the results of these Dijkstra searches into an undirected graph $\mathcal{H}$, by identifying sources with their corresponding sinks as well as opposite paths; thus, each node of $\mathcal{H}$ is a source of $\mathcal{G}$ and each arc of $\mathcal{H}$ corresponds to a source-to-source field-coherent geodesic path. Since running a complete Dijkstra search may be computationally expensive if the number of singularities is high, we allow the search to stop after a given number k of sinks has been reached. Note that in this case graph $\mathcal{H}$ may be not complete.

Our approach offers another advantage: for each pair of arcs of $\mathcal{H}$, which corresponds to two potential separatrices crossing in a common face, we can precisely identify whether they intersect orthogonally or tangentially, by simply looking at their spanned sectors (see Figure 5). Intersections are computed while generating the arcs of $\mathcal{H}$ and stored for subsequent use. Moreover, since graph $\mathcal{H}$ encodes each singularity s of the cross field with as many nodes as the valence of s, by construction $\mathcal{H}$ already encodes whether two different separatrices incident at s are compatible (i.e., they reach it from different directions of the cross field) or not. This means that all information needed to test the compatibility of separatrices during the following optimization step are already encoded in $\mathcal{H}$ in combinatorial (hence robust) form.

4. Quad Patch Layout

We aim at finding an optimal set of separatrices that subdivide the mesh into quadrilateral patches. As remarked in [CBK12, RRP15], this can be achieved only if the following constraints are satisfied:
Completeness: each port of each singularity must be incident at exactly one separatrix.
Orthogonality: separatrices composing the layout can intersect only orthogonally.

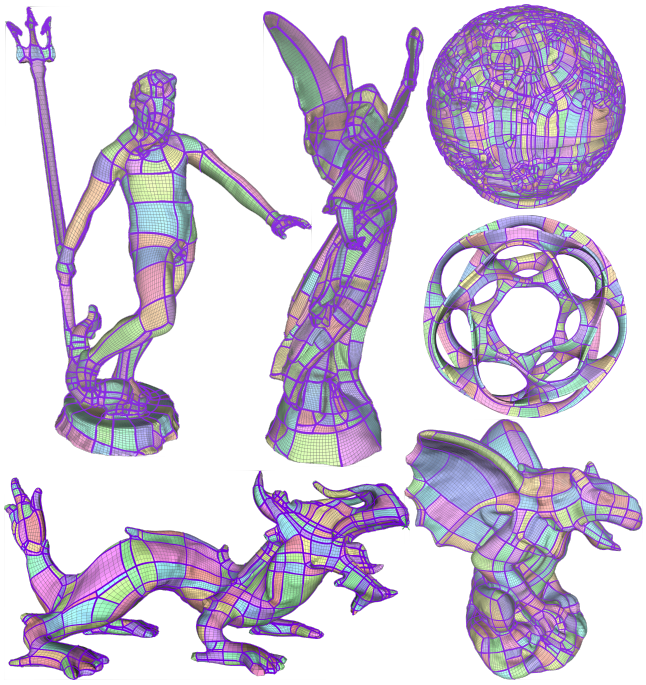


Figure 6: *Complex meshes requiring a few t-junctions.*

As explained above, our setup allows us to test these conditions in a very efficient and robust manner that does not involve geometric computations. From Theorem 4.1 in [CBK12], it follows that these two constraints guarantee the arrangement of separatrices to form a quad layout, provided that all separatrices are field-coherent paths. On the other hand, it is quite easy to show that non-quadrilateral patches may appear in the layout if field-coherency is broken. See the right side of Figure 2 for an example.

A feasible solution is any subgraph of $\mathcal{H}$ that fulfils completeness and orthogonality. Within the space of all feasible solutions we want to choose the one that minimizes the total length of the paths in the anisotropic metric.

4.1. A relaxed setup

If $\mathcal{H}$ is complete, an optimal solution can be found by resolving a 0-1 programming problem, similarly to [RRP15]. However, we already remarked that finding a complete graph for an arbitrarily complex cross field may be prohibitively time consuming. Moreover, a conforming quad layout may become very fragmented for complicated fields containing many singularities; in these cases, it is sometimes more convenient to find a simpler quad layout with a few t-junctions. We thus relax the problem by substituting the *Completeness* constraint with the following weaker constraint:

Consistency: each port must have *at most* one incident separatrix.

We aim at keeping as many separatrices as possible subject to the *Orthogonality* and *Consistency* constraints and, at the same time, while minimizing the total length of the selected separatrices. Note that we can obtain a solution in which some nodes are not reached by any separatrix, i.e., they remain *dangling*. We will show later on how a layout with t-junctions can be obtained by partially expanding separatrices from dangling nodes.

We define a boolean variable c_i for each arc s_i of $\mathcal{H}$. Each candidate separatrix s_i joins two nodes. In order to fulfil *Consistency*, we impose a constraint at each source node n_j:

$$0 \leq \sum_{s_i \in Sep(n_j)} c_i \leq 1 \tag{2}$$

where $Sep(n_j)$ is the set of arcs of $\mathcal{H}$ incident at node n_j. Moreover, in order to fulfil *Orthogonality*, for each pair of arcs s_i, s_j that intersect tangentially we impose that they are not both selected, by constraint:

$$0 \leq c_i + c_j \leq 1 \tag{3}$$

Hence, we obtain a system of linear constraints with 0-1 variables, containing as many equations as the nodes plus the pairs of arcs that intersect tangentially in $\mathcal{H}$.

