Integrable Frame Fields using Odeco Tensors

Mattéo Couplet* Alexandre Chemin* Jean-François Remacle*

Abstract

We propose a method for computing integrable orthogonal frame fields on planar surfaces. Frames and their symmetries are implicitly represented using orthogonally decomposable (odeco) tensors. To formulate an integrability criterion, we express the frame field's Lie bracket solely in terms of the tensor representation; this is made possible by studying the sensitivity of the frame with respect to perturbations in the tensor. We construct an energy formulation that computes smooth and integrable frame fields, in both isotropic and anisotropic settings. The user can prescribe any size and orientation constraints in input, and the solver creates and places the singularities required to fit the constraints with the correct topology. The computed frame field can be integrated to a seamless parametrization that is aligned with the frame field.

1 Introduction

Meshes composed of quadrilaterals are known to offer superior performance than their triangular counterpart when used as a support in numerical simulations, however quad meshers still do not meet the same level of robustness, flexibility and quality required for industrial applications as triangular meshes. In three dimensions, generating fully hexahedral meshes is an even more difficult task, and their industrial use is consequently very limited despite a persistent demand from practitioners.

Quadrilateral and hexahedral meshing are challenging because they couple (a) a *geometric* problem, minimizing the distortion of the elements, and (b) a *combinatorial* problem, achieving a conforming connectivity structure. If one also desires a multi-block structure, a layer of complexity is added, requiring (c) an adequately coarse block structure. The last two decades have seen the emergence of *field-based approaches*, which divide the problem in two main steps. (1) The combinatorial constraints are ignored and a *frame field* is computed; a frame is a set of 2/3 directions representing the orientation (and sometimes the size) of a quad/hex. This frame field can be seen as a continuous extension of a mesh. (2) A mesh is generated using guidance from the frame field. This can be done through numerous approaches which we review below.

*University of Louvain, Belgium.

An issue shared by most existing works is that no quad/hex mesh exactly follows the frame field computed a priori. This is due to the frame field not being *integrable*; we elaborate on this in section 2. This limitation of frame field-based methods means that the mesh deviates from the frame field and the user must compromise on element quality, control over element size and orientation, and control over the topology of the mesh. A more fundamental issue arises when the computed frame field has a global topology that is not meshable, making it unusable. This happens in the 2D case when limit cycles appear; in 3D the singular structure is very often invalid due to the presence of non-meshable singular nodes.

Our contribution overcomes this limitation in 2D and allows to generate frame fields on planar surfaces that are *integrable* (Figure 1, left); during this process the user can prescribe any orientation or size constraints along feature curves. Integrability guarantees that the frame field can be integrated to a seamless parametrization that is exactly aligned with the frame field and respects the prescribed constraints (Figure 1, right). The parametrization can then be quantized to extract a quad mesh, or the frame field can be directly used as a guide to a frontal mesher; we illustrate the latter in our results.

Our approach relies on orthogonally decomposable (*odeco*) tensors which serve as an algebraic representation for the frames. By studying the so-called *eigenvalue sensitivity problem* for tensors, we are able to formulate the problem completely in terms of this implicit algebraic representation. This work is the first achieving integrability using odeco tensors. As they can be extended to three dimensions, this contribution paves the way for hex-meshable frame fields, which are allegedly the key to the long-standing problem of robustly generating optimal hexahedral meshes.

1.1 Related work

Frame field design. 4-direction fields, i.e., assignments of four directions to every point of a surface, and their application to quad meshing have been studied extensively in the computer graphics community; paper [22] provides a review. Some important works [18, 5] assume the field topology to be known in advance and

