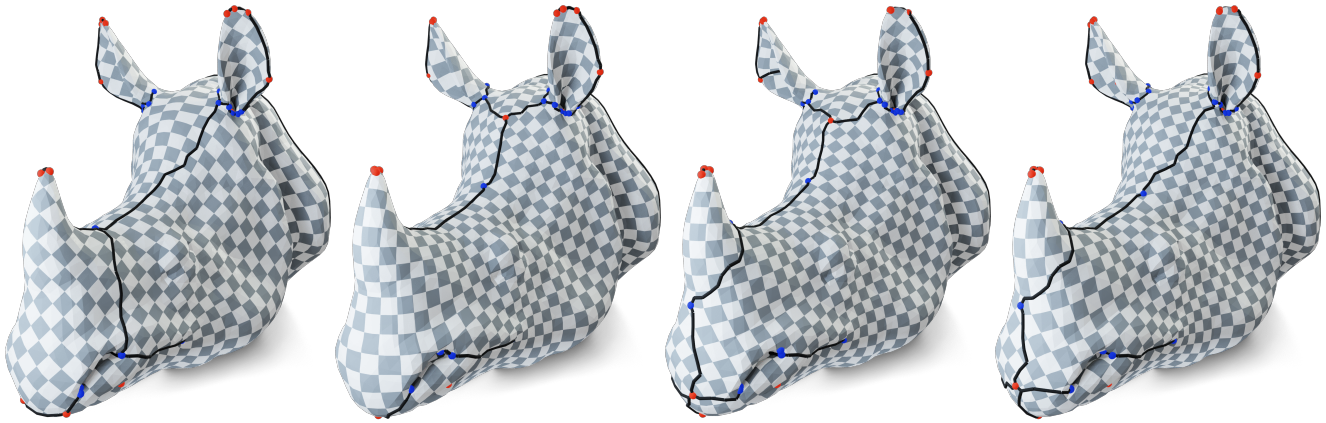
Seamless Global Parametrization in a Single Optimization

Guillaume Coiffier, Etienne Corman

Seamless Global Parametrization in a Single Optimization

GUILLAUME COIFFIER, Université de Lorraine, CNRS, Inria, LORIA, France
ETIENNE CORMAN, Université de Lorraine, CNRS, Inria, LORIA, France



60 cones	60 cones	70 cones	62 cones
iso = 0.168	iso = 0.198	iso = **0.151**	iso = 0.181
shear = 0.117	shear = 0.071	shear = 0.091	shear = **0.065**
stretch = 1.218	stretch = 1.128	stretch = 1.165	stretch = **1.113**

(a) Fixed Frame Field (b) Shear (c) Isometric (d) LSCM

Fig. 1. We propose to solve a single optimization problem for *simultaneously* computing a frame field and a global seamless parametrization. This allows the singularity cones to be moved during optimization in order to minimize any differentiable distortion energy. Our results produce less distorted parametrizations than two-steps methods relying on integrating a precomputed frame field. This figure presents our parametrizations of the Rhino head model computed: (a) by integrating a smooth frame field, (b) by reducing the shear, (c) by maximizing isometric deformations, (d) by minimizing the LSCM energy [Lévy et al. 2002] effectively reducing the stretch. We represent positive $\pi/2$ cones in red and negative $-\pi/2$ cones in blue. Distortion measurements are computed as a mean over all triangles.

The challenge of computing high quality seamless parametrizations is to account for both the optimal placement of singularity cones and distortion minimization. Existing methods rely either on greedy solvers or on a simplified two steps problem, first finding the cone positions using a cross field and second optimizing the parametrization under fixed topology. Both approaches, however, loosen the link between cone placement and distortion, effectively leading to suboptimal solutions. We instead formulate the problem of global seamless parametrization as a single continuous optimization problem where singularities may appear anywhere inside triangles. To achieve this, the input mesh is subdivided into charts centered at vertices whose orientations are related by rotations along edges. We then exhibit a set of continuous constraints for the parametrization to be seamless, locally injective and aligned with feature and boundary edges. Formulated as a non-linear least-squares problem, these constraints can be easily optimized. We compare our algorithm to previous techniques on a variety of smooth and CAD models, often achieving lower distortion and more practical cone placement for subsequent quad remeshing.

CCS Concepts: • **Computing methodologies** → **Mesh geometry models**; **Volumetric models**.

Additional Key Words and Phrases: Seamless parametrization, quad-meshing, hex-meshing

Authors' addresses: Guillaume Coiffier, Université de Lorraine, CNRS, Inria, LORIA, Nancy, 54000, France, guillaume.coiffier@inria.fr; Etienne Corman, Université de Lorraine, CNRS, Inria, LORIA, Nancy, 54000, France, etienne.corman@cnrs.fr.

1 INTRODUCTION

Meshes play a crucial role in numerical algorithms for geometry processing and scientific computing. They decompose the space into atomic cells allowing to store a discrete approximation of a continuous process. Depending on the application, meshes with different properties are needed. Quadrangle meshes have this unique property that they can tile space in a grid-like manner everywhere except at a few local exceptions called singularities or cones.

Singularities are bound to appear to compensate the topology of non-torus-like surfaces. These extraordinary points also have a great impact on the quad mesh distortion, *i.e.* how close to a perfect square are the quads. Depending on the surface curvature, the number of cones and their positions can drastically increase the mesh quality. And, on this matter, the fewer is not always the better. In fact, as singularity indices must be integer multiple of $1/4$, finding optimal positions minimizing the overall distortion is challenging and often leads to sophisticated integer programming problems.

To overcome this apparent complexity, current quad-meshing pipelines often relies on two-steps: first choosing the singularity locations using a cross field or other proxies and then compute a *global seamless parametrization* from which quads will be extracted. This procedure, however, loosens the link between cone placement and the distortion of the final mesh. As a consequence, quads can

stray far away from squares because the connectivity of the quad-mesh is optimized almost independently of its geometry.

This limitation is the starting point of this paper: we propose a *single step* seamless parametrization algorithm. In our setup, cones are positioned based on distortion minimization as part of a *global* optimization problem. This approach is in total opposition to greedy methods used in previous work heavily relying on local considerations. Moreover, we do *not* need integer variables making our system of equations solvable by any off-the-shelf continuous optimizer.

To achieve this, we rely on two key ingredients. First, we consider vertex-based cross field so that singularities can appear and move *inside* triangles, making the parametrization more flexible than face based frame field integration. Second, we unveil a necessary and sufficient system of equations for seamless parametrization which is independent of the cross field symmetries. This is done by using the so-called *parallel transport* introduced in previous work [Crane et al. 2010; Knöppel et al. 2013]. This integrability condition allows us to compute *simultaneously* a cross field and its parametrization while minimizing a distortion criterion. To achieve this, we rely on two key ingredients. First, we consider vertex-based cross field so that singularities can appear and move *inside* triangles, making the parametrization more flexible than face based frame field integration. Second, we unveil a necessary and sufficient system of equations for seamless parametrization which is independent of the cross field symmetries. This is done by using the so-called *parallel transport* introduced in previous work [Crane et al. 2010; Knöppel et al. 2013]. This integrability condition allows us to compute *simultaneously* a cross field and its parametrization while minimizing a distortion criterion.

Our results demonstrate that cone positions depend on the distortion energy and that we have lower average distortion than previous work including directly integrating a smooth frame field (see Fig. 1).

Algorithm. The overall algorithm takes as input a triangle mesh along with boundary and feature constraints and follows the steps:

(1) Cut the input triangle mesh into disconnected charts centered at vertices (Sec. 3.2);
(2) Flatten the boundary to fit the prescribed boundary and feature edge angles (Sec. 4.2);
(3) Solve a non-linear least-squares optimization problem to obtain optimal charts, parallel transport and cross field (Sec. 5.5);
(4) Reconstruct the final parametrization by inserting additional vertices at singular points (Sec. 3.3);
(5) Reposition singular points in the initial mesh (Sec. 5.6).

The final results are a valid seamless parametrization and a vertex-based cross field with consistent cone points.

1.1 Related Work

As our goal is to build a "single step" seamless parametrization algorithm, it is interesting to consider our work as a specific instance of the broader problem of computing distortion minimizing injective maps [Hormann et al. 2008]. Given a target topology (disk [Lévy et al. 2002; Tutte 1963], orbifold [Aigerman and Lipman 2015], sphere [Kazhdan et al. 2012] or cone metric [Sawhney and Crane 2018; Springborn et al. 2008]), standard distortion minimization finds an optimal geometry in parameter space. In our setting, we aim to find both the geometry *and* the optimal topology minimizing our distortion criterion. Since this problem is particularly challenging, the most common approach is to first solve for the topology independently of the geometry using a practical proxy such as cross fields or cone placement. Only a few work attempt to directly solve for seamless mappings.

Our method falls into the category of direct seamless mapping with a cross field proxy but proceeds in a single optimization step.

Cross field based parametrizations. A cross field is an extremely useful tool for seamless parametrization: it has the same symmetry as a square and thus fixes the element orientation. Direction fields generation is generally well understood for surface meshes. A complete review of the literature is out-of-scope of this paper and we refer to the relevant surveys [de Goes et al. 2016; Vaxman et al. 2016]. In this paper, we will represent crosses by the 4^{th} root of a complex number [Palacios and Zhang 2007; Ray et al. 2008]. Most recent results guarantee that the smoothest cross field can be computed using a diffusion/renormalization scheme [Viertel and Osting 2019]. However, as shown by our experiments, the smoothest cross field topology does not necessarily minimize the parametrization distortion.

More precisely, a cross field is the rotation part of the QR decomposition of the Jacobian of the parametrization [Panozzo et al. 2014], so computing a parametrization implies recovering the missing part of the Jacobian. The simplest method is to find the functions whose gradient is closest to the field directions [Kälberer et al. 2007]. This can also be done using global periodic functions [Fang et al. 2018; Ray et al. 2006] with the advantage of constraining singularities to integer coordinates. A more stable approach traces the motorcycle graph of the field and then solves for the graph edge lengths [Myles et al. 2014].

Fu *et al.* [2016] take an ingenious approach: they divide the input mesh into a set of independent triangles and constrain adjacent simplices to be equal up to the cross field rotation. This simple constraint allows them to compute seamless global parametrization for triangle-based frame field. In this paper, we use a similar formulation but adapted to cross field defined at vertices so that singularities are not bound to appear at vertices.

While convenient, this proxy has one major drawback: not all frame fields can generate a valid parametrization. In particular, frame fields can exhibit limit cycles creating degenerated triangles in parameter space. Many solutions have been proposed to circumvent this issue. Limit cycles can be detected and fixed locally by adding a singularity dipole [Myles et al. 2014]. However, this solution stays local and does not take into account the global problem of minimizing the distortion. Diamanti *et al.* [2015] promote the frame field integrability by constraining it to be the Jacobian of a parametrization, but the distortion minimization has a weak impact on the solution. Therefore, it seems natural to compute the frame field and its parametrization at the same time, so that the frame field constrains the map to be seamless and the map forces the field to be integrable.