In order to maximize the number of separatrices while minimizing their length at the same time, we maximize the following objective function:

$$\sum_i \frac{c_i}{\gamma(s_i)} \tag{4}$$

subject to the above system of constraints, where $\gamma(s_i)$ is the length of the candidate separatrix, computed using the anisotropic metric of equation 1. To solve this ILP problem we use [Ach09]. Note that the objective function is significantly different from the one of [RRP15]. Due to the relaxed constraints, minimizing directly the total length would inevitably lead to an empty solution. On the other hand, if the graph $\mathcal{H}$ contains enough arcs (which does not necessarily require it to be complete) a complete solution is found also in our case.

4.2. The solving cycle

The approach described above can be iterated to reduce the number of dangling nodes and, consequently, of t-junctions. Once we gather a valid layout, we iterate a Dijkstra search from the dangling nodes, while freezing the current solution as a set of barriers. This means that the search is stopped as soon as the propagating path intersects tangentially a separatrix which belongs to the current solution (see Figure 7).

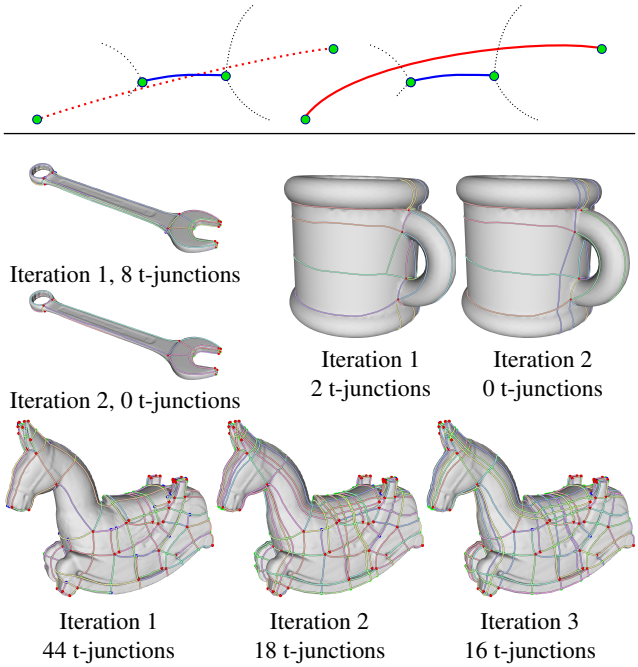


Figure 7: *Top: The dotted red separatrix on the left would be shorter than the continuous red separatrix on the right. However, adding the blue separatrix to the layout in a first run prevents the insertion of the dotted red separatrix, while the continuous red separatrix is found as best shot, compatible with the blue constraint. Bottom: a few iterations allow us to remove or considerably reduce the number of t-junctions in the final layout.*

Then, we solve the same LP problem bound to dangling nodes and this new set of candidate separatrices. Note that the new set of separatrices does not interfere with the previous solution, because they are incident at different nodes, then the two sets of paths are guaranteed to mutually intersect only orthogonally. We iterate this process until no new separatrix can be added to the final layout. Usually this process converges in 3-4 iterations. It is worth noticing that a solution found with this lazy approach does not necessarily coincide with the solution found starting at a complete graph $\mathcal{H}$, but the speedup may be dramatic for a complicated input.

Few cycles of this iterated approach are sufficient to eliminate all dangling nodes, thus providing a conforming quad layout, for a simple enough input, like in most examples processed in the previous literature. However, for rather complicated examples and with all bordered meshes, the process may terminate while leaving some dangling nodes. In this case, we perform a further propagation of a separatrix from each dangling node. The propagation is stopped at its first orthogonal intersection with an existing separatrix, thus adding a t-junction to the final layout. This propagation step is constrained by the graph of separatrices that has been computed in the previous steps. If a cylinder partition is present, we trace a field

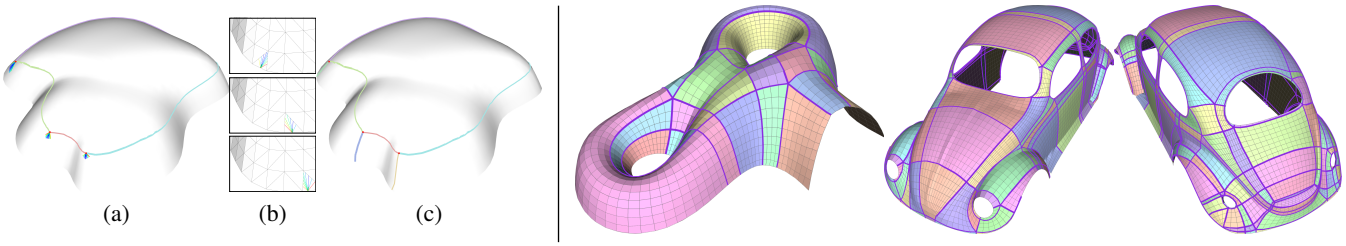


Figure 8: *Left: some nodes may be unsolved because they need to be connected with the border (a); it is sufficient to propagate paths from the singularity until they reach the border (b) to complete the patch layout (c). Rigth: some examples of models with multiple complex borders.*