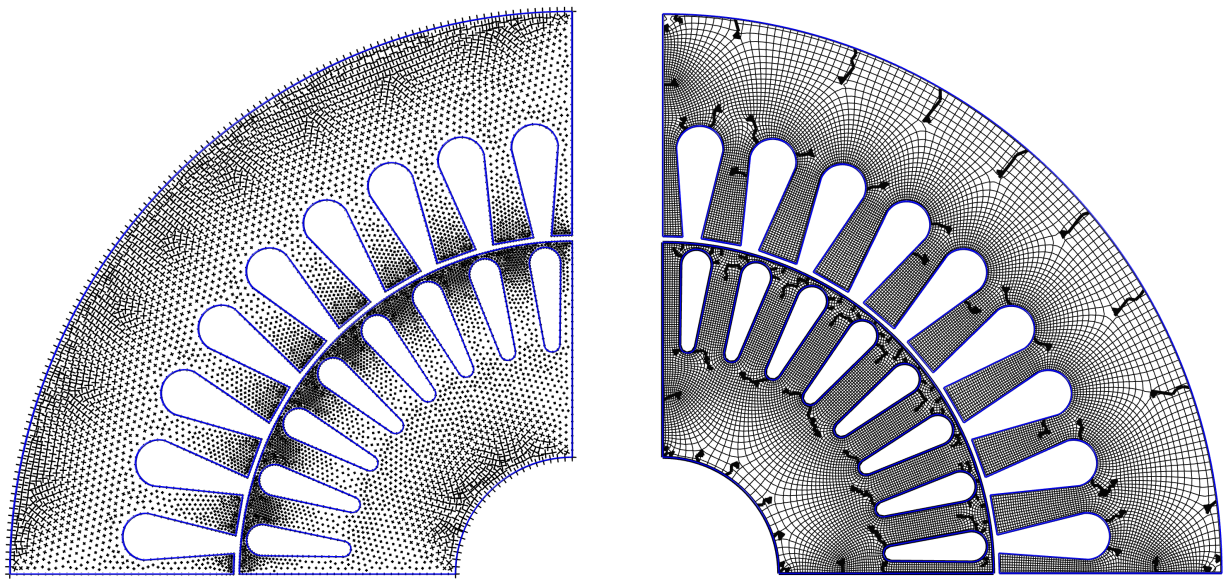

Figure 1: Overview of the approach on a machine composed of a rotor and stator. (Left) An integrable frame field is computed respecting user-prescribed size constraints: size of 1 on the inner and outer arcs, and size of 0.3 on the other boundaries. The frames indicate the local size and orientation of the quadrilateral mesh elements. (Right) A seamless parametrization is computed by integrating the frame field. Black triangles represent singularities and thick lines form the cut graph.

optimize for cross field smoothness. If the topology is unknown, it can be represented explicitly and optimized through integer variables such as in [2], which often leads to mixed-integer formulations. Alternatively, the topology can also be implicitly encoded in the field representation; our work fits in this category. Earlier works in this line of research include [17, 11]. A key challenge is being able to represent singularities while maintaining unit norm frames. More recently, this problem has been re-framed in the Ginzburg-Landau framework, which replaces the ill-posed unit norm constraint by a penalty term taken to the limit [1, 23].

Field-guided meshing. As an intermediate step before generating a quad mesh, a global parametrization is often computed on the domain using the guidance from the frame field. Among notable works we can cite [10, 2] who integrate the frame field in a least-squares sense to find the parametrization that best aligns to the frame field, or [17], who perform a curl-reduction procedure a posteriori. Another approach, e.g., in [14], is to trace out parametric lines from the frame field to form quadrilateral patches. Getting a quad mesh from a global parametrization is typically done using quantization methods, where a T-mesh is traced out and its edges are given integer lengths; see, e.g., [3, 13].

Obtaining a quad mesh from a frame field can also be achieved robustly using frontal method, by inserting points with guidance from the frame field. Remeshing and smoothing is often necessary a posteriori to fix regions of bad quality; see, e.g., [19], which we use as a quad mesher in our pipeline.

In this work, the parametrization step is made trivial as an integrable frame field is equivalent to a seamless parametrization. Hence, integrating the frame field provides a parametrization that is exactly aligned and matches the sizing of the frame field. This confers the user a complete control over the sizing and orientation of the mesh, a desirable property that cannot be achieved through conventional cross field guided meshing.