Cone metric deformation. Overall, a cross field is a convenient way of placing *quantized* singularities necessary for quad-meshing, however, it is not the only way to solve this problem. Most alternatives

rely on conformal deformations as they offer a simple relationship between area distortion and cone positions. Greedy methods place singularities at places of highest distortion [Ben-Chen et al. 2008], sometimes using a diffusion process [Vintescu et al. 2017]. Myles and Zorin [2012] compute seamless parametrizations by incrementally flattening regions of smallest Gaussian curvature. Its follow-up [Myles and Zorin 2013] enables feature alignment. However, these methods only use an approximation of the distortion during optimization and the actual parametrization is only computed in a post-processing step. Again, the link between distortion and cone placement is weakened. Only Springborn *et al.* [Springborn et al. 2008] actually uses the real mapping distortion but is, again, incremental. Cone placement with these greedy algorithms is very local and the global impact of singularities is never considered, therefore leading to suboptimal solutions. This is especially true for seamless parametrizations as ±1/4-index singularities are obtained by a rounding procedure *after* finding their positions.

Soliman *et al.* [2018] is one of the only global approach available. It also relies on conformal deformation and is very efficient for minimizing area distortion. However, *quantized* cone points are not considered.

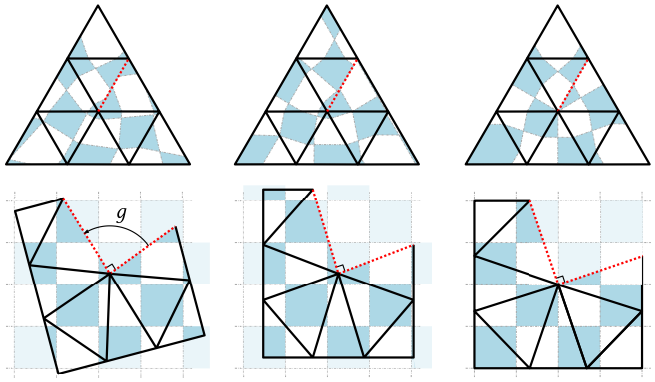
Moreover, these methods suffer from the same problem as frame fields: not all sets of cones admit a parametrization. For surfaces without boundary, prescribed cones satisfying Gauss-Bonnet are always feasible [Campen et al. 2018; Levi 2021b]. However, the existence of boundaries or feature curves can lead to situations where quantized cones and Gauss-Bonnet theorem are no longer sufficient conditions for valid seamless parametrization.

It appears to us that the only way to be certain that a singularity graph is a consistent with a seamless mapping is to *compute* the underlying parametrization. Thus, in this work, we attempt to simultaneously optimize the topology and the parametrization. Besides, previous work heavily rely on conformal maps which do not generalize well in higher dimension. Our method opens the door to a generalization for seamless parametrization of volumes.

Direct seamless parametrization. Very few work have tried to directly compute seamless maps while minimizing the distortion. To the best of our knowledge only Levi [2021a] achieves the goal. The author proposes to compute distortion minimizing seamless maps by directly using its definition as a constraint in an integer programming solver. An initial parametrization is computed by laying out triangles along a spanning tree. The seamless parametrization is then obtained by greedily rounding unvisited edge rotations and recomputing the triangle layout. The immediate advantage is that any distortion energy can be used. However, this approach is greedy and it is impossible to undo previously made decision. Moreover, the initial parametrization influences the singularity distribution, making it unsuitable for subsequent quad-remeshing. Instead, we propose a set of constraints that can be solved by any smooth optimization solver with no integer variables involved, for surfaces with and without boundaries, resulting in reduced stretch distortion and well distributed singularities.

1.2 Overview

Our main contributions:



(a) Seamless (b) Adapted + Seamless (c) Integer

Fig. 2. Three parametrization properties used in this paper illustrated by texture mapping (top) and uv-coordinates (bottom).

- We compute a parametrization of a triangular mesh from any *edge-based* parallel transport rotation, allowing singularities to appear anywhere inside triangles;
- We derive a set of *necessary and sufficient* conditions for seamless parametrization;
- We propose an algorithm optimizing, *at the same time*, a cross field and its parametrization so that singularities can move in order to reduce the overall parametrization distortion.

Organization. The paper is organized as follow. We first give a formal definitions and properties of seamless maps (Sec. 2.1). We then continue, in Sec. 3, by exposing the key construction of this paper: a constraint for a parametrization to be compatible with a parallel transport. In Sec. 4, we further constrain the parametrization to be seamless and adapted to the boundary, leading to a global optimization problem solved in Sec. 5. The results are evaluated and compared with previous methods in Sec. 6.

2 GLOBAL PARAMETRIZATIONS

In this section, we give a formal definition to three key properties of parametrization mappings used throughout the paper: seamless parametrization, feature adaptation and integer mapping. These properties are illustrated in Fig. 2.

Let $\mathcal{M} = (V, E, T)$ be a triangle mesh, consisting of vertices, edges and triangles. A *piecewise linear parametrization* is a map $f : \mathcal{M} \subset \mathbb{R}^3 \to \mathbb{C}$ assigning, to every point $x \in \mathbb{R}^3$ inside a triangle $t \in T$, a coordinate in the (complex) plane $f_t(x) = (u, v) \in \mathbb{C}$. Such a parametrization can exhibit *cuts* along some edges, namely adjacent triangles in the initial mesh are no longer adjacent in the parametrization. The linear functions g relating an edge on one side of the cut to its duplicated version on the other side of the cut are called *transition functions* (see Fig. 2a). Given a closed loop of triangles $(t_0, t_1, \ldots, t_p, t_0)$, all incident to the same vertex $i \in V$, the vertex is said *singular* if the accumulated transition function $g_i = g_{t_0 t_p} \circ \ldots \circ g_{t_2 t_1} \circ g_{t_1 t_0}$ is not identity.

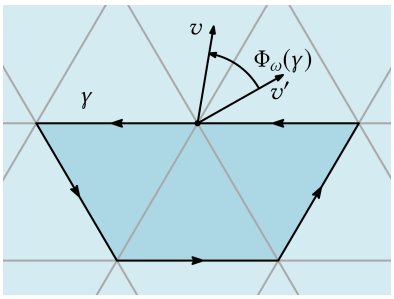


Fig. 3. Every parallel transport has an associated holonomy. In the discrete case, the holonomy of a loop is just the "rotation defect" of a vector transported around the loop. When the loop enclose a single triangle, the holonomy angle is the discrete Gaussian curvature.

We say that a parametrization is seamless if duplicated edge vectors are equal up to a $k\pi/2$ rotation for any two adjacent triangles (Fig. 2a). Namely, the transition functions are square preserving.

Definition 2.1. (Seamless) We call a parametrization seamless if for two adjacent triangles $t_1, t_2 \in T$ sharing an edge $ij \in E$, their edge vectors are related by the transition function:

$$f_{t_1}(p_i) - f_{t_1}(p_j) = e^{\imath k\pi/2}\left(f_{t_2}(p_i) - f_{t_2}(p_j)\right), \quad k \in \{0,1,2,3\}.$$

Another important property is that the surface boundaries and feature edges are isolines of the parametrization (Fig. 2b).

Definition 2.2. (Adapted) A parametrization is adapted to its boundary if all feature edges and boundary edges $ij \in E$ of a boundary triangle $t \in T$ have one zero coordinate in parameter space:

$$\langle f_t(p_i) - f_t(p_j), e_k\rangle = 0, \quad k \in \{1,2\},$$

where $e_1 = 1 + \imath 0, e_2 = 0 + \imath$ are the plane axes.

In order to be able to extract quads from a parameterization, we need boundary vertices to have one integer coordinates and singular vertices have two integer coordinates [Lyon et al. 2016], as shown in Fig. 2c. In this paper, we do not aim for integer maps and limit ourselves to seamless and boundary adapted parametrizations. The quantization step is seen as an independent problem possibly, solved by other methods like [Bommes et al. 2013; Campen et al. 2015].

Def. 2.1 is challenging to use in an optimization process: if the cuts are not given, moving a singularity requires to change the transition functions of multiple triangles. Instead we propose to compute seamless map by "spreading" the transition rotations over the entire mesh, namely integrating the *parallel transport* of a cross field.

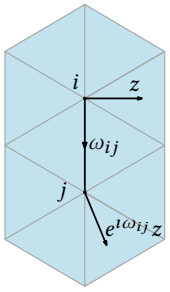
3 PARAMETRIZATION FROM A PARALLEL TRANSPORT

In this section, we prove a necessary and sufficient system of equations (Eqs. ($\mathcal{E}$)) for a parametrization to have the same holonomy as a given parallel transport – an assignment of angle per oriented edge. This system of constraints is applied to a collection of vertex charts $\mathcal{M}^c$ (see Fig. 5b), emanating from the intersection of the primal and the dual mesh of $\mathcal{M}$. Our output parametrization may have a different connectivity than the input mesh, since singularities may appear inside triangles. Therefore, we do not parametrize $\mathcal{M}$, but the triangle mesh $\mathcal{M}_p$ where singular triangles are subdivided by inserting a vertex as represented in Fig. 5c.

This parametrization process is summarized in Fig. 4.

3.1 Discrete Parallel Transport

We consider tangent vectors defined at vertices. As in [Knöppel et al. 2013], tangent planes are mapped to the complex plane, so a tangent vector is a complex number in a local coordinate system. To compare vectors at neighboring vertices ij, they must be rotated by an angle ω_{ij} to take into account the change of local coordinates. Transporting a vector from tangent space to tangent space along an edge path is called the *discrete parallel transport*.



A parallel transport is nothing more than the assignment of an angle ω per oriented edge. Therefore, a vector $v_i \in \mathbb{C}$ at vertex i is parallel transported to vertex j by the rotation: $e^{\omega_{ij}} v_i \in T_j\mathcal{M}$. Note that the transport from j to i is given by the inverse rotation thus: $\omega_{ji} = -\omega_{ij}$.

The integrability constraints built in this section requires only an assignment of a rotation angle per edge. Thus, we do not need to know how the local bases are constructed and this discussion is delayed to Sec. 4.1.