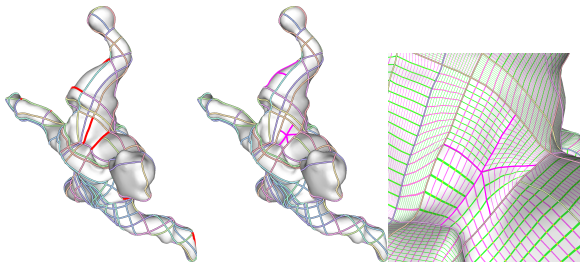


Figure 9: *An example of zero chain removal: The faces that do not satisfy equality constraints are highlighted (a); and then subdivided (b); leading to a globally smooth parametrisation (c).*

line starting from a node which is perpendicular to the boundary reducing the partition to a quad face, as in [MPZ14].

4.3. Handling Borders

In the case the input mesh is not watertight, some singularity may have some associated dangling nodes (see figure 8.a). In this case we simply use the same process we used to diffuse the t-junctions. We propagate the separatrix from each dangling node until it reaches a border node (see figure 8.b). This process produces a proper patch subdivision (see figure 8.c). Some example of models with multiple borders is shown in figure 8.

4.4. Partition consistency

Every quad in the patch layout can be mapped to a rectangle in parametric space with sides of integer length. A seamless globally smooth parametrization can be obtained only if the opposite sides of each patch map to the same length in parametric space, as well as coincident sides of adjacent patches map to the same length in parametric space. These constraints give a system of linear equations with integer variables.The objective function to be minimized accounts for the difference between an ideal length of sides of rectangles in parametric space. The resulting ILP problem is solved easily with the same solver [Ach09] that we used to find separatrices.

In case of a purely quadrilateral patch layout these constraints can be always satisfied. However in a more generic scenario that allows t-junctions this system of equations may lead to a zero chains [MPZ14], i.e an inconstant set of constraints that are satisfied only when some variable becomes zero. In order to detect and isolate such inconsistencies we set size equivalence constraints only between sides with no t-junctions. As opposite, we set equivalence constraints in a least squares sense when t-junctions are involved. As a consequence, when a zero chain occurs, this system of equations produces a length assignment that does not satisfy the opposite side equivalence. In such case, we perform one step of Catmull-Clark subdivision on each face with side inconsistency. This operation will erase the t-junction that causes the inconsistency by introducing one irregular vertex. We proceed resolving again the system providing a new edge side assignment.

Notice that the Catmull-Clark subdivision of a single patch may create additional t-junction(s) leading to another possible source of inconsistence. We force opposite side equivalence constraint across this newly created t-junctions. We repeat this sequence of operations until all opposite sides are consistent. In the worst case, every face containing a t-junction may be subdivided, leading to a mesh with neither t-junctions, nor inconsistencies. An example of this procedure is shown in figure 9.

4.5. Quadrangulation

The final quadrangulation can be optioned by sampling each patch uniformly in parametric space. As patches have integer sizes that are globally consistent along shared edges, this procedure produces a proper quadrangulation. The shape of faces and their alignment to shape features may be improved with a final smoothing operation. We obtained good results by applying the simple technique proposed in [PTP*15], which interleaves steps of polygonal regularization with closest point re-projection on the original surface. We believe the quadrilateral patch layout could be further optimized by using the technique proposed by [CK14b].

5. Results

Our method relies on two main parameters: the drift coefficient α for the anisotropic metric (see Equation 1); and the number of sinks k reached from each source during Dijkstra propagation for building graph $\mathcal{H}$. Coefficient α is used to tune the alignment of separatrices to the input field: the higher α, the more aligned the layout, the less freedom we have in connecting singularities.

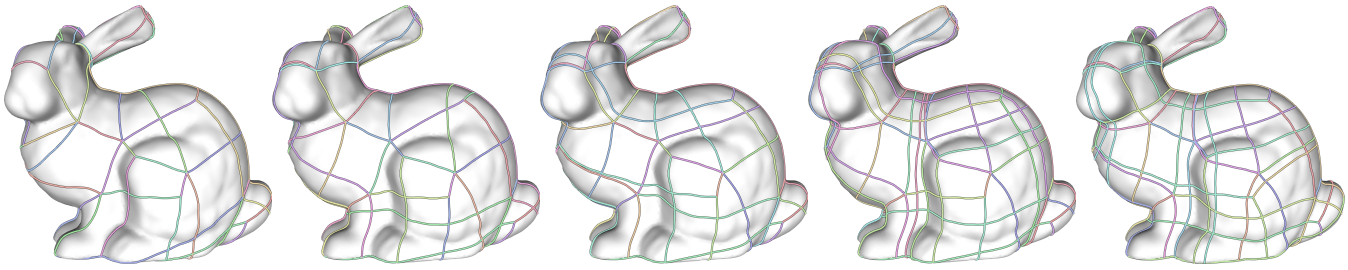


Figure 10: *Increasing the value of α results in a batter alignment to the input field at the cost of a larger number of patches in the final layout. No t-junction has been introduced.*