Frame representation. Computing frame fields with implicit topology requires a representation that is invariant to the ordering and symmetries of the frame vectors. The majority frame field-driven methods use representations of *unit* frames, or *crosses*, as they only care about the orientation of the mesh; a review can be found in [22]. One of the most prominent representations is the trigonometric pair $(\cos(4\theta), \sin(4\theta))$ (or, equivalently, the complex exponential $e^{i4\theta}$) which

effectively encodes rotational symmetries and is trivially normalized to unit norm [17, 15, 12, 11]. However, this representation cannot encode sizes along with the directions. On the other hand, PolyVectors, introduced by [6], encode the frame vectors as the complex roots of a degree 4 polynomial. They have the advantage that they can represent non-orthogonal frames (as done in [21]), but they offer no straightforward extension to volumetric frames. For representing 3D frames, a spherical function is often used that is maximal in the directions of the frame. This function is encoded by its coefficients in the basis of spherical harmonics [8]. As highlighted by [4], it can also be interpreted as a 4th-order tensor. Recently, Palmer et al. [16] have proposed an extension of these representations that encodes sizes along the directions, relying on the so-called orthogonally decomposable (Odeco) tensors. As they are the only known representation that can encode three-dimensional orthogonal and scaled frames, we propose to use them in the 2D setting so that they can offer a 3D extension for later work.

Integrable frame fields. Motivated by the issue where the mesh is not aligned to the computed frame field, a handful of works have focused on computing frame fields that are *integrable*, i.e., the direction vectors are the gradients of a seamless parametrization. This property holds if the two vector fields are *curl-free*. Diamanti et al. [7] achieve this using PolyVector fields. Unlike our approach, the integrability condition cannot be expressed in terms of the PolyVector coefficients, and the optimization is done on the explicit direction vectors. In [21], PolyVector fields are used to construct so-called Chebyshev nets, i.e., quadrilateral meshes with a uniform size but not necessarily orthogonal. Our goal is slightly different (orthogonal quad meshing) but we address the same challenges regarding control of size and orientation. Like us, they optimize a frame field instead of a parametrization Jacobian, and express the integrability through the Lie bracket. The main advantage of odeco tensors compared to PolyVectors is that the former offer a natural extension to 3D frame fields.

In [9], an integrable frame field is computed given a user-imposed (or pre-computed) singularity configuration. The frame fields are exactly integrable and the method can even remove limit cycles that appear in most field-based approaches. Having to specify singularities however is a significant limitation, especially in the 3D case where almost all existing frame field solvers produce invalid singular configurations.

1.2 Overview In section 2 we review the mathematical formalism behind parametrizations, their Jacobians and frame fields. We define the integrability property for frame fields and show how they are equivalent to seamless parametrizations. In section 3, we review the algebraic representation for our frames, which is the two-dimensional version of the odeco tensors proposed by [16]. In section 4, we present two contributions: (a) an expression for the sensitivity of a higher-order tensor's eigenvectors with respect to small perturbations of the tensor, and we use this result to show (b) an expression of a frame field's Lie bracket only in terms of the tensor field coefficients and their derivatives. In section 5 we present a last contribution, an energy formulation that optimizes for smooth integrable frame fields, for both the isotropic and anisotropic cases. In section 6 we briefly present our pipeline to compute a seamless parametrization from an integrable frame field, and in section 7 we demonstrate the effectiveness of the algorithm.

2 Integrable frame fields

Seamless parametrizations. The idea of parametrization-based quadrilateral meshing is to map a coordinate system onto the domain to be meshed. The coordinate lines of this map then provide, at least locally, a quadrilateral mesh on the domain. Consider a planar domain Ω; we wish to parametrize Ω by assigning to each point $\mathbf{p}$ of Ω a pair of coordinates $(u(\mathbf{p}), v(\mathbf{p}))$. In general, this parametrization cannot be defined as a global continuous function; *cuts* need to be introduced to obtain a disk topology on which u and v can be defined continuously. The parametrization is said to be *seamless* if across every cut, the coordinates transform through a rigid rotation of some multiple of $90°$; if u' and v' are the coordinates on the other side of the cut, then the transformation is seamless if

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}}_{\mathbf{R}}^{k} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} s \\ t \end{pmatrix}, \tag{2.1}$$

for some $k \in \{0, 1, 2, 3\}$ and fixed translation (s, t). If, on top of being seamless, the translation (s, t) is integer and the singularities lie at integer coordinates, then the parametrization is called an *integer-grid map* and exactly corresponds to a quadrilateral mesh.