Holonomy. The parallel transport is deeply related to the notion of curvature. Indeed, when transporting a vector around a closed loop, it will in general not return to its original orientation. The rotation between the original vector and the transported one is the called the *holonomy* [Crane et al. 2010] and is illustrated in Fig. 3. Given a vertex closed loop $\gamma = (i_0, i_1, \ldots, i_N, i_0)$, the holonomy Φ_ω around γ is simply the composition of all rotations along the path:

$$\Phi_\omega(\gamma) := \exp\left(\imath\omega_{i_N i_0}\right) \prod_{k=0}^{N-1} \exp\left(\imath\omega_{i_k i_{k+1}}\right).$$

When the loop is reduced to a triangle $ijk \in T$, the discrete Gaussian curvature K is defined as the angle of the holonomy rotation [2013]:

$$\exp(\imath K_{ijk}) = \exp(\imath\omega_{ki})\exp(\imath\omega_{jk})\exp(\imath\omega_{ij}). \tag{1}$$

Of course, a valid parallel transport must satisfy the Gauss-Bonnet theorem:

$$\sum_{t\in F} K_t = 2\pi\chi,$$

where χ is the Euler characteristic of $\mathcal{M}$. If this is not the case, the topology of the parallel transport does not match the one of the triangle mesh, so a parametrization cannot be defined and our system of equations ($\mathcal{E}$) does not have a solution. In Sec. 4 we will show how to build a parallel transport compatible with a cross field following the steps of Knoppel *et al.* [2013]

Our goal is to find a parametrization embedding whose transition functions match a given parallel transport holonomy. Namely, a triangle with zero Gaussian curvature will be parametrized by a flat

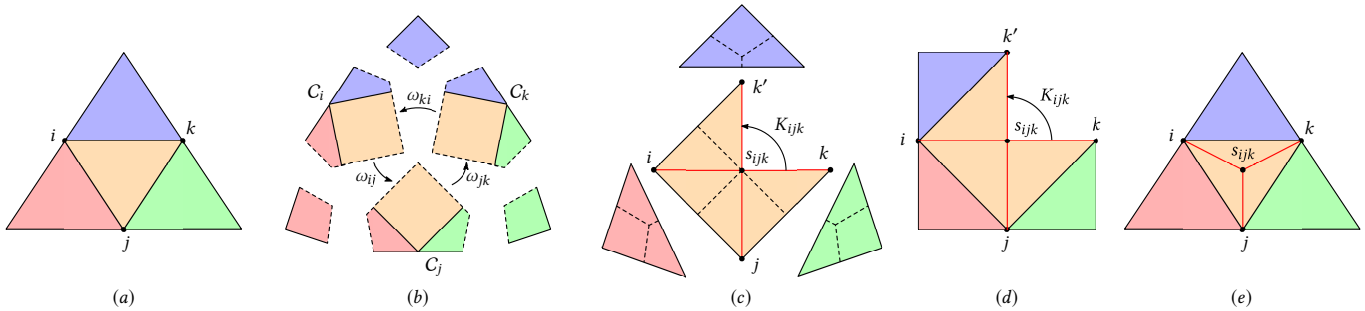


Fig. 4. Parametrization from a parallel transport ω. A triangle mesh (a) is divided into a set of vertex charts (b) satisfying the integrability conditions of Eqs. ($\mathcal{E}$), namely duplicated edges are equal up to the cross field rotation ω. The triangles are assembled independently (c). The triangle ijk is singular so the vertex s_{ijk} is added along with new edges in red. The parametrization (d), obtained by gluing triangles to each other, corresponds to the mesh (e) where the singular point is inserted in triangle ijk.

triangle, whereas a singular triangle (non trivial curvature) must be cut open by the parametrization with an angle matching the Gaussian curvature.

3.2 Parametrization Constraints

Vertex charts and chart collection. In order to parametrize an edge based parallel transport, we must define charts based at vertices. A chart C_i is a small quad-mesh around vertex i living in parameter space, in our case the complex plane. Charts are disconnected from each other. We define the chart collection $\mathcal{M}^c = \cup_{i \in V} C_i$ as the set of all charts emanating from the subdivision of the input mesh.

As singularities appear inside triangles, the charts must include a point inside each triangle where a singularity may appear. For this reason, we build our charts as the intersection of mesh $\mathcal{M}$ with its dual. Namely, each triangle is split into three quads, as in Fig. 5b, by adding three edges linking the triangle center to mid-edge points. We explicitly build the chart C_i at vertex i as the union of quads containing i. In this construction, adjacent charts are disjoint but share two types of duplicated edges: the primal edges from the input mesh and the dual edges linking edge center to triangle center. They are highlighted in red in Fig. 5b.

To ease the notation, we use specific letters for each point type. The letter p denotes the vertex coordinates (with a complex number) of the input mesh in parameter space, the letter e is the midpoint of an edge and s the center of the triangle and thus a potential singularity. Moreover, underscripts are reserved to make explicit the membership of a vector to a chart. For example, $s_i^{jk} - e_i^j$ is the vector connecting the edge midpoint to a triangle center in the chart C_i whereas $s_j^{ki} - e_j^i$ is the duplicated edge in the chart C_j.

Parallel Transport Constraints. By definition, a parallel transport prescribes relative orientation of tangent planes. For a chart to be consistent with the parallel transport, the transfer of a tangent vector in C_i to an adjacent chart C_j should undergo a rotation of angle ω_{ij}. Any edges shared by two adjacent charts are tangent vectors in two different bases and, therefore, must be equal up to a rotation by the parallel transport. Overall, the charts must satisfy

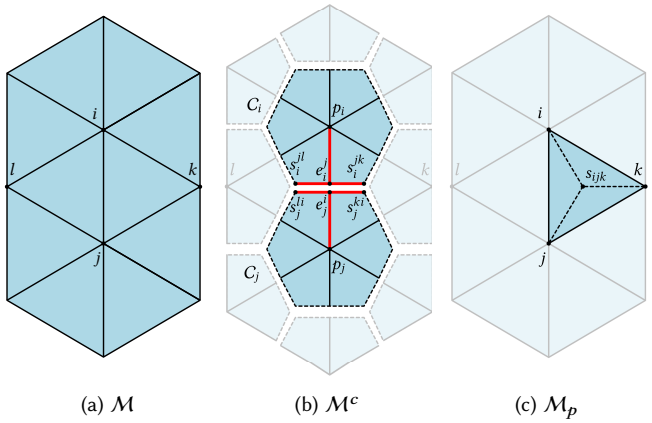


Fig. 5. The initial mesh (a) is subdivided in a chart collection (b) used for imposing integrability constraints. Edge midpoint are denoted with letter e and triangle center point with the letter s. We do not parametrize the initial mesh but a subdivision of it where singularities s are inserted inside singular triangles (c).

the integrability constraints:

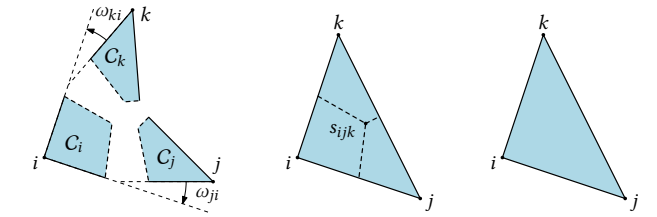
$$\left| \begin{array}{lcll} e_j^i - p_j & = - & \exp\left(\imath\omega_{ij}\right)\left(e_i^j - p_i\right), & \forall ij \in E \\ s_j^{ki} - e_j^i & = & \exp\left(\imath\omega_{ij}\right)\left(s_i^{jk} - e_i^j\right), & \forall ijk \in F \end{array} \right. \quad (\mathcal{E})$$

Note that Eqs. ($\mathcal{E}$) are written only in term of the chart edges. So, without loss of generality, we can reduce the number of variables by fixing the chart translation and setting $p_i = 0$, $\forall i \in V$.

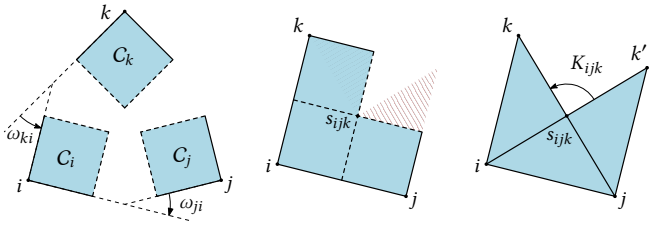
3.3 Parametrization Reconstruction

In this section, we assume that the charts satisfy the system of equations in Eqs. ($\mathcal{E}$). We will show how to recover a parametrization from the chart collection $\mathcal{M}^c$.

Triangle Parametrization. The charts have coordinates in parameter space, so to recover the parametrization of a triangle ijk we simply have to reassemble the three pieces separated into charts C_i, C_j and C_k. This is always possible because Eqs. ($\mathcal{E}$) ensure that



(a) Regular triangle



(b) Singular triangle

Fig. 6. Reconstruction of triangle parametrization from vertex charts. Left: independent charts related by a parallel transport. Middle: synchronization of C_j and C_k with C_i. Right: final triangle parametrization. Singular triangles are remeshed (b).

duplicated edges have same length. Thus, the triangle piece in C_j can be stitched to the piece in C_i by applying the rigid transformation $x \mapsto \exp(\imath\omega_{ji})(x - e_j^i) + e_i^j$. Synchronizing C_j and C_k with C_i leads to a triangle parametrization.

We distinguish two cases: (1) the synchronization creates a valid triangle and the final parametrization is obtained by removing dual edges (Fig. 6a), (2) a cut appears so we triangulate the synchronization by adding edges linking the singular point s and triangle vertices (Fig. 6b).

Mesh Parametrization. Given the parametrization of each triangle, it is easy to recover the parametrization of the entire mesh $\mathcal{M}_p$ by gluing adjacent triangles along a dual spanning tree. The overall algorithm is summarized in Alg. 1 and by Fig. 4.

Crucially, we can prove that if a chart collection satisfies constraints Eqs. ($\mathcal{E}$), then the parametrization transition functions are determined by the parallel transport holonomy. In particular, a regular triangle (*i.e.* $K_{ijk} = 0$) is mapped to a regular triangle in the parametrization and singular triangle admits a singularity at point s_{ijk} with an angle defect equal to K_{ijk}. Interestingly, only the parallel transport holonomy matters: changing ω while keeping the holonomy constant does not change the set of solutions of Eqs. ($\mathcal{E}$).

Theorem 3.1. *Given a parallel transport ω, if the vertex charts satisfy Eqs. ($\mathcal{E}$) then the transition functions of the parametrization constructed with Alg. 1 are equal to the parallel transport holonomy.*

Conversely, given any parametrization whose singular vertices are of valence three, there exists a parallel transport ω and vertex charts satisfying Eqs. ($\mathcal{E}$).

In particular, the parametrization is seamless if and only if the holonomy of ω around any loops is a rotation by an integer multiple of $\pi/2$.

ALGORITHM 1: ParametrizationReconstruction

Input: A parallel transport ω and a chart collection $\mathcal{M}^c$ satisfying Eqs. $\mathcal{E}$
Output: A mesh $\mathcal{M}_p$ and its parametrization
$\mathcal{M}_p \leftarrow \mathcal{M}$
forall $ijk \in T$ **do**
  Stitch C_j and C_k with C_i.
  if $K_{ijk} = 0$ **then**
    The parametrization of ijk is a triangle: remove dual edges.
  else
    Triangulate the charts.
    Insert a vertex inside triangle ijk in $\mathcal{M}_p$.
  end
end
Add an arbitrary initial triangle to a queue.
while *Queue not empty* **do**
  Find the translation/rotation matching the edge shared with previous triangle.
  Add adjacent triangles to queue.
end

An immediate corollary to Thm. 3.1 is that a parametrization is seamless if and only if the parallel transport is compatible with an *integrable* cross field.

The parallel transport curvature is not explicitly present in Eqs. ($\mathcal{F}$), so that the singularity locations are needed *only* during the reconstruction. This is the great benefit of our approach, allowing us to change the curvature while optimizing for the parametrization.

4 SEAMLESSNESS AND BOUNDARY ALIGNMENT

So far, we have found conditions to define a parametrization from a parallel transport ω. However, we do not have guarantees that this parametrization is seamless, adapted to the boundary or locally injective. In this section, we present three additional necessary and sufficient set of constraints to satisfy these properties.