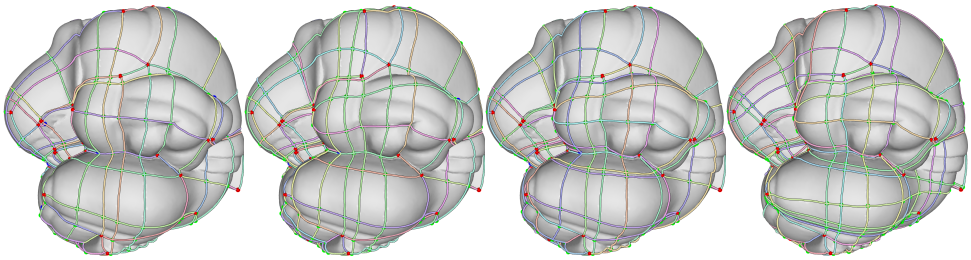


Figure 11: *Increasing the value of k reduces the number of t-junctions in the final layout, but slows down the entire process.*

The number of patches in the layout may increase for higher values of α (see Figure 10). We experimentally found that values between 10 and 30 give a good trade-off between alignment and complexity of the final layout. Parameter k is used to trade-off between completeness of graph $\mathcal{H}$ and speed: the higher k, the more complete $\mathcal{H}$, the slower its computation. A low value of k may result in more iterations of the solving cycle and possibly in a sub-optimal layout with more t-junctions; on the other hand, depending on the complexity of the dataset, a high value of k may slow down computation dramatically (see Figure 11).

Parameter ϕ controls the creation and distribution of auxiliary points in the intermediate graph $\mathcal{G}$ used for the computation of graph $\mathcal{H}$. We set its value equal to 10° for all models. Smaller values of ϕ usually just produce slightly smoother paths, without significantly changing the final layout (see Figure 12).

Finally, note how the complexity of the cross field in input may affect the final result: a field containing many singularities may result in better alignment to geometric features, but it can increase considerably the number of patches in the final layout (see Figure 13).

Figure 3 shows a sequence of quad patch layouts for relatively simple cross fields that required no t-junctions. The same datasets have been used several times in the previous literature and our results are at least qualitatively comparable with the state of the art. Figure 6 shows results on complicated input fields that have seldom been analyzed by quad layout methods in the previous literature: these results require the use of a few t-junctions.

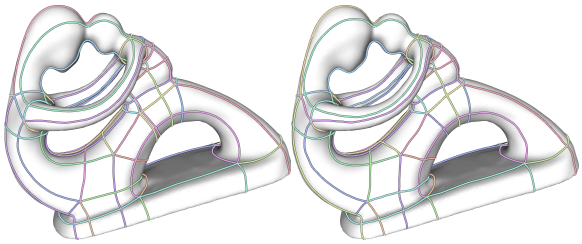
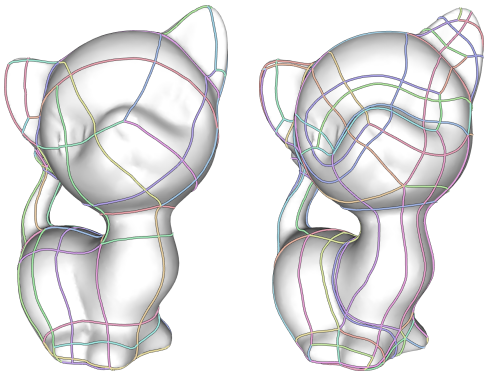


Figure 12: *Changing the value of ϕ, that controls the auxiliary points in the intermediate graph $\mathcal{G}$, usually does not affect the final layout.*

Results are summarized in Table 1; values of α and k used for each dataset are shown in the table.

5.1. Comparisons

Figure 14 shows a comparison with the integer-grid (IG) technique [BCE*13]. Note that IG is capable of reducing significantly the number of patches composing the final layout, however this comes at the price of a large deviation from the input field, thus introducing a high distortion in the final layout. This effect is highlighted with circles in Figure 14. Conversely, our technique supports direct control of alignment to the field, which is implicitly bounded by field coherency, even for low values of α. Field distortion can be reduced in [BCE*13] by increasing the size of the integer grid.



32 singularities
101 patches

46 singularities
189 patches

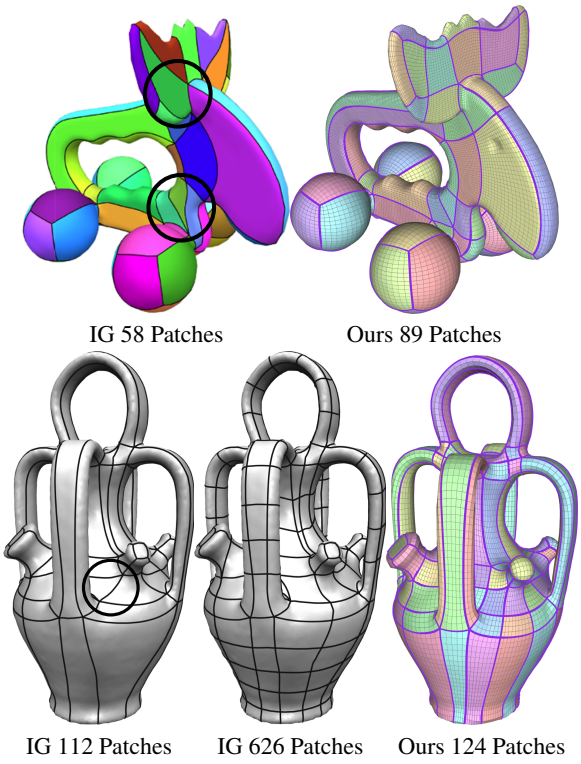
Figure 13: *A more complex field in input results in a more complex patch layout.*

