EUROGRAPHICS 2021 / N. Mitra and I. Viola
(Guest Editors)

Volume 40 (2021), Number 2

Quad Layouts via Constrained T-Mesh Quantization

M. Lyon[1], M. Campen[2], and L. Kobbelt[1]

[1]RWTH Aachen University, Germany
[2]Osnabrück University, Germany

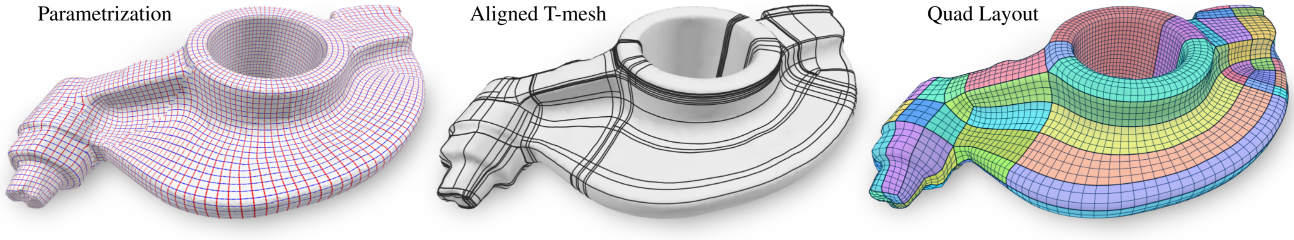


Figure 1: *Given a seamless parametrization of a surface as input we construct a non-conforming T-mesh aligned with it. A modified version of the motorcycle graph is employed for this purpose, in which traces do not stop at the first collision. We solve an integer linear program to assign integers to the arcs of this T-mesh. By carefully constraining this quantization, the result implies a coarse conforming quad layout whose separatrices deviate less than a user given bound from the directions prescribed by the seamless input parametrization.*

Abstract

We present a robust and fast method for the creation of conforming quad layouts on surfaces. Our algorithm is based on the quantization of a T-mesh, i.e. an assignment of integer lengths to the sides of a non-conforming rectangular partition of the surface. This representation has the benefit of being able to encode an infinite number of layout connectivity options in a finite manner, which guarantees that a valid layout can always be found. We carefully construct the T-mesh from a given seamless parametrization such that the algorithm can provide guarantees on the results' quality. In particular, the user can specify a bound on the angular deviation of layout edges from prescribed directions. We solve an integer linear program (ILP) to find a coarse quad layout adhering to that maximal deviation. Our algorithm is guaranteed to yield a conforming quad layout free of T-junctions together with bounded angle distortion. Our results show that the presented method is fast, reliable, and achieves high quality layouts.

CCS Concepts

• ***Computing methodologies*** → ***Computer graphics; Mesh models; Mesh geometry models;*** *Shape modeling;*

1. Introduction

Quad meshes are the preferred discrete surface representation for many shape modeling applications of design and engineering alike. Therefore, the automatic generation of such meshes has been an ongoing topic of research in computer graphics. For a high quality quad mesh, individual elements should have angles close to $\pi/2$ and be aligned in certain ways, e.g. to the underlying surface's principal curvature directions. A variety of approaches have been explored for the generation of quad meshes [BLP*13]. A class of algorithms with a particular focus on element shape and alignment quality is that of parametrization-based field guided methods [KNP07, BZK09, KMZ11, BCE*13, PPTSH14, ESCK16].

An additional quality criterion that these methods, however, do not explicitly promote is the global structure of the mesh, in particular the simplicity of the mesh's so-called *block structure* or *base complex* [BLP*13]§1.1. This aspect is closely related to the question how the irregular vertices are connected in the mesh by sequences of edges. This connectivity constitutes the mesh's *quad layout* [Cam17]. If the quad mesh has a simple, i.e., a coarse layout it can be viewed as a "mesh of meshes", a coarse quad partition with finer regular quad grids inside each patch (Figure 1 right). This enables the construction of mesh hierarchies, the structured parametrization of the mesh over simple domains, or the definition of spline spaces on top of the mesh [TPP*11, MAB*19, HSJ*20].