Computing seamless parametrizations is a challenging task due to the discrete nature of the cut graph involved, which is unknown a priori. Instead, most existing works focus on computing the Jacobian $(\boldsymbol{\nabla} u, \boldsymbol{\nabla} v)$ of a parametrization. Let $(\mathbf{d}u, \mathbf{d}v)$ be a pair of vector fields on Ω; they form the Jacobian of a parametrization (and are said to be *integrable*) if they are curl-free:

$$\boldsymbol{\nabla} \times \mathbf{d}u = \mathbf{0}, \quad \boldsymbol{\nabla} \times \mathbf{d}v = \mathbf{0}, \tag{2.2}$$

and they transform through a 90°-multiple rotation across cuts (which amounts to differentiating (2.1)):

$$\begin{pmatrix} \mathbf{d}u' \\ \mathbf{d}v' \end{pmatrix} = \mathbf{R}^k \begin{pmatrix} \mathbf{d}u \\ \mathbf{d}v \end{pmatrix}. \tag{2.3}$$

If these properties are verified then the vector fields $(\mathbf{d}u, \mathbf{d}v)$ are essentially equivalent to a seamless parametrization, up to a global rigid rotation of the coordinate map. Computing a parametrization's Jacobian is a more doable task, provided one can appropriately encode the symmetries in (2.3); we elaborate on that in section 3.

Frame fields. Consider now the inverse of the parametrization $\mathbf{r}$ that maps a pair of coordinates to point of the domain: $\mathbf{p} = \mathbf{r}(u, v)$. This map can only be defined locally since the parametrization can map different points to the same coordinates. The Jacobian matrix of $\mathbf{r}$ defines the *coordinate vectors* $\mathbf{u}$ and $\mathbf{v}$:

$$\mathbf{J}_\mathbf{r} = \begin{pmatrix} \frac{\partial \mathbf{r}}{\partial u} & \frac{\partial \mathbf{r}}{\partial v} \end{pmatrix} = \begin{pmatrix} \mathbf{u} & \mathbf{v} \end{pmatrix}, \tag{2.4}$$

forming a *coordinate frame* $\mathbf{F} = \begin{pmatrix} \mathbf{u} & \mathbf{v} \end{pmatrix}$. Since $\mathbf{r}$ is the inverse of the parametrization, their Jacobians are inverse of one another, meaning that the coordinate frame is the inverse of the parametrization's Jacobian:

$$\mathbf{J}_{\mathbf{r}^{-1}} = (\mathbf{J}_\mathbf{r})^{-1} \quad \Rightarrow \quad \begin{pmatrix} \nabla u \\ \nabla v \end{pmatrix} = \mathbf{F}^{-1}. \tag{2.5}$$

Note that, looking at the off-diagonal terms in $\mathbf{F}^{-1}\mathbf{F} = \mathbf{I}$, we have $\nabla u \cdot \mathbf{v} = \nabla v \cdot \mathbf{u} = 0$, meaning that the coordinate vectors $\mathbf{u}, \mathbf{v}$ are orthogonal to the gradients of v, u, respectively, and therefore parallel to the isolines of v, u respectively. This shows how the frame field corresponds to a quad mesh through discrete isolines of the coordinate map u, v. Looking at the diagonal terms, we have $\nabla u \cdot \mathbf{u} = \nabla v \cdot \mathbf{v} = 1$, which indicates that the integer isolines are spaced out according to $\|\mathbf{u}\|$ and $\|\mathbf{v}\|$. We illustrate the introduced concepts and notations on Figure 2.

Orthogonal case. In this work we are interested in frames that are orthogonal: $\mathbf{u} \perp \mathbf{v}$. Let $\hat{\mathbf{u}} = \mathbf{u}/\|\mathbf{u}\|$ and $\hat{\mathbf{v}} = \mathbf{v}/\|\mathbf{v}\|$; the frame matrix $\mathbf{F}$ is then diagonal in the local orthonormal basis $(\hat{\mathbf{u}}, \hat{\mathbf{v}})$:

$$\begin{pmatrix} \nabla u \\ \nabla v \end{pmatrix} = \begin{pmatrix} \|\mathbf{u}\| & 0 \\ 0 & \|\mathbf{v}\| \end{pmatrix}^{-1} = \begin{pmatrix} 1/\|\mathbf{u}\| & 0 \\ 0 & 1/\|\mathbf{v}\| \end{pmatrix}. \tag{2.6}$$

The Jacobian of the coordinate map can thus be computed by simply scaling the frame's coordinate vectors: $\nabla u = \mathbf{u}/\|\mathbf{u}\|^2$ and $\nabla v = \mathbf{v}/\|\mathbf{v}\|^2$.