mesh	△	S	α	k	$\|\mathcal{G}\|$	□	⊣	Time
bunny	30K	50	15	30	528K	179	0	26.3
bolt	58K	24	30	20	243K	52	0	7.5
casting	37K	126	60	30	107K	515	0	76,6
chinese	52K	290	30	10	920K	941	38	73.6
dragon	25K	343	15	10	466K	1024	90	64.0
fertility	28K	38	30	30	492K	128	0	22.0
gargoyle	20K	141	30	10	462K	528	44	18.2
heptoroid	20K	168	30	10	979K	825	0	63.0
lucy	50K	173	30	10	2039K	671	62	101.0
neptune	52K	226	5	5	925K	640	62	32.9
pensatore	100K	72	20	30	3287K	336	0	193.0
redbox	50K	1170	30	5	2148K	4586	450	209.0
rockerarm	70K	30	30	30	1239K	75	0	43.3
sculpt	73K	16	30	10	130K	78	0	2.9

Table 1: *Statistics of the processed data: name of the input mesh, number of faces, number of singularities S, field alignment factor α, number of sinks reached from each source k, number of nodes composing $\mathcal{G}$, number of quadrilateral patches, number of T-Junctions and processing times in seconds.*

Since the method is based on a global parametrization, the resolution is increased globally in every region of the input surface, thus obtaining a layout made of many more patches. Conversely, our method can adapt the tessellation locally without the need of a global scaling factor.

Figure 15 shows a comparison with the Perfect Matching Quad Layouts (PMQ) approach [RRP15]. For each face, we show the deviation between the original cross field and the direction field in the resulting parametrization (considering invariance to $k\frac{\pi}{2}$ rotations). Notice that the distortion induced by the Poisson parametrization required by [RRP15] can affect the final patch layout in an unexpected manner, increasing the misalignment or creating artifacts. This undesirable effect may increase if the underlying parametrization has foldovers. For these reasons [RRP15] can include separatrices in the quad layout, which are not well aligned with the original cross field. As opposite, tracing separatrices that directly



IG 58 Patches
Ours 89 Patches

IG 112 Patches
IG 626 Patches
Ours 124 Patches

Figure 14: *Comparison with Integer Grid method [BCE*13]. No t-junction in all layouts.*

follow the original cross field leads to a better conforming patch layout with out method.

Figure 16 shows a comparison with the approach of [MPZ14]. While [MPZ14] is able to trace the field more precisely (see Table 2), we obtain a significantly simpler patch layout because we introduce few t-junctions, as seen in the Chinese dragon model. Since our method minimizes the total number of required t-junctions, zero loops configurations happen more rarely than in [MPZ14]. Figure 17 shows a comparison where the approach proposed by [MPZ14] generates a zero loop configuration that is not present in our patch layout. This is due to the fact that our approach is able to align singularities, thus reducing the total number of t-junctions and implicitly collapsing several possible zero cycles. Finally, differently from [MPZ14], our method has already integer length assigned to each edge and it can be used to produce a quadrangulation.

Figure 18 shows a comparison with the quadrangulation produced by [BZK09]. Quads are colored according to distortion with respect to a rigidly aligned ideal square, as described in [PTP*15]. Note that we achieve a fully structured quad patch layout at the cost of a slightly larger distortion: the greenish zone in our model is due to horizontal stretching that makes slightly rectangular quads. On the other hand, [BZK09] does not capture the global quad layout, thus containing more skewed elements, e.g., inside the hole on the right.