In this paper we present an algorithm for the creation of coarse quad layouts on 2-manifold surfaces. Such layouts can then, for instance, be refined to block-structured quad meshes, or be passed as connectivity constraints to the above mentioned (natively layout-unaware) quad mesh generation techniques. The layout construction – akin to the above parametrization-based mesh generation methods – is built on top of a seamless surface parametrization. This allows us to offer explicit control over the layout's singularity configuration and its directional alignment, to reliably yield valid layouts, and to share a common foundation with mesh generation techniques for seamless integration. As a particular feature, our method offers explicit and precise control over the balance between the two key opposing objectives inherent to quad layouts: coarseness and directional alignment.

Problem Statement: Given a seamless surface parametrization with arbitrary singularities on a surface of arbitrary topology, generate a coarse quad layout with exactly these singularities as irregular nodes such that its arcs (also called separatrices) do not directionally deviate from the parametrization's isolines by more than a user-given angular bound α.

After reviewing previous methods for the creation of coarse quad layouts in Section 2, we describe in Section 3 how we construct a T-mesh by tracing parametric iso-lines of a seamless parametrization taken as input. In our method we implicitly encode layout connectivity via a discrete function on this T-mesh. Section 4 details this encoding and shows how non-negative integers can be assigned to the T-mesh's arcs such that a valid and high quality layout is implied. Such an assignment, called quantization, can be found by solving an integer linear program as presented in Section 5, before ultimately making the layout explicit in Section 6. We show the effectiveness of our algorithm on a variety of examples in Section 7. Figure 1 illustrates the process.

2. Related Work

Early work that involved the generation of quad layouts as a substep produced rather unstructured layouts without any particular form of shape-aware directional alignment [EH96, BMRJ04]. Later this problem received dedicated attention and subsequent methods often take some form of directional guidance into account, whether by means of the underlying surface's principal curvature directions, or more flexibly and controllably by specifically designed or prescribed cross fields, quad meshes, or seamless parametrizations.

Layout Simplification. For instance, the method by Bommes et al. [BLK11] takes as input a quad mesh with a possibly dense base complex and iteratively modifies it so as to remove certain helical connectivity patterns – which are one, though not the sole cause of low quality base complexes. A coarser quad layout can then be extracted from the modified quad mesh.

Tarini et al. [TPP*11] follow a similar strategy but enable more general modifications by working directly on separatrices (i.e., the paths forming the layout's arcs, its patches' borders). They iteratively improve a layout energy which is based on length and direction deviation of the separatrices by removing a separatrix and bringing the then incomplete layout back into a (coarser) complete state by a series of separatrix reconnections and an insertion.

Instead of starting with conforming dense quad layouts, Viertel et al. [VOS19] start from an initial non-conforming layout, a T-mesh with many T-junctions. It is obtained by tracing streamlines of a surface cross field. In this layout so-called chord collapses are applied greedily, in order from narrow to wide, while excluding collapses that would result in too much directional deviation. Particularly on complex or closed surfaces, T-junctions may remain in the final layout, making it non-conforming.

These methods have in common that modifications are applied iteratively in a greedy fashion.

Layout from Separatrix Candidates. A common strategy to create layouts from scratch is based on finding a set of separatrix candidates, i.e., paths connecting pairs of prescribed (irregular) layout nodes in topologically distinct ways, from which a subset is then selected to define a complete layout.

Razafindrazaka et al. [RRP15] trace isolines of a seamless parametrization, starting from its singularities (which form the layout's irregular nodes). Whenever two traces meet, this implies a separatrix candidate between their two origin singularities. Each candidate is associated with a cost, penalizing directional deviation from the parametric isolines. A binary problem is then solved to select a cost-minimizing subset that properly connects all singularities without crossing in improper ways. In theory the candidate set is infinite; in practice one needs to restrict to a finite subset. This may preclude the existence of a valid solution (or a high-quality solution). Tracing up to a maximum distance limit is reported to commonly work well, but the existence of a solution is not guaranteed (unless trial-and-error with increasing distance limit is performed).