4.1 Seamlessness Constraints

According to Thm. 3.1, a parametrization is seamless if it is compatible with a parallel transport whose holonomy is square preserving. This property is deeply related to cross fields.

Discrete Cross Fields. A cross field assigns at each vertex a set of two orthogonal directions:

$$\left\{ e^{\imath k\pi/2} v_i, \quad k = 0, 1, 2, 3 \right\}.$$

A key insight is that, when raised to the 4^{th} power, this set reduces to a single complex value:

$$z_i := v_i^4$$

Thus, a cross can be uniquely represented by a unit complex number whose fourth roots gives the four vectors of the cross [Palacios and Zhang 2007; Ray et al. 2008].

Since $v_i \in \mathbb{C}$ is a tangent vector, it can be transported to an edge adjacent vertex by a parallel transport. Conversely, a vector field defines a parallel transport as the rotation relating any two adjacent vectors. The same can be said of an arbitrary cross field. When

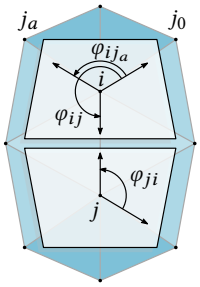
represented by the fourth root of a unit vector, it defines a parallel transport ω as:

$$z_j = \exp(4\imath\omega_{ij}) z_i. \tag{2}$$

This parallel transport, associated to a cross field, has the distinctive property of preserving the cross symmetry. In particular, it means that its holonomy around any loop are rotations by an angle of $k\pi/2$.

As the cross field lives in the complex tangent space, Eq. 2 is purely combinatorial in the sense that the geometry of $\mathcal{M}$ is not necessary to build it. However, for visualization purposes and for the stability of our optimization scheme, it is important to build local bases and give the cross field a geometrical meaning.

Parallel transport from Levi-Civita connection. The construction of a vertex-based frame field follows the one by Knöppel *et al.* [Knöppel et al. 2013]. A tangent vector $v_i \in \mathbb{C}$ is expressed in the local basis of the vertex i. This local basis defines a *tangent space* at this vertex. Vertices on a triangulated surface are generally not flat as the inner angles θ_i^{jk} of triangles incident to i do not sum to 2π. A local coordinate system is constructed by intrinsically "flattening" each vertex, namely inner angles are normalized:



$$\tilde{\theta}_i^{jk} := 2\pi\theta_i^{jk}/\Theta_i$$

where $\Theta_i = \sum_{ijk} \theta_i^{jk}$ is the total inner angle at vertex i. At each vertex, we assign a reference edge ij_0 whose angle coordinate φ_{ij_0} is by definition zero. The angles of other ordered edges $ij_0, \ldots, ij_n$ are obtained by accumulating modified inner angles:

$$\varphi_{ij_a} := \sum_{p=0}^{a-1} \tilde{\theta}_i^{j_p j_{p+1}}. \tag{3}$$

To compare adjacent vectors, we define the parallel transport $\rho : E \to \mathbb{R}$ as the rotation angle aligning the basis at j to the one at i:

$$\rho_{ij} := \varphi_{ji} - \varphi_{ij} + \pi,$$

by comparing the angles of the shared edge ij. The change of basis is then $r_{ij} = \exp(\imath\rho_{ij})$. By construction, this parallel transport is associated to the Levi-Civita connection. As shown by Knöppel *et al.* [Knöppel et al. 2013], its Gaussian curvature, as defined in Eq. (1) satisfies the Gauss-Bonnet theorem (for meshes without boundaries).

Therefore, any parallel transport can be decomposed into the sum of the cross field rotation α and the Levi-Civita parallel transport ρ:

$$\left| \; z_j = \exp(4\imath\alpha_{ij}) r_{ij}^4 z_i. \right. \tag{$\mathcal{F}$}$$

In our optimization scheme we use the cross field rotation as a variable as it can be easily bounded and initialized.

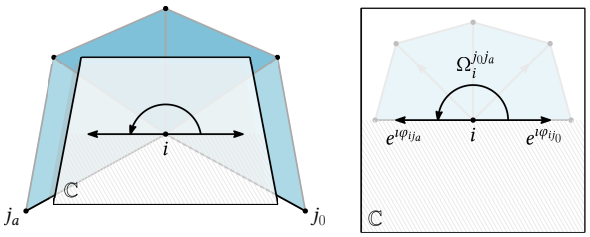


Fig. 7. Boundary edges, depicted in black, are laid out in the tangent complex plane so that they match the provided corner angle $\Omega_i^{j_0 j_a}$.

4.2 Boundary and feature edge constraints

Another requirement we discussed in Section 2 is that the parametrization should be *adapted* to the mesh boundary and features edges, so additional constraints are required on the cross field and on the vertex charts. We process feature edges extracted from a boundary or user specified in the same manner.

Single vertex corner. We define a vertex corner as a set of adjacent triangles incident to the same vertex i and delimited by two feature edges ij_0 and ij_a. In this paragraph, we consider a single vertex corner. We assume that target corner angles $\Omega_i^{j_0 j_a}$ integer multiple of $\pi/2$ are provided as input. At a corner ij_0j_a, inner angles are normalized to match the prescribed angle:

$$\tilde{\theta}_i^{j_p j_{p+1}} := \Omega_i^{j_0 j_a} \theta_i^{j_p j_{p+1}} / \Theta_i^{j_0 j_a}$$

where $\Theta_i^{j_0 j_a}$ is the total inner angle between edges ij_0 and ij_a. The edge angles in the tangent plane are obtained by accumulating the modified inner angles as in Eq. (3). The representation vector z_i is constrained to be equal to $e^{4\imath\varphi_{ij_0}}$, by construction the vectors of this cross agrees with the feature edge directions. An illustration of this procedure is given in Fig. 7.

The corresponding vertex corner in chart C_i must also be constrained to the corner angle. To do so, the primal edge vector corresponding to a feature edge must remain orthogonal to the feature edge normal. Therefore, at a vertex corner, we enforce three constraints linear with respect to the chart coordinates and the cross field:

$$\left| \begin{array}{l} z_i = \exp(4\imath\varphi_{ij_0}), \\ \langle p_i - e_i^{j_0}, \imath \exp(\imath\varphi_{ij_0}) \rangle = 0, \\ \langle p_i - e_i^{j_a}, \imath \exp(\imath\varphi_{ij_a}) \rangle = 0. \end{array} \right. \tag{$\mathcal{B}$}$$

Note that since the cross field is constrained on feature vertices, the rotation α is also known along feature edges.

Singular vertices. Let us consider the case where an *interior* vertex i is incident to multiple feature edges $(ij_{a_0}, ij_{a_1}, \ldots, ij_{a_n})$ forming $n+1$ corners. If the sum of all corner angles $\Omega_i = \sum_{p=0}^{n-1} \Omega_i^{j_{a_p} j_{a_{p+1}}} + \Omega_i^{j_{a_n} j_{a_0}}$ is a multiple of 2π, then the flattening of each corner creates a valid vertex chart. However, when $\Omega_i \neq 2k\pi$, vertex i is *singular* and an additional seam is necessary to lay out edges in the tangent plane. Thus, we cut open the vertex neighborhood at edge ij_{a_0} introducing a duplicated edge ij'_{a_0}. For the parametrization of chart C_i to remain seamless, edges on each side of the cut must have equal

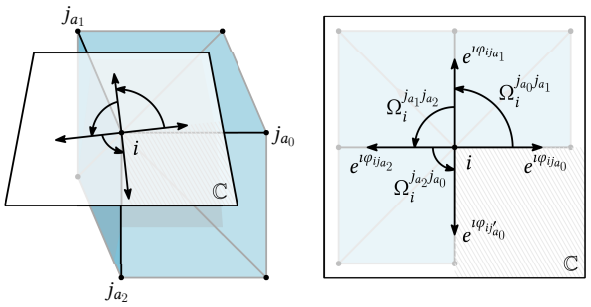


Fig. 8. Three feature edges, depicted in black, are laid out in the tangent complex plane. A seams is introduced at edge ija_0 because the sum of the corner angles Ω is not an integer multiple of 2π.

length yielding an additional linear constraint:

$$\left| \quad \langle p_i - e_i^{j_{a_0}}, \exp(\iota\varphi_{ij_{a_0}})\rangle = \langle p_i - e_i^{j'_{a_0}}, \exp(\iota\varphi_{ij'_{a_0}})\rangle. \qquad (\mathcal{B})$$

Fig. 8 illustrates the computation of the tangent plane at a cube corner.

4.3 Local injectivity

We need to ee that the final parametrization is locally injective, in other words triangle areas must remain positive in parameter space. This boils down to two different constraints. First, the orientation of primal edges in vertex charts should be preserved:

$$\left| \quad \det(p_i - e_i^j, p_i - e_i^k) \quad > 0. \qquad (\mathcal{D}_1)$$

Second, the singularity point s_{ijk} should stay inside triangle ijk:

$$\left| \begin{array}{ll} \det(p_i - e_i^j, p_i - s_i^{jk}) & > 0, \\ \det(p_i - s_i^{jk}, p_i - e_i^k) & > 0. \end{array} \right. \qquad (\mathcal{D}_2)$$

4.4 Theoretical guarantees

The constraints introduces in Sections 3, 4.1, 4.2 and 4.3 define a system of equations whose solution is a boundary adapted seamless parametrization. Most importantly all of these constraints are independent of the cross field symmetries and do not rely on integer variables. Therefore they are readily usable in any standard continuous solver.

Theorem 4.1. *If a set of charts, a cross field and a parallel transport satisfy the system of Eqs. $(\mathcal{E})$, $(\mathcal{F})$, $(\mathcal{B})$, $(\mathcal{D}_1)$ and $(\mathcal{D}_2)$, then we can recover a locally injective, boundary adapted seamless parametrization. Moreover, the singular vertices appear only in singular faces with the angle defect prescribed by the parallel transport curvature.*

5 DISTORTION MINIMIZING MAP

So far, we have defined a system of necessary and sufficient equations for computing valid global seamless parametrizations. In this section, we describe our optimization process in order to find vertex charts, cross field rotations α and representation vector z solutions to this system.

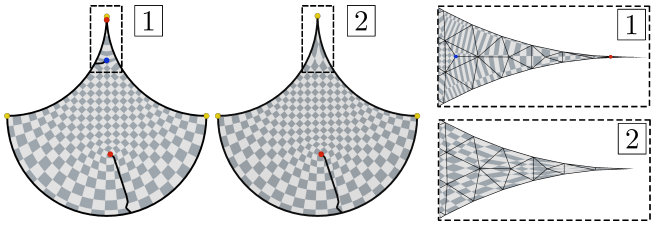


Fig. 9. Influence of the initialization of the cross field rotations on a sharp corner parametrization. (1) Initialization with zeros rotation. (2) Initialization with modified parallel transport prevents a dipole of singularities to appear. Singularities $\frac{\pi}{2}$ are depicted in red, $-\frac{\pi}{2}$ and boundary corners in yellow.