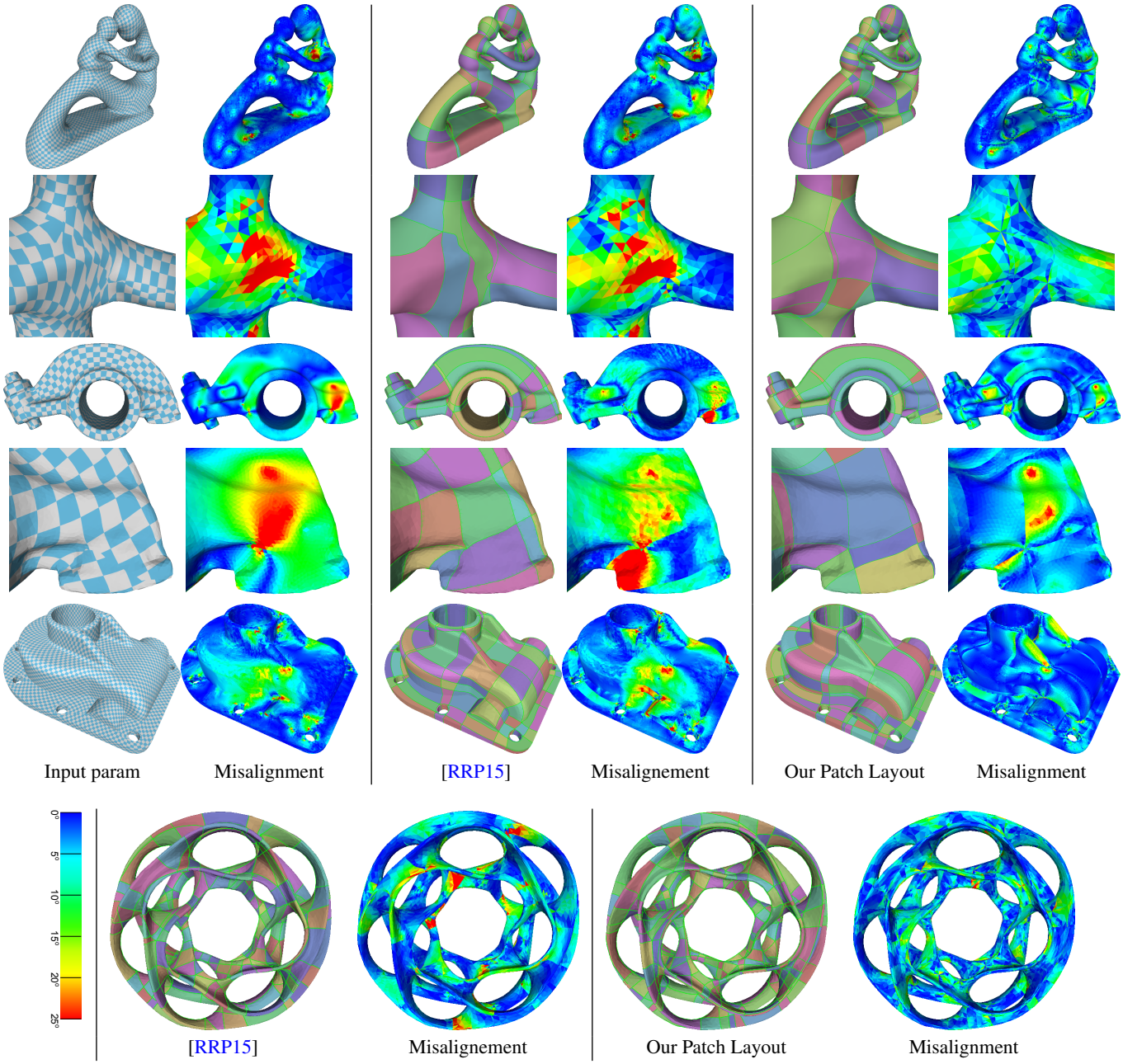


Figure 15: *Comparison with Perfect Matching Quad Layouts [RRP15]: the patch layout of [RRP15] can be arbitrarily distorted by the underlying parametrization; our method traces field-coherent paths directly on the surface, producing a patch layout with a better field alignment. No t-junctions in all layouts.*

6. Conclusions

We have presented a novel method for the automatic computation of a field-aligned, low resolution, quadrilateral patch layout of an input shape. The method takes in input a triangle mesh endowed with a cross field and it does not require any intermediate quadrangulation or parametrization. It is based on the construction of a directed graph, a Dijkstra propagation and a global optimization step. The final layout is a bijective parametrization and can be used for quadrangulation or texture mapping. We demonstrated that by keeping the field coherency, the proposed method excels in robustness and adherence to the original field with respect to previous techniques, being able to create quad layouts that conform to highly complicated geometries and fields.

The proposed method can be extended to become a user-assisted

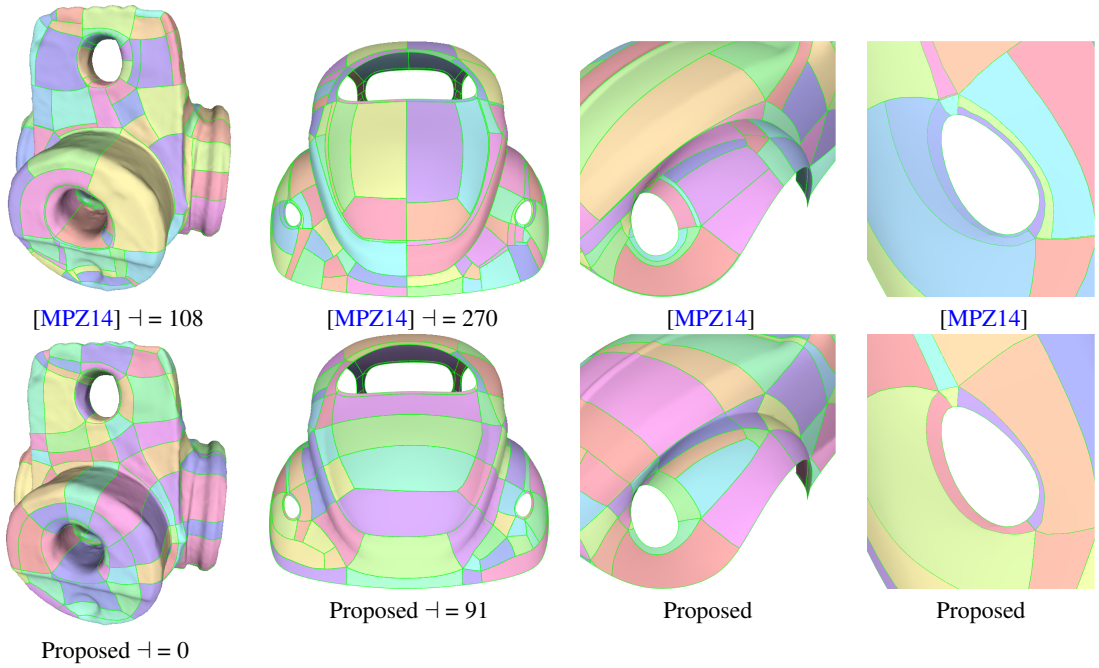


Figure 16: *Our method reduces the total number of t-junctions with respect to [MPZ14]. As a consequence, the zero loop generated by [MPZ14] (about the hole in the close-up) vanishes implicitly in our final layout (right).*