Pietroni et al. [PPM*16] follow a similar approach but create separatrix candidates based on a cross field [VCD*16] rather than a parametrization (which is harder to obtain with the required properties [CSZZ19]). Similar to [RRP15] they solve a binary linear program to choose a non-conflicting subset of these. As a consequence of not deriving directional guidance from a parametrization (which corresponds to an integrable cross field) but rather from a generic cross field, a complete conforming quad layout cannot be guaranteed; T-junctions have to be accepted, similar to the approach of Viertel et al. [VOS19] discussed above. T-junctions may be reduced by increasing the number of separatrix candidates established for each singularity or by iterating the process with fixed partial layout, but complete removal can only be achieved by inserting additional irregularities.

For both these methods, runtime, quality, and even the existence of a solution depend on the precomputed set of separatrix candidates. If the set is chosen too small there may be no valid subset. Increasing the candidate set size at the cost of increased runtime only increases the probability of program feasibility. In contrast, our formulation, instead of using a binary program which picks separatrices from a finite set of candidates, employs an integer program which can choose from an infinite set of separatrices and is always guaranteed to be feasible.

[RP17] and [ZZY16] propose further candidate set based methods, but start from a quad mesh as input. This effectively enables a fallback to the quad mesh itself or its base complex as a valid (though commonly rather dense) output layout in case no other solution is found due to the involved restriction.

Dual Approach. Campen et al. [CBK12] tackle the problem in the dual space and search for a set of dual loops which separate all singularities. While the dual setting allows for relatively simple conditions guaranteeing a valid primal layout it does not allow explicit control over the geometric quality of the implied separatrices.

Mesh as Layout. An alternative approach to create coarse quad layouts is to use a generic quad mesh generation method (without explicit layout considerations) and aim for a mesh with very large elements – which can then be considered a coarse layout. While some of these methods are robust enough to operate reliably under this extreme requirement (e.g. [BCE*13, CBK15]), it is typically difficult for these methods to control the quality of the resulting layout under these circumstances (cf. Section 7.1).

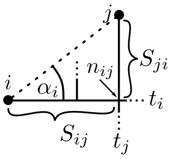
3. T-Mesh Construction

A motorcycle graph is a cell decomposition of a surface [EGKT08]. Campen et al. [CBK15, MC19] show how a rectangular partition of a surface can be constructed by tracing motorcycles starting at singularities along the iso-lines of a seamless parametrization. The resulting T-mesh $\mathcal{T} = (\mathcal{N}, \mathcal{A}, \mathcal{P})$ consists of nodes $\mathcal{N}$ for every singularity and intersection of traces, arcs $\mathcal{A}$ consisting of the segments of a trace between two nodes, and patches $\mathcal{P}$ representing rectangular regions bounded by arcs.

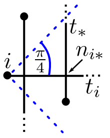
Such a motorcycle graph forms the basis of our quantization (cf. Section 4) which assigns integer lengths to the arcs and thereby defines the connectivty of the resulting layout.

We adopt a construction similar to [CBK15], with one important difference: in our case motorcycles may survive a crash with an existing trace and continue driving. In such a case a regular valence 4 node is formed at the intersection. Before defining the criterion that determines this, we establish a few definitions and notation.

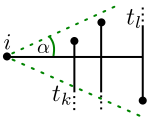
Notation. Given two traces t_i and t_j starting in singularities i and j, respectively, we refer to the node created at their intersection as $n_{ij} \in \mathcal{N}$. (For simplicity of notation we ignore the fact that two traces may intersect more than once; it will be clear from the context which intersection node is referred to.) We define $S_{ij} \subset \mathcal{A}$ as the set of arcs between the start of trace t_i and node n_{ij}, and $l_{ij} \in \mathrm{R}$ as the total parametric length of those arcs. The two arc sets S_{ij} and S_{ji} form the legs of a right triangle. Let $\alpha_{ij} \in (-\frac{\pi}{2}, \frac{\pi}{2})$ be the signed (ccw) angle of that triangle at the start of trace t_i.



We further give a dedicated name, n_{i*}, per trace t_i to the intersection closest to the origin of t_i which satisfies $l_{i*} > l_{*i}$, i.e. the first intersection of t_i with a trace that starts inside the $\pi/2$-sector (blue) around t_i (see inset). In the classical motorcycle graph a trace t_i would stop exactly at n_{i*}. By contrast, we keep on tracing and only stop based on the following criterion.