5.1 Simplifying constraints using the Cayley transform

One great challenge when numerically solving Eqs. $(\mathcal{E})$ and $(\mathcal{F})$ is the presence of a periodic complex exponential. Luckily, there exists a clever way to parametrize the space of rotations by non-periodic polynomial functions, thus considerably reducing the complexity of the problem. The *Cayley map* [Kobilarov et al. 2009; Zhang et al. 2021], noted cay, is a complex fraction equal to the rotation by an angle $\tan(\alpha/2)$:

$$\mathrm{cay}(\alpha) := \frac{1 - \iota\alpha}{1 + \iota\alpha} = \exp\left(\iota \tan\frac{\alpha}{2}\right)$$

Instead of using the cross field rotation angle α_{ij}, we will consider its tangent half angle. So, using the change of variables $\overline{\alpha}_{ij} = \tan(\alpha_{ij}/2)$, Eqs. $(\mathcal{E})$ and $(\mathcal{F})$ can be rewritten as polynomial equations:

$$\left| \begin{array}{lll} (1 + \iota\overline{\alpha}_{ij})(e_i^j - p_i) & = - & (1 - \iota\overline{\alpha}_{ij}) r_{ij} (e_j^i - p_j) \\ (1 + \iota\overline{\alpha}_{ij})(e_i^j - s_i^{jk}) & = & (1 - \iota\overline{\alpha}_{ij}) r_{ij} (e_j^i - s_j^{ki}) \end{array} \right. \qquad (\bar{\mathcal{E}})$$

$$\left| \quad (1 + \iota\overline{\alpha}_{ij})^4 z_i = (1 - \iota\overline{\alpha}_{ij})^4 r_{ij}^4 z_j \qquad (\bar{\mathcal{F}}) \right.$$

5.2 Log-barrier

The inequality constraints of Eqs. $\mathcal{D}_1$ and $\mathcal{D}_2$ are enforced using the continuously differentiable barrier function $B_\eta : \mathbb{R}^{>0} \to \mathbb{R}$:

$$B_\eta(x) = \begin{cases} +\infty & x \le 0, \\ \log(\frac{x}{\eta}) & 0 < x \le \eta, \\ 0 & \eta \le x. \end{cases}$$

Given a determinant d whose initial value is d_0, we set the threshold to $\eta = d_0/2$.

5.3 Optimization

We would like to find the chart vertex coordinates (x, y), the frame rotation $\overline{\alpha}$ and the cross field z solution of the constrained optimization problem:

$$\begin{array}{ll} \min\limits_{\overline{\alpha}, z, x, y} & E(x, y) \\ \text{s.t.} & (\bar{\mathcal{E}}), (\bar{\mathcal{F}}), (\mathcal{B}), (\mathcal{D}_1), (\mathcal{D}_2), \end{array} \qquad (4)$$

where $E : \mathcal{M}^c \to \mathbb{R}$ is any given distortion energy.

To make the problem tractable, we enforce non-linear constraints as a sum of squared energy:

$$R(\overline{\alpha}, z, x, y) = \lambda_E \|(\bar{\mathcal{E}})\|^2 + \lambda_F \|(\bar{\mathcal{F}})\|^2 + \lambda_{\det} \|B_\eta(\mathcal{D}_1)\|^2 + \lambda_{\det} \|B_\eta(\mathcal{D}_2)\|^2,$$

leading to a non-linear least-squares optimization with linear constraints:

$$\min_{\overline{\alpha},z,x,y} \varepsilon E(x,y) + R(\overline{\alpha},z,x,y) \qquad \text{s.t. } (\mathcal{B}). \tag{5}$$

The Levenberg-Marquardt algorithm is tailored for minimizing non-linear least squares with zero being the global minimum. We follow the implementation of [Marumo et al. 2020] whose evolution of the damping parameter ensures global convergence toward a local minimum and a second order rate of convergence near a zero of the objective function. Each step involves solving a quadratic optimization problem with linear constraints. For this, we use the open source solver OSQP [Stellato et al. 2020].

The problem in Eq. (5) is solved several time for decreasing values of ε. For our distortion minimizing experiments, we start at $\varepsilon = 10$ and divide by a factor of 10 until $\varepsilon = 10^{-4}$. Unless specified otherwise, we set all weights λ to 1.

Note that for surfaces with boundary or feature edges, we do not need to force z to be unit norm. It suffices that the cross field is constrained somewhere so that whenever Eqs. ($\bar{\mathcal{F}}$) are satisfied, the cross is unitary everywhere. For surfaces without boundary or feature edges, we constrain z at a random vertex to avoid the trivial zero solution.

5.4 Initialization

As for any non-convex optimization, the initialization should be chosen carefully.

The charts are initialized using the tangent plane flattening introduced in Secs. 4.1 and 4.2. Given the normalized triangle inner angles, the chart primal edges are laid out in the plane with their initial edge lengths. The chart is completed by placing singular points s at the triangle barycenters. These initial charts keep the triangle edge lengths unchanged so they can be seen as an *isometric* parametrization of the input triangle mesh.

The frame field is set to zero everywhere except at boundary and feature edges where it is fully constrained (see Sec. 4.2).

For surfaces without boundary or feature edges, the frame rotation α is simply set to zero. However, in some cases this initialization can lead to highly distorted solutions. Typically, for CAD models where feature edges meet at very acute angles, a singularity appears directly inside the sharp triangle. Inspired by [Desobry et al. 2021], we compute an initial frame rotation α by forcing the cross field to be regular on all triangles belonging to 3-ring $T(i)$ of a sharp corner i. In practice, we solve the quadratic problem:

$$\begin{aligned} \min_{\alpha\in\mathbb{R}^{|E|}} \quad & \|\alpha\|^2 \\ \text{s.t.} \quad & \alpha_{ij} + \alpha_{jk} + \alpha_{ki} = -K_{ijk}, \quad \forall ijk \in T(i) \\ & \alpha_{ij} = \alpha^0_{ij}, \quad \forall ij \in \partial E \end{aligned}$$

where α^0 are the frame rotations along feature edges. Fig. 9 illustrates how this initialization drastically changes the parametrization and reduces the distortion.

5.5 Distortion Energy

Among all possible valid seamless parametrizations, we would like to find the one minimizing a distortion criterion. In theory, we could use any distortion energy on the charts. In practice, we mostly focus on quad remeshing application, so we would like to avoid as much as possible shearing *i.e.* quads with non-orthogonal edges. The simplest way to promote these square-like quads is to penalize parametrizations whose parameter gradients are non-orthogonal . Thus, as a distortion measure, we use the LSCM [Lévy et al. 2002] energy sum over all charts:

$$E(x,y) = \sum_{i\in V} \sum_{t\in T_{C_i}} \|\nabla y_t - \mathcal{J}\nabla x_t\|_2^2,$$

where (x,y) are the chart vertex coordinates and $\mathcal{J}$ is the 90° counter-clockwise rotation around the normal. The gradient operator is computed with respect to the initial isometric chart parametrization described in Sec. 5.4. According to Prop. 5.1, this energy is zero if and only if the vertex positions (x,y) are equal to the initial charts coordinates up to a rotation and a global scaling. Thus, this energy penalizes shearing and stretching deformations of each vertex rings while staying quadratic.

Proposition 5.1. *The LSCM energy of a flat triangle mesh vanishes only for rotations and global scaling of the vertex positions.*

Unless specified otherwise our results are obtained with the LSCM energy.

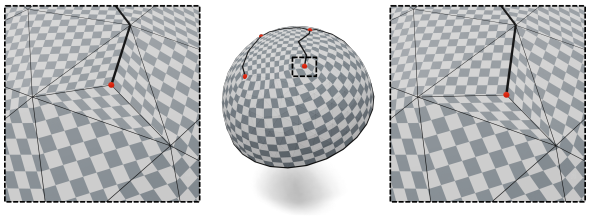


Fig. 10. Replacing the singularity inside the triangle mesh $\mathcal{M}_p$ is achieved by solving an ARAP problem between the split triangle and its representation in (u, v) space. Left: close-up a naive singularity placement: the barycenter of the triangle. Right: ARAP places the singularity in order to minimize local distortion. Middle: overview of the whole half-sphere model.

5.6 Singularity Repositionning

Once the parametrization is reconstructed, each singular triangle is split into three (Fig. 6b). However, the position of the singular point on the surface mesh $\mathcal{M}_p$ is not yet optimally determined. We would like to find the vertex position s_{ijk} on the surface mesh minimizing the distortion of the three incident triangles (t_1, t_2, t_3). To do so, we optimize the as-rigid-as-possible (ARAP) energy:

$$s_{ijk} = \underset{\substack{x\in\mathbb{R}^2 \\ R^\top R = I}}{\operatorname{argmin}} \sum_{i=1}^{3} A_{t_i} \|J_{t_i}(x) - R_{t_i}\|_F^2,$$

where A_t is the triangle area in parameter space and J_t is the Jacobian matrix from uv-coordinates to the surface mesh $\mathcal{M}_p$. We solve this problem alternating minimization in x and in the rotation R, as described by Sorkine *et al.* [Sorkine and Alexa 2007].

Model	[Myles and Zorin 2012]		[Diamanti et al. 2015]		[Levi 2021a]		Fixed Frame Field		Ours	
	Singus	Stretch	Singus	Stretch	Singus	Stretch	Singus	Stretch	Singus	Stretch
Airplane	62	1.197	60	1.339	-	-	44	1.202	40	**1.152**
Bunny	50	1.132	56	1.171	-	-	34	1.135	32	**1.124**
Dilo	82	1.331	140	1.397	204	1.314	72	1.347	70	**1.241**
Dancer	66	1.308	146	1.348	105	1.228	66	1.243	70	**1.120**
Shark	64	**1.199**	187	1.522	105	1.202	66	1.280	64	1.283
Rhino	84	1.138	-	-	-	-	60	1.189	62	**1.113**
Spot	35	1.160	-	-	-	-	40	1.181	42	**1.131**
Head	93	1.184	-	-	-	-	20	1.276	8	**1.085**

Table 1. Comparison of the number of singularity cones and the mean stretch distortion for eight models. Dash indicate missing values.

Model	[Levi 2021a]		Fixed Frame Field		Ours	
	Singus	Stretch	Singus	Stretch	Singus	Stretch
Sculpt	217	1.422	16	1.288	16	**1.206**
Beetle	31	1.505	42	1.358	74	**1.215**
Metatron	188	1.838	112	1.431	80	**1.434**
Fandisk	68	**1.284**	36	1.369	34	1.338
Casting	126	1.512	120	**1.423**	116	1.512
Hilbert	-	-	248	**1.000**	248	**1.000**
M6	-	-	60	**1.183**	52	1.186
S40	-	-	48	1.215	48	**1.184**

Table 2. Comparison of the number of singularity cones and the mean stretch distortion for eight models with feature edges. Dash indicate missing values.

6 EVALUATION AND RESULTS

We evaluate our method on a variety of triangular meshes, both with and without feature edges. Models were taken from databases gathered by [Myles et al. 2014], [Levi 2021a] and the Mambo CAD dataset[1]. All our results are included in supplemental material.