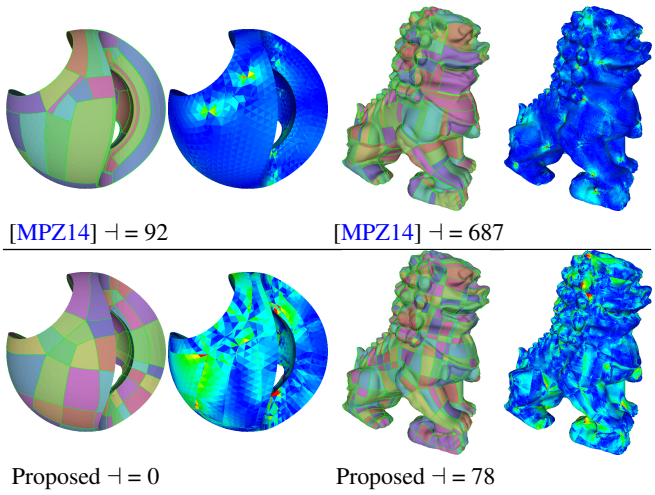


Figure 17: *The proposed method (Bottom) may derive a patch layout almost as accurate as [MPZ14] (Top), with the main advantage that it introduces much fewer t-junctions in the final layout*

one by supporting interactive selection of separatrices during the patch layout optimization process: since each optimization cycle takes results from the previous one as constraints, manual and automatic cycles can be combined freely. Moreover, the method can be generalized to work on arbitrary k-RoSy fields, hence it can be adapted to derive regular triangulations or hexagonalizations of input shapes.

References

[Ach09] ACHTERBERG T.: SCIP: Solving constraint integer programs. *Mathematical Programming Computation 1*, 1 (2009), 1–41. 6, 7

[BCE*13] BOMMES D., CAMPEN M., EBKE H.-C., ALLIEZ P., KOBBELT L., ET AL.: Integer-grid maps for reliable quad meshing. *ACM Trans. Graph. 32*, 4 (2013). 2, 3, 8, 9

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

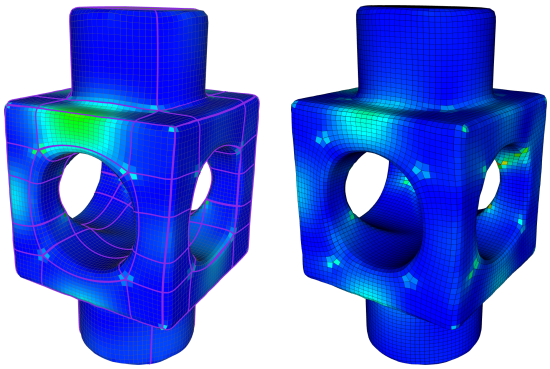
[BLP*13] BOMMES D., LÉVY B., PIETRONI N., PUPPO E., SILVA C., TARINI M., ZORIN D.: Quad-mesh generation and processing: A survey. *Comput. Graph. Forum* (2013). 1, 2

[BMRJ04] BOIER-MARTIN I., RUSHMEIER H., JIN J.: Parameterization of triangle meshes over quadrilateral domains. In *Proc. Eurographics Symposium on Geometry Processing* (2004), pp. 193–203. 2

[BZK09] BOMMES D., ZIMMER H., KOBBELT L.: Mixed-integer quadrangulation. *ACM Trans. Graph. 28*, 3 (2009), 77. 2, 9, 12

model	input param	[RRP15]	Ours
fert	3.64	9.80	5.9
rocker	2.93	5.29	2.6
casting	2.6	5.15	2.32
heptoroid		12.0	9.45
		[MPZ14]	Ours
sculpt		1.49	4.46
chinese		1.57	6.76

Table 2: *Comparison of the misalignment of our patch layout with respect to [RRP15] and [MPZ14]. Values indicate the percentage of faces whose misalignment is over* $\pi/16$.



Ours [BZK09]

Figure 18: *Comparison with the quadrangulation produced by [BZK09].*

[CBK12] CAMPEN M., BOMMES D., KOBBELT L.: Dual loops meshing: quality quad layouts on manifolds. *ACM Trans. Graph. 31*, 4 (2012), 110. 2, 5

[CBK15] CAMPEN M., BOMMES D., KOBBELT L.: Quantized global parametrization. *ACM Trans. Graph. 34*, 6 (2015), 192:1–192:12. 2

[CHCH06] CARR N. A., HOBEROCK J., CRANE K., HART J. C.: Rectangular multi-chart geometry images. In *Proc. Eurographics Symposium on Geometry Processing* (2006), pp. 181–190. 2

[CHK13] CAMPEN M., HEISTERMANN M., KOBBELT L.: Practical anisotropic geodesy. *Comput. Graph. Forum 32*, 5 (2013), 63–71. 2