Stopping Criterion. Given a user defined angular bound α, a trace is stopped as soon as it intersected two traces t_k and t_l such that $\alpha_{ik} \in [0, \alpha]$ and $\alpha_{il} \in [-\alpha, 0]$, i.e., as soon as on both sides of trace t_i a trace is found such that the formed triangles are contained in the sector of half-angle α around t_i. With this stopping criterion we ensure that we reach the traces of at least one singularity on each side of the trace which, intuitively, could be connected to singularity i by a separatrix which respects the maximum angular deviation bound. We will see in detail in Section 4.3 how this construction helps to guarantee a maximum separatrix deviation. Figure 2 shows a few examples of the resulting T-meshes for different α and how tighter bounds lead to longer traces.



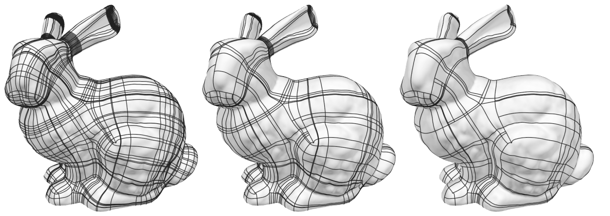


Figure 2: *T-meshes on the* BUNNY *model created for angle deviation bound* $\alpha = 5^\circ, 15^\circ, 35^\circ$*, respectively.*

4. Quantization

A quantization of a T-mesh is an assignment of a non-negative integer q_i to every arc $a_i \in \mathcal{A}$. These values are to be interpreted as parametric length specification for the arcs, virtually overriding their length in the input parametrization. Since every arc is associated with a parametric iso-line, the quantization implies an assignment of integer parametric distances between singularities [CBK15]. These distances can be used as constraints for a global re-parametrization of the surface where every singularity is located on an integer position, enabling the extraction of a quad mesh [CBK15, LCBK19]. This quad mesh's base complex defines a quad layout. The edges of the base complex – the separatrices – connect singularities that lie on the same parametric iso-line. Effectively, the quantization implies which singularities are connected by separatrices by assigning zero lengths to certain arcs. Depending on the T-mesh structure and the quantization, singularities may be connected by separatrices corresponding to different parametric directions (cf. Figure 3(left, middle)) or not (cf. Figure 3(right)).

While the above previous works aim to find a quantization which approximates the arcs' original parametric lengths, we detail in this section how we instead find a quantization that promotes a coarse quad layout. We begin by discussing two important properties of a quantization – *validity* and *consistency* – in the following section. We continue by presenting a sufficient condition which guarantees validity, and finally describe which additional constraints we set up to enforce a high quality quad layout.

4.1. Consistency and Validity

In order to compute a valid (locally injective, fold-over free) parametrization that adheres to the singularity distance constraints defined by the quantization, two properties need to be fulfilled. The quantization needs to be consistent and valid [CBK15].

Figure 3: Layouts resulting from different quantizations. Singularities (dots) are separated in vertical direction leading to an angle $\alpha > \pi/4$ compared to the desired direction (left). With our separation constraint, singularities are separated in the direction corresponding to the larger difference in the seamless parametrization always leading to separatrix angles smaller than $\pi/4$ (middle). If both separatrices form an unacceptably large angle, singularities need to be separated in both directions (right). Note that the actual quantization (beyond zero vs non-zero) is less important for the layout, consisting of the (bold) edges of the base complex only.

A quantization is *consistent* if pairs of opposite sides of each T-mesh patch are quantized to the same length. This property ensures that all patches remain rectangular in the parametric domain.

A quantization is *valid* if the distance between any two singularities is strictly positive, such that they do not collapse parametrically. In terms of the quantization this means there must be no arc path with total quantized length of 0 between any two singularities.

Let $d_{ij} \in \mathbb{R}^2$ be the (u,v)-difference between singularities i and j measured in the seamless parametrization (in some path homotopy class) and $q_{ij} \in \mathbb{Z}^2$ be the quantized difference vector. W.l.o.g. assume that $d^u_{ij} > 0$ and $|d^u_{ij}| \geq |d^v_{ij}|$. To ensure validity it is sufficient to require $q^u_{ij} > 0 \lor q^v_{ij} \neq 0$ [CBK15]. We propose to use instead the simpler sufficient condition of $q^u_{ij} > 0$, i.e. singularities are separated by the quantization in the dominant separation direction in the input parametrization. While this condition is stricter, what we effectively exclude are quantizations with $q^u_{ij} = 0$ and $q^v_{ij} \neq 0$ (Figure 3 left); thereby we already ensure a maximal separatrix deviation of $\pi/4$. Even more practically relevant, this disjunction-less condition is easier to formulate and more efficient to handle.