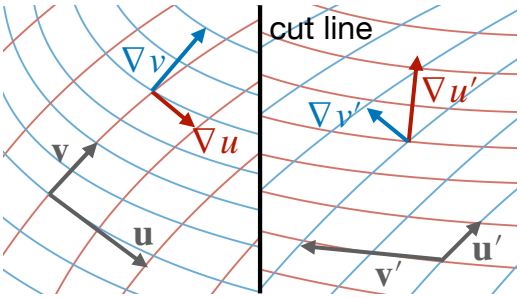

Figure 2: A seamless parametrization across a cut. The isolines of the coordinate maps $u(\mathbf{p})$ and $v(\mathbf{p})$ are depicted in red and blue, respectively. Notice how the gradients $\nabla u, \nabla v$ and the frame vectors $\mathbf{u}, \mathbf{v}$ undergo a 90° rotation as they cross the cut.

Frame field integrability. Several previous works, e.g., [7], build upon the curl-free condition (2.2) and compute parametrizations by computing vector fields $(\mathbf{d}u, \mathbf{d}v)$. However, in our case we want to be able to express the integrability condition directly in terms of our implicit frame representation, and expressing the individual curls of $\mathbf{d}u$ and $\mathbf{d}v$ is not possible as the two vectors are mixed in a single algebraic representation. Instead, we compute a frame field $\mathbf{F}$ and rely on the analogous statement: a frame field $\mathbf{F} = \begin{pmatrix} \mathbf{u} & \mathbf{v} \end{pmatrix}$ guides a parametrization (and is *integrable*) if its *Lie bracket* $[\mathbf{u}, \mathbf{v}]$ vanishes:

$$[\mathbf{u}, \mathbf{v}] = \nabla_\mathbf{u}\mathbf{v} - \nabla_\mathbf{v}\mathbf{u} = \mathbf{0}, \tag{2.7}$$

where $\nabla_\mathbf{u}$ and $\nabla_\mathbf{u}$ are directional derivatives, and this parametrization is seamless if the frame vectors transform through 90° rotations across cuts, as in (2.3). We will see in section 4 how the Lie bracket can be expressed in terms of the implicit frame representation.

3 Algebraic representation of frames

Due to the topology of the domain Ω and the presence of singularities in the frame field, it is not possible to define a globally continuous frame field $(\mathbf{u}, \mathbf{v})$. One needs to introduce *cuts* in the domain across which the frame vectors are symmetric according the rotations defined by (2.3). In order to compute a frame field without the need to introduce cuts, we extend the notion of frame such that it is invariant up to 90° rotations:

$$\mathbf{F} = \{(\mathbf{u}, \mathbf{v}), (\mathbf{v}, -\mathbf{u}), (-\mathbf{u}, -\mathbf{v}), (-\mathbf{v}, \mathbf{u})\}. \tag{3.8}$$

These symmetries are illustrated in Figure 3. Given this equivalence class, one needs to introduce an algebraic representation that unambiguously represents a frame and supports arithmetic operations. To this end we resort to the class of two-dimensional orthogonally decomposable (or Odeco) tensors, which were introduced by

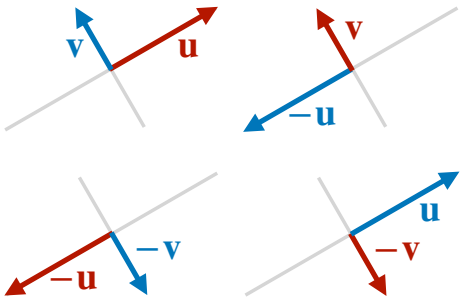

Figure 3: Equivalence class of the 90° rotations of an orthogonal frame $(\mathbf{u}, \mathbf{v})$.

Palmer et al. [16] in the three-dimensional case. In this section we briefly lay out the theory of odeco tensors, and refer the reader to the paper for more details.

Odeco tensors. Although the odeco theory applies to tensors of arbitrary order, we restrict the definitions to fourth-order tensors, as they are sufficient to represent frames. Let $S^4(\mathbb{R}^n)$ be the space of $n \times n \times n \times n$ *fully symmetric tensors*, i.e., their real entries T_{i_1,i_2,i_3,i_4} are invariant up to permutations of the indices i_1, i_2, i_3, i_4. n is the *dimension* of the tensor and corresponds to the dimension of the frame we wish to represent (2 or 3). A combinatorial inspection shows that tensor $\mathbf{T}$ has 5 independent components for $n = 2$, and 15 components for $n = 3$.