[CK14a] CAMPEN M., KOBBELT L.: Dual strip weaving: Interactive design of quad layouts using elastica strips. *ACM Trans. Graph. 33*, 6 (2014), 183:1–183:10. 3

[CK14b] CAMPEN M., KOBBELT L.: Quad layout embedding via aligned parameterization. *Comput. Graph. Forum 33*, 8 (Dec. 2014), 69–81. 7

[CWW13] CRANE K., WEISCHEDEL C., WARDETZKY M.: Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph. 32*, 5 (Oct. 2013), 152:1–152:11. 2

[DISC09] DANIELS II J., SILVA C. T., COHEN E.: Semi-regular quadrilateral-only remeshing from simplified base domains. *Comput. Graph. Forum 28*, 5 (July 2009), 1427–1435. 2

[DVPSH15] DIAMANTI O., VAXMAN A., PANOZZO D., SORKINE-HORNUNG O.: Integrable PolyVector fields. *ACM Trans. Graph. 34*, 4 (2015). 3

[HZ00] HERTZMANN A., ZORIN D.: Illustrating smooth surfaces. In *Proc. 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00)* (2000), pp. 517–526. 2

[JLW10] JI Z., LIU L., WANG Y.: B-mesh: A modeling system for base meshes of 3d articulated shapes. *Computer Graphics Forum (Proceedings of Pacific Graphics) 29*, 7 (2010), 2169–2178. 3

[Kan00] KANAI T.: Approximate shortest path on a polyhedral surface and its applications. In *Computer-Aided Design* (2000), pp. 241–250. 2

[KNP07] KÄLBERER F., NIESER M., POLTHIER K.: QuadCover: Surface Parameterization using Branched Coverings. *Comput. Graph. Forum 26*, 3 (2007), 375–384. 2

[KS98] KIMMEL R., SETHIAN J. A.: Computing geodesic paths on manifolds. *Proc. Natl. Acad. Sci. USA* (1998), 8431–8435. 2

[Lip12] LIPMAN Y.: Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph. 31*, 4 (2012), 108. 3

[LMS97] LANTHIER M., MAHESHWARI A., SACK J.-R.: Approximating weighted shortest paths on polyhedral surfaces. In *Proc. Thirteenth Annual Symposium on Computational Geometry* (1997), ACM, pp. 485–486. 2

[MMP87] MITCHELL J. S. B., MOUNT D. M., PAPADIMITRIOU C. H.: The discrete geodesic problem. *SIAM J. Comput. 16*, 4 (Aug. 1987), 647–668. 2

[MPKZ10] MYLES A., PIETRONI N., KOVACS D., ZORIN D.: Feature-aligned t-meshes. *ACM Trans. Graph. 29*, 4 (2010), 117:1–117:11. 2

[MPZ14] MYLES A., PIETRONI N., ZORIN D.: Robust field-aligned global parametrization. *ACM Trans. Graph. 33*, 4 (July 2014), 135:1–135:14. 2, 3, 7, 9, 11

[MTP*15] MARCIAS G., TAKAYAMA K., PIETRONI N., PANOZZO D., SORKINE O., PUPPO E., CIGNONI P.: Data-driven interactive quadrangulation. *ACM Trans. Graph. (Siggraph 2015) 34*, 65 (2015). 3

[PTP*15] PIETRONI N., TONELLI D., PUPPO E., FROLI M., SCOPIGNO R., CIGNONI P.: Statics aware grid shells. *Comput. Graph. Forum 34*, 2 (2015), 627–641. 7, 9

[PTSZ11] PIETRONI N., TARINI M., SORKINE O., ZORIN D.: Global parametrization of range image sets. *ACM Trans. Graph. (SIGGRAPH Asia 20119 30*, 6 (2011). 2

[RRP15] RAZAFINDRAZAKA F. H., REITEBUCH U., POLTHIER K.: Perfect Matching Quad Layouts for Manifold Meshes. *Comput. Graph. Forum* (2015). doi:10.1111/cgf.12710. 2, 3, 5, 6, 9, 10, 11

[RS14] RAY N., SOKOLOV D.: Robust polylines tracing for n-symmetry direction field on triangulated surfaces. *ACM Trans. Graph. 33*, 3 (June 2014), 30:1–30:11. 2

[RVLL08] RAY N., VALLET B., LI W., LÉVY B.: N-Symmetry direction field design. *ACM Trans. Graph. 27* (2008), 2. 2

[SSK*05] SURAZHSKY V., SURAZHSKY T., KIRSANOV D., GORTLER S. J., HOPPE H.: Fast exact and approximate geodesics on meshes. *ACM Trans. Graph. 24*, 3 (2005), 553–560. 2

[TPP*11] TARINI M., PUPPO E., PANOZZO D., PIETRONI N., CIGNONI P.: Simple quad domains for field aligned mesh parametrization. In *ACM Trans. Graph.* (2011), vol. 30, pp. 142:1–142:12. 1, 2

[TPSHSH13] TAKAYAMA K., PANOZZO D., SORKINE-HORNUNG A., SORKINE-HORNUNG O.: Sketch-based generation and editing of quad meshes. *ACM Trans. Graph. 32*, 4 (2013), 97:1–97:8. 3

[ZZFJ14] ZHUANG Y., ZOU M., F N., JU T.: Anisotropic Geodesics for Live-wire Mesh Segmentation. *Comp. Graph. Forum* (2014). 2

[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. 3