Quantitative results are gathered in Table 1 for model without features and in Table 2 for CAD models. We choose two evaluation criterion: the number of cone singularities and the average *stretch distortion*. The stretch is a scale-invariant measure computed as the ratio σ_1/σ_2 of the largest and the smallest singular value of the Jacobian matrix (best score is 1). Compared with other methods, we produce fewer cones, with comparable or smaller average distortion.

In the following section, we provide more in depth comparison with previous works by considering the *scale distortion*, defined as the determinant of Jacobian. This metric is scaled with respect to the total area so that the optimal value is 1. Moreover, we plot distortion histograms in log scale of the triangle count. When quad meshes are provided, they are obtained by computing an integer grid map using [Bommes et al. 2013] and by then extracting quads with [Ebke et al. 2013]. Most importantly, they have the same singularities (positions and indices) as the input parametrizations. More results and comparisons can be found in the supplemental material.

[1]https://gitlab.com/franck.ledoux/mambo

All our experiments were conducted on a Ubuntu workstation with a height-core, 2.6-GHz Intel Core i5. Our python implementation takes a few minutes on a thousand triangle mesh and up to an hour for $30k$ triangles.

Convergence and local minima. As shown by Thm. 4.1, we have mathematical guarantees that if our constraints are satisfied, then we can extract a global parametrization. However, our optimization problem in Eq. (5) is non-convex and could get stuck in a local minimum. In practice, we never encounter this situation. To our understanding, the objective function can always be locally decreased by either reducing the norm of z or by making the cross field better satisfy Eq. ($\bar{\mathcal{F}}$). Thus, assuming $\varepsilon = 0$, our optimization either converges toward a zero of the objective or tries to reach a global minimum at infinity.

The latter case is typically created by a frame field with a limit cycle whose parametrization is degenerated. Fig. 17a shows an example of such a configuration. Our optimization with standard parameters is not able to escape this limit cycle.

However, we can lower the weight on the cross field constraint to $\lambda_F = 0.1$, so that the edge constraint dominates the energy, and switch to the isometric distortion energy:

$$E(x, y) = \sum_{i \in V} \sum_{t \in T_{C_i}} \|J_t^\top J_t - I\|_2^2, \tag{6}$$

where J_t is the Jacobian of the triangle t. This way our algorithm converges to a global minimum and introduces two new singularities as in Fig. 17b. Thus, our method does deliver its promises: the parametrization and the frame field influence each other in order to create a valid parametrization.

Influence of distortion energies. Fig. 1 illustrates the influence of distortion energies on singularity positions. The LSCM energy (Fig. 1d) is compared with the isometric energy in Eq. (6) (Fig. 1c) and with the shear energy:

$$E(x, y) = \sum_{i \in V} \sum_{t \in T_{C_i}} \|\nabla y_t^\top \mathcal{J} \nabla x_t\|_2^2,$$

penalizing orthogonal but possibly stretch quads (Fig. 1b). As expected the singularity positions depends on the distortion being optimized. For this model, the LSCM energy is best to minimize both stretching and shearing of the parametrization.

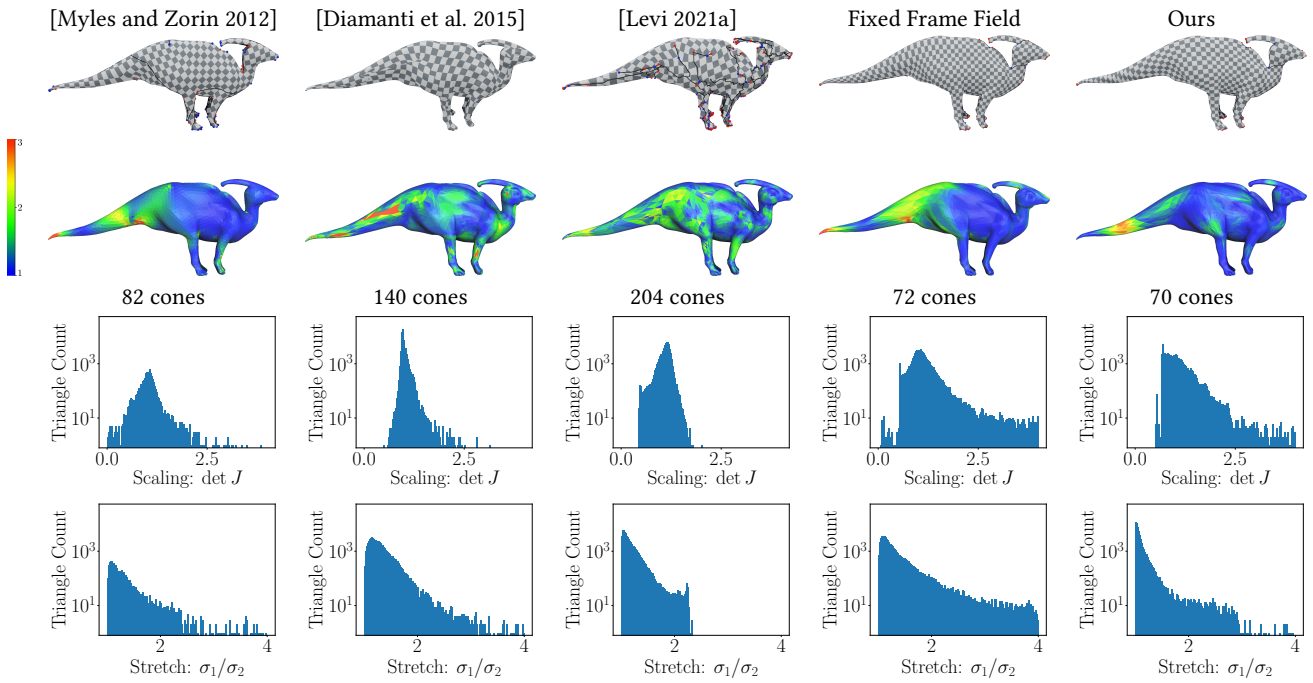


Fig. 11. Comparison of five methods on the dilo model. Top row: texture mapping with highlighted seams and singularities. Second row: stretch distortion distribution over the model. A blue indicates a perfect value of 1 while the red color indicates a value of 3 or higher. Third row: histograms of area distortion. Fourth row: stretch distortion histograms. Our method minimize stretching at the cost of more area distortion.

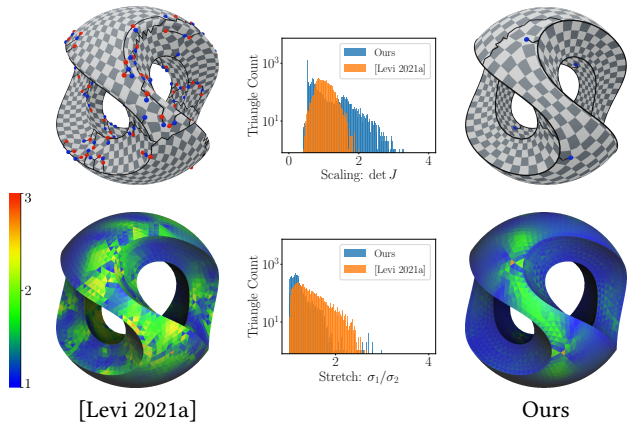


Fig. 12. Comparison with [Levi 2021a] on the sculpt model both methods minimize the stretch distortion. As shown by the heat map and the histogram, our method improves triangle stretch with ten times less cone singularities.

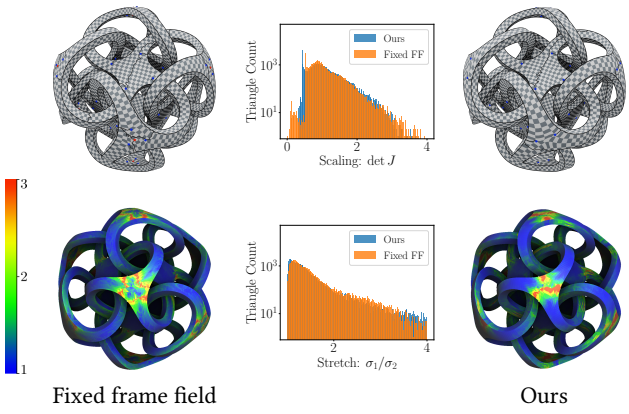


Fig. 13. Metatron model. Comparison between our parametrization and the one obtained by integrating a frame field generated with [Viertel and Osting 2019]. For similar distortion patterns we place two singularities instead of four.

Comparison with frame field integration. In this experiment, we investigate the benefits of optimizing for the rotation α along with the parametrization. To this end, we run our algorithm with a *fixed* frame field generated by Viertel *et al.* [Viertel and Osting 2019].

Although the two parametrizations are optimized for the same distortion energy, we remark that our free cone placement always outperforms the fixed frame field method in term of average distortion (Tables 1 and 2). Thus, we consistently obtain a better singularity distribution for stretching minimization. For the Metatron, Fig. 13 shows that reducing the number of cones by 30% leads to an equivalent stretch distribution over triangles at the price of more area distortion.

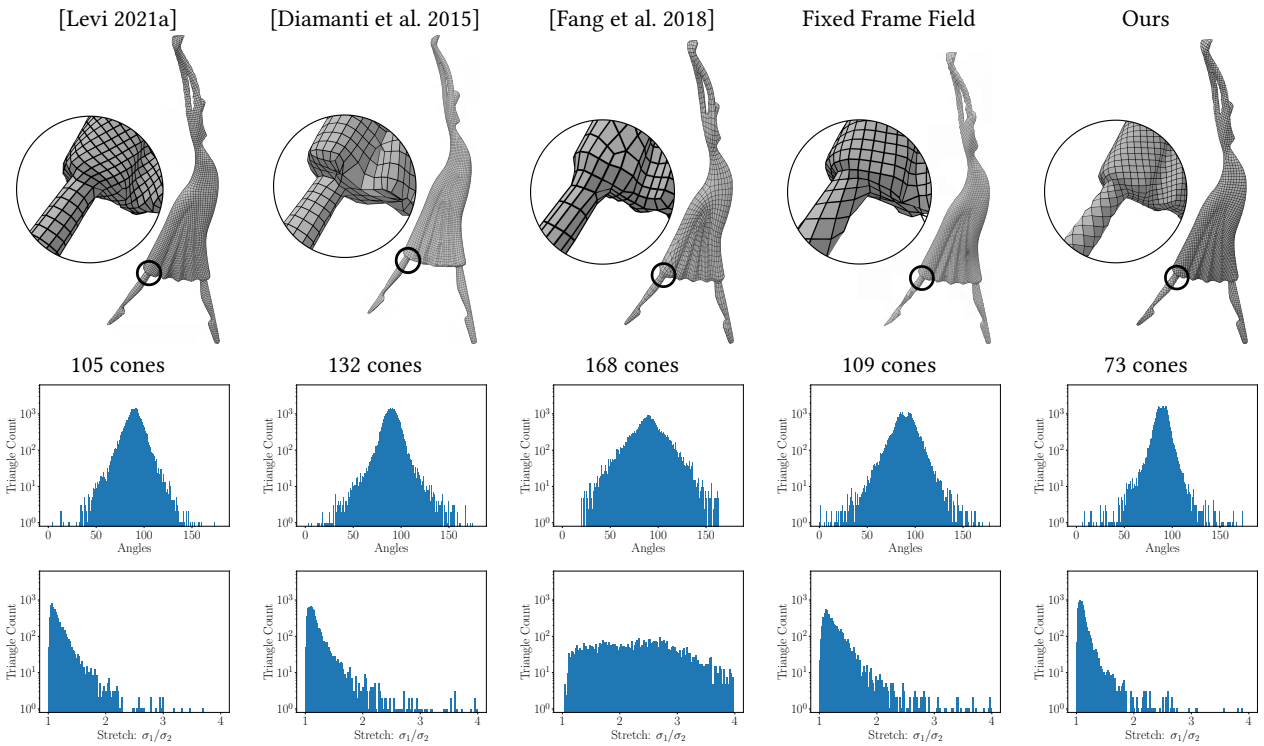


Fig. 14. Comparison of quadmeshes obtained by five algorithms on the dancer model. Top row: quad meshed. Second row: angles distribution over the model. Third row: histograms of stretch distortion. Our method yields minimal stretch and most orthogonal quads.