Robeva et al. [20] have studied a special class of tensors $\mathbf{T}$ that are said to be *orthogonally decomposable*, or *odeco* for short, if they can be written as

$$\mathbf{T} = \lambda_1 \mathbf{v}_1^{\otimes 4} + \cdots + \lambda_n \mathbf{v}_n^{\otimes 4}, \tag{3.9}$$

where $\mathbf{v}_1, \ldots, \mathbf{v}_n$ form an orthonormal basis of $\mathbb{R}^n$. Note that for second-order tensors, $S^2(\mathbb{R}^n)$ is the space of real symmetric matrices; these are always orthogonally decomposable according to the spectral theorem. Higher-order tensors (≥ 3) however, admit rank-one decompositions with more than n terms.

The notion of eigenvector can be generalized to higher-order tensors: $\mathbf{w}$ is an eigenvector of $\mathbf{T}$ with eigenvalue λ if

$$\mathbf{T}\mathbf{w}^3 = \lambda \mathbf{w}, \quad \text{or, in Einstein notation,} \quad T_{i,j_2,j_3,j_4} w_{j_2} w_{j_3} w_{j_4} = \lambda w_i. \tag{3.10}$$

One can easily see that, for an odeco tensor $\mathbf{T}$, the vectors $\mathbf{v}_k$ in (3.9) are eigenvectors of $\mathbf{T}$ with corresponding eigenvalue λ_k.

As odeco tensors only form a small subset of the space of fully symmetric tensors, one needs to characterize what makes a tensor odeco. The major result of Robeva et al. [20] is the fact that odeco tensors form an algebraic variety defined by a set of homogeneous quadratic equations in the tensor coefficients. More precisely, $\mathbf{T}$ is odeco if the contraction $(\mathbf{T} * \mathbf{T})$ defined by

$$(\mathbf{T} * \mathbf{T})_{i_1,i_2,i_3,j_1,j_2,j_3} = T_{i_1,i_2,i_3,s} T_{j_1,j_2,j_3,s} \tag{3.11}$$

is a fully symmetric tensor, i.e.,

$$\mathbf{T} * \mathbf{T} \in S^6(\mathbb{R}^n). \tag{3.12}$$

Frames as odeco tensors. To represent a 2D orthogonal frame in a way that is invariant to the permutations in (3.8), we use an odeco tensor of which the eigenvalues and eigenvectors, as defined in (3.10), match the frame directions and sizes. Specifically, let $\mathbf{u}, \mathbf{v}$ be the frame vectors, λ, μ their norms and $\hat{\mathbf{u}}, \hat{\mathbf{v}}$ their normalization such that $\mathbf{u} = \lambda \hat{\mathbf{u}}$, $\mathbf{v} = \mu \hat{\mathbf{v}}$. Then the corresponding tensor is defined as

$$\mathbf{T} = \lambda \hat{\mathbf{u}}^{\otimes 4} + \mu \hat{\mathbf{v}}^{\otimes 4}. \tag{3.13}$$

Notice how this tensor remains the same when plugging in any of the rotations of (3.8). We can now justify the choice for a fourth-order tensor: an even order is required for the tensor to be invariant to the signs of the vectors, and order 2 cannot be chosen since, in the case $\lambda = \mu$, the tensor degenerates to a multiple of the identity matrix, $\mathbf{T} = \lambda \mathbf{I}$, and then any vector of $\mathbb{R}^2$ is an eigenvector of $\mathbf{T}$ and one loses the direction vectors $\hat{\mathbf{u}}$, $\hat{\mathbf{v}}$. Order 4 is therefore the lowest possible order for our purpose.