Campen et al. propose for their iterative quantization improvement to test validity after each change by explicitly searching singularity connecting arc paths with quantized length of zero. In the worst case this test is in $O(n^2)$ and in general not easily applicable in the context of solving an integer linear program. In the following section we present how separation of singularities can be guaranteed using only one linear constraint per trace.

4.2. Singularity Separation

To guarantee the separation of singularities in the quantization we propose a simple criterion based on the following lemma:

Lemma 1. If for every trace t_i one of the arcs in S_i is quantized to at least 1, the quantization is valid.

Here, S_i is the set of arcs between the start of t_i and n_i, i.e. the first intersection of t_i with an earlier trace (cf. Section 3).

Proof. We show that every singularity in the $\pi/2$-sector centered at a singularity i and around the positive u direction is separated from i. The same argument then trivially holds for all other parametric directions as well.

Let j be the singularity at the start of t. Then, j is separated from i in u direction by at least 1 according to the lemma's premise. Consider another singularity k within the $\pi/2$-sector and its trace t_k in negative (or positive) v direction towards t_i. If t_k intersects the parametric iso-line of t_i it must do so further away than t_n since t is the intersection closest to i by definition. Since the intersection lies behind that of t, k is separated by at least 1 in u direction as well. If, on the other hand, the trace of k does not intersect the parametric iso-line of t_i it must have been stopped before reaching it. Since at least one of the arcs of t_k is quantized to at least one according to the condition in the lemma, k is separated from i in positive (or negative) v direction. □

Note that Lemma 1 does not depend on our special T-mesh construction and also holds if traces stop at the first intersection.

We conclude that it is sufficient to require one arc of S_i to be quantized to a strictly positive number for each trace t_i to ensure validity, i.e., one constraint per trace is sufficient.

4.3. Layout Constraints

In the previous section we detailed how simple constraints on the minimal quantization of certain arcs guarantee a valid quantization. Computing the smallest quantization adhering to these constraints results in a coarse quad layout. However, the separatrices of that layout may deviate up to $\pi/4$ from the directions of the seamless parametrization (cf. Figure 3 b). In particular the separatrix corresponding to a trace t_i will connect to the first singularity within the $\pi/2$-sector which is not separated in the direction orthogonal to t_i. We propose a simple solution to create layouts in which the separatrices do not deviate from the intended directions more than a user specified maximum of α by enforcing the additional separation of offending singularities in orthogonal direction (cf. Figure 3 c).

Given two intersecting traces t_i and t_j with $l_{ij} \leq l_{ji}$ originating from singularities i and j, we defined in the previous sections constraints that ensure the arc set S_{ij} is quantized to at least 1, separating i and j in the direction of t_i. If S_{ji} will be quantized to 0, i and j would lie on the same iso-parameter line and would therefore be connected by a separatrix (unless there is a singularity closer to i which also lies on the same iso-parameter line). If angle α_{ij} of this separatrix is larger than the user defined bound α we need to prevent such a quantization. By additionally separating i and j in the direction of t_j it is ensured that the corresponding separatrix is not formed. We therefore include the additional constraint that S_{ji} is quantized to at least 1 for every pair of intersecting traces t_i and t_j with $|\alpha_{ij}| > \alpha$.

In the light of this, we can now explain the rationale behind our choice of stopping criterion in the T-mesh construction, continuing a trace t_i until at least two traces t_k and t_l are intersected with $\alpha_{ik} \in [0,\alpha]$ and $\alpha_{il} \in [-\alpha,0]$ (cf. Section 3). We want to guarantee that all separatrix deviations are below the user defined bound. With the constraints above, any separatrix created between singularities whose traces intersect fulfills this. But what if the quantization implies none of them, i.e. S_{ji} is quantized to at least 1 for all traces t_j that intersect t_i, which may be necessary due to a combination of separation and consistency constraints? In that case the separatrix will lie in the corridor (blue) between the two potential separatrices connecting the start of t_i to the starts of t_k or t_l. Since both these potential separatrices satisfy the bound, the actually implied separatrix (blue dotted line) lying in between satisfies it as well.