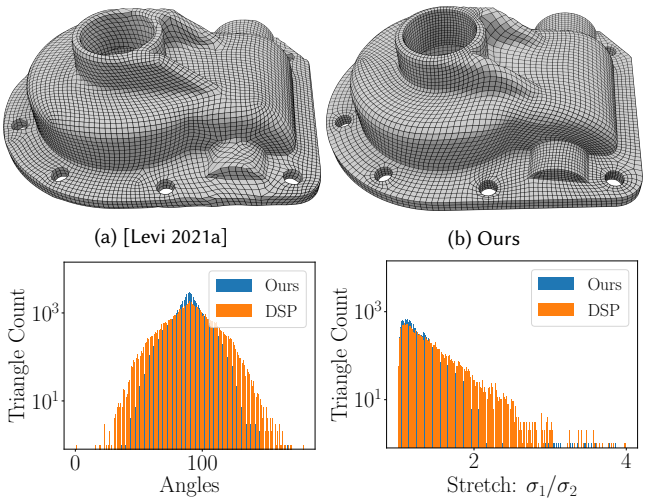


Fig. 15. Quad remeshing computed from the parametrization of [Levi 2021a] and ours. Our quads are less stretched and more orthogonal.

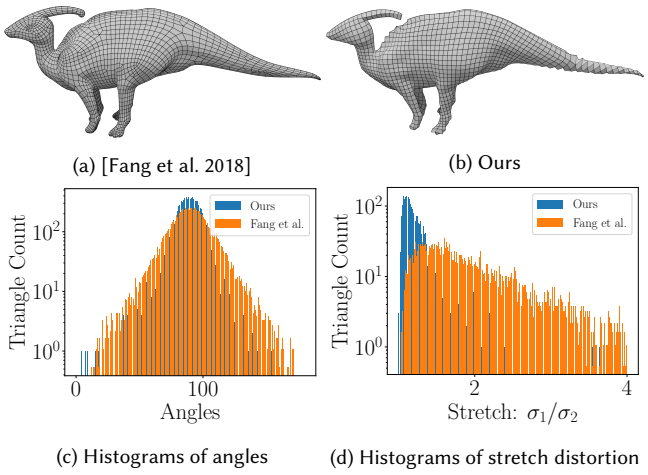


Fig. 16. Quad remeshing comparison with [Fang et al. 2018]. Our quads have less stretching and are have angles closer to 90°.

Comparison with [Levi 2021a]. Levi [2021a] pursues the same goal as us: directly computing a seamless parametrization from a triangle mesh. His method is greedy and is biased by its initialization. For the Dilo model in Fig. 11, the singularities appear on a single side of the mesh seem unrelated to the geometry. The parametrizations provided by the author minimize the maximum stretch. This can be observe in Fig. 12 where the stretch bounded. In comparison, our method yields a higher maximal value but a smaller stretch for a

majority of triangles (see histograms), as well as a reduced number of cone singularities – 16 against 217. Our optimization does not attempt to reduce triangle scaling thus our area distortion is not as competitive. Moreover, a great number of triangles tend to be stuck around 0.5 because of the barrier function threshold.

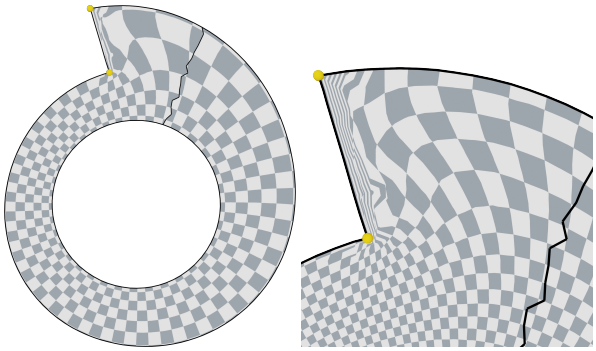
We also compare quad-meshes computed from our respective parametrizations in Figs. 15 and 14. In both cases, our quads have smaller stretch and are more orthogonal.

Comparison with [Diamanti et al. 2015]. The integrable Polyvector field method [Diamanti et al. 2015] takes an initial frame field and optimizes it to guarantee its integrability. However, this process only weakly minimizes distortion and is prone to introduce new singularities. In Fig. 11, we again exhibit a lower distortion for a majority of triangles as well as a smaller number of cones – 140 against 70 in this example.

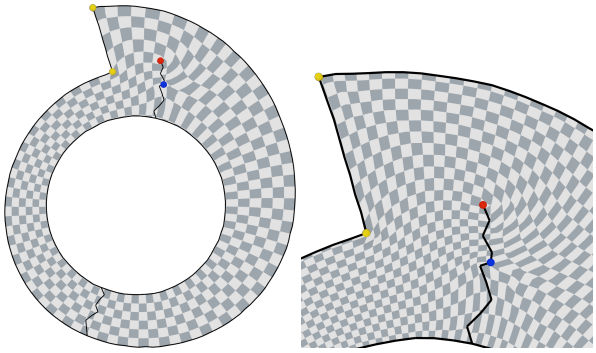
Comparison with Myles and Zorin [2012]. Myles and Zorin [2012] propose a greedy algorithm for cone placement along with a post-processing procedure to obtain $k/4$ singularity indices. Cones tend to be placed in high curvature regions which can be suboptimal. In general, we achieve lower stretch for fewer singularities (Table 1) as typically represented for the Dilo model in Figs. 11.

Comparison with Fang et al. [2018]. We also compare our results to a method producing quad-meshes without the quantization step. Fang *et al.* [2018] use periodic functions to integrate a frame field directly into an integer parametrization. This type of methods generally output a high quality mesh on most of the input surface. However, some localized regions need to be repaired which often introduce new singularities.

Figs. 16 and 14 show the histograms of our metrics for the Dilo model. Our quads are more orthogonal and suffer less from stretching.



(a) Typical failure case of our method.



(b) Changing the energy weights allows our method to place a dipole of singularities and retrieve a correct parametrization.

Fig. 17. On this model our method fails to find a valid parametrization (a). By using an isometric distortion energy and setting $\lambda_F = 0.1$, we are able to find a valid solution (b).

7 CONCLUSION AND FUTURE WORKS

By constraining a parametrization to a parallel transport instead of a cross field, we constructed an algorithm for global seamless parametrization without prescribing cones in advance. This direct link between singularities and the parametrization enables us to find cone points whose position minimize a given distortion energy. Unlike previous methods our solver is not greedy and does not rely on integer variables.

In the perspective of quad meshing, the main limitation of our work is that it does not handle integer coordinates of cone points. We still rely on a post-processing quantization step which impacts the overall distortion distribution. So the cone placement is optimal for the seamless parametrization but suboptimal for the final quadmesh.

A second limitation is that our current implementation is slower than previous methods because of the a higher number of variables – $2|F|+4|E|$ for vertex charts coordinates, $|E|$ for the parallel transport and $2|V|$ for the cross field. In the future, we would like to remove the cross field in the parallel transport constraints, to reduce the number of unknowns.

We believe that this contribution will have a great impact on future works because the parallel transport integrability constraints (Sec. 3) can be rewritten for higher dimension meshes. This opens the door to computing valid seamless parametrization for tetrahedral meshes which is a currently a major challenge for hexmeshing.

REFERENCES

Noam Aigerman and Yaron Lipman. 2015. Orbifold tutte embeddings. *ACM Trans. Graph.* 34, 6 (2015), 190–1.

Mirela Ben-Chen, Craig Gotsman, and Guy Bunin. 2008. Conformal flattening by curvature prescription and metric scaling. In *Computer Graphics Forum*, Vol. 27. Wiley Online Library, 449–458.

David Bommes, Marcel Campen, Hans-Christian Ebke, Pierre Alliez, and Leif Kobbelt. 2013. Integer-grid maps for reliable quad meshing. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–12.

Marcel Campen, David Bommes, and Leif Kobbelt. 2015. Quantized global parametrization. *Acm Transactions On Graphics (tog)* 34, 6 (2015), 1–12.

Marcel Campen, Hanxiao Shen, Jiaran Zhou, and Denis Zorin. 2018. Seamless Parametrization with Arbitrarily Prescribed Cones. *arXiv preprint arXiv:1810.02460* (2018).

Keenan Crane, Mathieu Desbrun, and Peter Schröder. 2010. Trivial connections on discrete surfaces. In *Computer Graphics Forum*, Vol. 29. Wiley Online Library, 1525–1533.

Fernando de Goes, Mathieu Desbrun, and Yiying Tong. 2016. Vector field processing on triangle meshes. In *ACM SIGGRAPH 2016 Courses*. 1–49.

David Desobry, François Protais, Nicolas Ray, Etienne Corman, and Dmitry Sokolov. 2021. Frame Fields for CAD Models. In *International Symposium on Visual Computing*. Springer, 421–434.

Olga Diamanti, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. 2015. Integrable polyvector fields. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–12.

Hans-Christian Ebke, David Bommes, Marcel Campen, and Leif Kobbelt. 2013. QEx: Robust quad mesh extraction. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 1–10.

Xianzhong Fang, Hujun Bao, Yiying Tong, Mathieu Desbrun, and Jin Huang. 2018. Quadrangulation through morse-parameterization hybridization. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–15.
Xiao-Ming Fu and Yang Liu. 2016. Computing inversion-free mappings by simplex assembly. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 1–12.
Kai Hormann, Konrad Polthier, and Alia Sheffer. 2008. Mesh Parameterization: Theory and Practice. In *ACM SIGGRAPH ASIA 2008 Courses* (Singapore) *(SIGGRAPH Asia '08)*. Association for Computing Machinery, New York, NY, USA, Article 12, 87 pages. https://doi.org/10.1145/1508044.1508091
Felix Kälberer, Matthias Nieser, and Konrad Polthier. 2007. Quadcover-surface parameterization using branched coverings. In *Computer graphics forum*, Vol. 26. Wiley Online Library, 375–384.
Michael Kazhdan, Jake Solomon, and Mirela Ben-Chen. 2012. Can mean-curvature flow be modified to be non-singular?. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 1745–1754.
Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2013. Globally optimal direction fields. *ACM Trans. Graph.* 32, 4 (2013).
Marin Kobilarov, Keenan Crane, and Mathieu Desbrun. 2009. Lie group integrators for animation and control of vehicles. *ACM transactions on Graphics (TOG)* 28, 2 (2009), 1–14.
Zohar Levi. 2021a. Direct Seamless Parametrization. *ACM Transactions on Graphics (TOG)* 40, 1 (2021), 1–14.
Zohar Levi. 2021b. Seamless Parametrization of Spheres with Controlled Singularities. In *Computer Graphics Forum*. Wiley Online Library.
Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. 2002. Least squares conformal maps for automatic texture atlas generation. *ACM transactions on graphics (TOG)* 21, 3 (2002), 362–371.
Max Lyon, David Bommes, and Leif Kobbelt. 2016. HexEx: Robust hexahedral mesh extraction. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–11.
Naoki Marumo, Takayuki Okuno, and Akiko Takeda. 2020. Constrained Levenberg-Marquardt method with global complexity bound. *arXiv preprint arXiv:2004.08259* (2020).
Ashish Myles, Nico Pietroni, and Denis Zorin. 2014. Robust field-aligned global parametrization. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–14.
Ashish Myles and Denis Zorin. 2012. Global parametrization by incremental flattening. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–11.
Ashish Myles and Denis Zorin. 2013. Controlled-distortion constrained global parametrization. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–14.
Jonathan Palacios and Eugene Zhang. 2007. Rotational symmetry field design on surfaces. *ACM Transactions on Graphics (TOG)* 26, 3 (2007), 55–es.
Daniele Panozzo, Enrico Puppo, Marco Tarini, and Olga Sorkine-Hornung. 2014. Frame fields: Anisotropic and non-orthogonal cross fields. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–11.
Nicolas Ray, Wan Chiu Li, Bruno Lévy, Alla Sheffer, and Pierre Alliez. 2006. Periodic global parameterization. *ACM Transactions on Graphics (TOG)* 25, 4 (2006), 1460–1485.
Nicolas Ray, Bruno Vallet, Wan Chiu Li, and Bruno Lévy. 2008. N-symmetry direction field design. *ACM Transactions on Graphics (TOG)* 27, 2 (2008), 1–13.
Rohan Sawhney and Keenan Crane. 2018. Boundary first flattening. *ACM Transactions on Graphics (ToG)* 37, 1 (2018), 5.
Yousuf Soliman, Dejan Slepčev, and Keenan Crane. 2018. Optimal cone singularities for conformal flattening. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–17.
Olga Sorkine and Marc Alexa. 2007. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, Vol. 4. 109–116.
Boris Springborn, Peter Schröder, and Ulrich Pinkall. 2008. Conformal equivalence of triangle meshes. In *ACM SIGGRAPH 2008 papers*. 1–11.
B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. 2020. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation* 12, 4 (2020), 637–672. https://doi.org/10.1007/s12532-020-00179-2
William Thomas Tutte. 1963. How to draw a graph. *Proceedings of the London Mathematical Society* 3, 1 (1963), 743–767.
Amir Vaxman, Marcel Campen, Olga Diamanti, Daniele Panozzo, David Bommes, Klaus Hildebrandt, and Mirela Ben-Chen. 2016. Directional field synthesis, design, and processing. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 545–572.
Ryan Viertel and Braxton Osting. 2019. An Approach to Quad Meshing Based on Harmonic Cross-Valued Maps and the Ginzburg–Landau Theory. *SIAM Journal on Scientific Computing* 41, 1 (2019), A452–A479.
Ana Maria Vintescu, Florent Dupont, and Guillaume Lavoué. 2017. Least squares affine transitions for global parameterization. (2017).
Jiayi Eris Zhang, Alec Jacobson, and Marc Alexa. 2021. Fast Updates for Least-Squares Rotational Alignment. In *Computer Graphics Forum*, Vol. 40. Wiley Online Library, 13–22.

A PROOF OF THEOREM 3.1

Proof. The parametrization resulting of Alg. 1 can be equivalently obtained by (1) synchronizing charts along a spanning tree on vertices, (2) triangulating quads by adding an edge between triangle centers and triangle vertices and (3) isometrically moving pieces of triangles across the cut so that only singular triangles are split and the cut follows primal edges. Therefore, the quad parametrization is equivalent to the parametrization of Alg. 1 in the sense that they share the same singularity index around loops. For this reason, we will carry out the proof on the quad parametrization which is simpler to relate to transport rotations.

Necessary condition. Suppose that there exists charts satisfying Eqs. ($\mathcal{E}$), we will show that Alg. 1 constructs a parametrization compatible with the holonomy of ω.

Since the quad parametrization is built by recursively aligning charts along a (primal) spanning tree, a loop of vertices is obtained by considering an unvisited edge $i_0 i_N$ and walking back to the tree root from both side of the edge. Let $\gamma = (i_0, \dots, i_N, i_0)$ be such a loop. We denote $r_i \in \mathbb{C}$ the total rotation of chart C_i during the synchronization. Let $u_{i_0} \in \mathbb{C}, v_{i_N} \in \mathbb{C}$ be duplicated edge vectors of charts C_{i_0}, C_{i_N} and $u = r_{i_0} u_{i_0}, v = r_{i_N} v_{i_N}$ their coordinates in the quad parametrization. Applying rotations recursively along the loop leads to:

$$\begin{aligned} u &= r_{i_0} u_{i_0} \\ &= r_{i_1} \exp(\imath \omega_{i_0 i_1}) v_{i_0} \\ &= r_{i_N} \prod_{k=0}^{N-1} \exp(\imath \omega_{i_k i_{k+1}}) v_{i_0} \\ &= r_{i_N} \prod_{k=0}^{N-1} \exp(\imath \omega_{i_k i_{k+1}}) \exp(\imath \omega_{i_N i_0}) v_{i_N} \\ &= \Phi_\omega(\gamma) v. \end{aligned}$$

Therefore, duplicated edges, which include edge at the quad parametrization cut, are equal up to the total rotation along the loop. If $\Phi_\omega(\gamma)$ is a rotation by an angle $k\pi/2, k \in \mathbb{Z}$ for all loops then the parametrization is seamless.

Sufficient condition. Suppose that we are given a parametrization whose singular vertices are of valence three. Let us subdivide regular triangles into three quad and turn the three triangles adjacent to a singularity into three quads. By moving triangle pieces across cuts so that the cut follows dual edges, we recover a quad-parametrization with transition rotations $r \in \mathbb{C}, |r| = 1$. The choice of ω equal to $\arg(r)$ on cuts and 0 on regular edges and charts coordinates equal to the parametrization coordinates obviously satisfies Eqs. ($\mathcal{E}$). □

B PROOF OF THEOREM 4.1

Proof. Since ω is compatible with a frame field (Eqs. ($\mathcal{F}$)), its holonomy on any loops is constrained to be an integer multiple of $\pi/2$. Since it also satisfies Eqs. $\mathcal{E}$, Thm. 3.1 guarantees that the reconstructed parametrization is seamless.

Eqs. ($\mathcal{D}_1$) ensure local injectivity as all triangles (p_i, p_j, s_{ijk}) composing the parametrization must have positive area.

Eqs. ($\mathcal{B}$) force the feature edge in charts to be aligned with the cross field and corner vertex have $k\pi/2$ total inner angle. Since

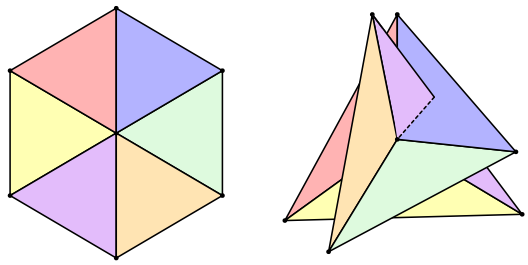


Fig. 18. Two version of the same vertex rings: one with an angle defect of 0 and the other with an angle defect of 2π, both have only positive area triangles.

Eqs. $\mathcal{E}$ are satisfied the feature edges are reconstructed as straight lines with possible corners at vertices. Because ω is compatible with a frame field aligned with feature edges, two disconnected sets of feature edges linked by a vertex path γ will be related by a total rotation of $k\pi/2$. Thus, feature edges are (up to a global rotation) isolines of the parametrization.

Double coverings. We are now going to prove that the parametrization singularities are exactly those prescribed by the parallel transport. According to Thm. 3.1 the parametrization transition functions match the parallel transport holonomy. Therefore the parametrization angle defect at vertices is equal to the parallel transport Gaussian curvature modulo 2π. We will show that this additional parasite angle corresponding to integer singularities cannot appear in our parametrization.

By construction, singular vertices have only three incident edges, thus the angle defect is strictly bounded by $-\pi$ and 2π – otherwise triangles are invalid. So, angle defects of singular triangles are exactly equal to the prescribed Gaussian curvature.

Vertices of the input mesh also have prescribed singularity indices: 0 for inner vertices and user prescribed on feature edges. At these vertices, unwanted *negative* integer index singularities could appear even if all triangles positive areas (see Fig. 18). However, since ω satisfies Eqs. ($\mathcal{E}$), its total Gaussian curvature is constraint by the Gauss-Bonnet theorem. Thus, any additional singularities must have indices summing to 0 to maintain feasibility of the parametrization. As a consequence, any strictly negative index singularities appearing at a vertex must be compensated by positive singularities. As we have shown, these singularities cannot appear inside triangles and vertices only accept negative singularities. Thus, double coverings cannot happen. □

C PROOF OF PROPOSITION 5.1

Proof. Let us show that if there exists two piece-wise linear functions $f, g \in \mathbb{R}^{|V|}$ on a flat triangle mesh such that $\nabla f = \mathcal{J}\nabla g$ then they must be a rotation and a global scaling of the vertex positions.

By construction of piece-wise linear finite elements, the gradient operator is continuous in the direction tangent to an edge:

$$\langle \nabla f_{ijk}, e_{ij} \rangle = \langle \nabla f_{ijl}, e_{ij} \rangle.$$

The function g satisfies the LSCM equation, so that ∇f is also continuous in the direction normal to the edge:

$$\begin{aligned}\langle \nabla f_{ijk}, \mathcal{J} e_{ij} \rangle &= \langle \mathcal{J}^\top \nabla f_{ijk}, e_{ij} \rangle \\ &= -\langle \nabla g_{ijk}, e_{ij} \rangle \\ &= -\langle \nabla g_{ijl}, e_{ij} \rangle \\ &= \langle \nabla f_{ijl}, \mathcal{J} e_{ij} \rangle.\end{aligned}$$

As ∇f and ∇g are continuous across all edges, they must be constant over the entire mesh. Therefore, there exists $a, b, c, d \in \mathbb{R}^2$ such that f and g are affine functions:

$$f_i = a^\top \begin{pmatrix} x_i \\ y_i \end{pmatrix} + b, \qquad g_i = c^\top \begin{pmatrix} x_i \\ y_i \end{pmatrix} + d.$$

Using the LSCM equation again, the vectors a and c are such that $c = \mathcal{J}a$, therefore f, g are scaled-rotation of the vertex coordinates. □