4.4. Feature Lines & Boundaries

Some models contain sharp creases for which it is typically desirable that they are represented by arcs in a layout. Similarly, boundaries of models should be represented by arcs. Both these cases as well as arbitrary, user defined feature curves are supported by our method by aligning the parametrization with these features and tracing them with motorcycles to integrate them into our T-mesh [CBK15]. To ensure that the traced separatrix is not diverted towards another singularity away from the feature curve it is enough to simply add layout constraints (cf. Section 4.3) for all intersecting traces. This can be interpreted as prescribing a maximum separatrix deviation of $\alpha = 0$ for all traces that follow a feature curve. Figure 9 shows examples where boundary alignment is enforced this way.

5. Integer Linear Program

We established how a quantization can be constrained such that no separatrix in the implied quad layout deviates more than a user given bound from the intended direction. In the space of feasible quantizations respecting these constraints, we are looking for a quad layout as coarse as possible. In this section we discuss how such a quantization can be found efficiently by solving an integer linear program (ILP). We first describe the basic integer linear program, which can be constructed in a straightforward fashion following the previous discussion. After that we discuss how the program size can be reduced for better performance of the solver.

5.1. Definition

Our ILP uses one integer variable $q_i \in \mathbb{Z}$ for every arc $a_i \in A$ which represents the quantization of this arc. Every arc $a_i \in A$ requires a non-negative quantization:

$$q_i \geq 0 \tag{1}$$

For consistency (cf. Section 4.1) we add the following constraint (analogous to previous work on consistent interval assignment for non-conforming partitions [CBK15, ULP15]) for each pair of arc sets S and S_o which make up two opposite sides of a patch:

$$\sum_{a_i \in S} q_i - \sum_{a_j \in S_o} q_j = 0 \tag{2}$$

To ensure validity we add the following validity constraints which ensure that for each trace t_i one of its arcs between its origin and the first intersection n_i with a trace starting in the $\pi/2$-sector around t_i is quantized to at least 1 (cf. Section 4.2):

$$\sum_{a_k \in S_i} q_k \geq 1 \quad \forall \text{ traces } t_i \tag{3}$$

The layout constraints of Section 4.3 have a very similar form but are created for every intersection of two arcs forming a triangle with angles larger than α to prevent the creation of separatrices with excessive deviation:

$$\sum_{a_k \in S_{ji}} q_k \geq 1 \quad \forall n_{ij} \text{ with } \frac{l_{ji}}{l_{ij}} > \tan\alpha \tag{4}$$

Finally, to promote layout coarseness, we define the objective to be minimized as

$$E = \sum_{a_i \in A} l_i^{\perp} q_i \rightarrow \min \tag{5}$$

where $l_i^{\perp}$ is half the parametric distance between the two arcs opposite of a_i (or half the parametric distance between the one opposite arc and a_i itself if a_i is boundary). Since the quantization of an arc specifies the number of quad strips that pass through this arc orthogonally (in the quad mesh implied by the quantized parametrization), this energy corresponds to the total length of the layout's quad strips.

An integer linear program is feasible if an assignment of variables exists such that all constraints are satisfied. Campen et al. [CBK15] show that a consistent quantization always exists in which all arcs are quantized to at least 1. Such a baseline quantization trivially fulfills all our constraints.

5.2. ILP Size

The size of the integer linear program described in the previous section depends largely on the size of the constructed T-mesh. For every arc there is one integer variable representing its quantized length (1), for every patch there are two consistency constraints (2), every trace adds a validity constraint (3), and for every node created

Figure 4: On the BUNNY mesh with 192 traces the number of arcs quickly grows for decreasing α values to 189k at $\alpha = 0.25°$ but the number of integer variables remains around 252 (left). Since motorcycles need to be traced further and the resulting T-mesh consists of more elements for decreasing α timings go up for both T-mesh construction and setup of the ILP (to 2.9 s and 0.4 s, respectively, at $\alpha = 0.25°$) but the time to solve the problem remains around 0.03 s.