Polynomial representation of tensors. There is a one-to-one correspondence between 4th-order fully symmetric tensors of dimension n, and degree 4 homogeneous polynomials of n variables, given by

$$p_{\mathbf{T}}(x_1, \ldots, x_n) = T_{i_1,i_2,i_3,i_4} x_{i_1} x_{i_2} x_{i_3} x_{i_4}. \tag{3.14}$$

Since it is homogeneous, one can restrict this polynomial to the sphere $\mathbb{S}_{n-1}$, since

$$p_{\mathbf{T}}(\mathbf{x}) = \|\mathbf{x}\|^4 \, p_{\mathbf{T}}\left(\frac{\mathbf{x}}{\|\mathbf{x}\|}\right). \tag{3.15}$$

We can therefore define a function $p_{\mathbf{T}}(\theta)$ such that $p_{\mathbf{T}}(\mathbf{x}) = \|\mathbf{x}\|^4 \; p_{\mathbf{T}}(\theta)$; this function is periodic with period π, since $p_{\mathbf{T}}(\mathbf{x}) = p_{\mathbf{T}}(-\mathbf{x})$. Such a function can be conveniently visualized by taking a unit circle and virtually stretching it according to the value that $p_{\mathbf{T}}$ takes on the circle; formally, one draws the parametric curve given by $r(\theta) = p_{\mathbf{T}}(\theta)$ for $\theta \in [0, 2\pi[$. We show this on Figure 4.

A natural way to encode this polynomial is to decompose it into an orthonormal basis of functions on

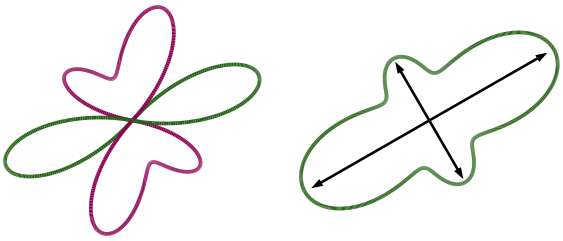

Figure 4: Polynomial representation $p_{\mathbf{T}}(\theta)$ of (left) an arbitrary tensor, and (right) an odeco tensor, with the corresponding frame vectors. Green represents positive values for $p_{\mathbf{T}}(\theta)$ and magenta represents negative values.

Figure 5: Orthonormal basis of circular harmonics used to represent odeco tensor polynomials. Green represents positive values for $p_{\mathbf{T}}(\theta)$ and magenta represents negative values.

the sphere $\mathbb{S}_{n-1}$. Considering the two-dimensional case $n = 2$, $p_{\mathbf{T}}(\theta)$ can be decomposed into the Fourier series (3.16)

$$p_{\mathbf{T}}(\theta) = \frac{1}{\sqrt{2\pi}}\, q_0 + \frac{1}{\sqrt{\pi}}\cos(2\theta)\, q_1 + \frac{1}{\sqrt{\pi}}\sin(2\theta)\, q_2 + \frac{1}{\sqrt{\pi}}\cos(4\theta)\, q_3 + \frac{1}{\sqrt{\pi}}\sin(4\theta)\, q_4,$$

where $\mathbf{q} = (q_0, \ldots, q_4)$ fully describes the polynomial and its corresponding tensor. These basis functions, shown in Figure 5, correspond to two-dimensional spherical harmonics and are sometimes called *circular harmonics*. The number of five is not surprising since the original tensor has five independent components: T_{1111}, T_{1112}, T_{1122}, T_{1222} and T_{2222}. The change of basis from $\mathbf{T}$ to $\mathbf{q}$ is done through a linear transformation, which can be derived by matching expressions (3.14) and (3.16).

Eigenvectors and eigenvalues of higher-order tensors, defined in (3.10), can be interpreted in the polynomial representation through the following property: $\mathbf{w}$ is an eigenvector of $\mathbf{T}$ with eigenvalue λ if and only if

$$\boldsymbol{\nabla} p_{\mathbf{T}}(\mathbf{w}) = 4\ \lambda\ \mathbf{w}. \tag{3.17}$$

In other words, the eigenvectors of $\mathbf{T}$ correspond to the stationary points of $p_{\mathbf{T}}(\theta)$.

Algebraic variety in the 2D case. A 2D fourth-order tensor is completely defined by a set of five coefficients $(q_0, \ldots, q_4)$. This tensor corresponds to a frame if the tensor is odeco, i.e., it lies on the algebraic variety defined by the set of quadratic equations (3.12). In the 2D case ($n = 2$), we can write out this set of equations and change the basis to $\mathbf{q}$, and derive the following set of linearly independent equations that define the variety:

$$\begin{cases} c_1(\mathbf{q}) = & q_0^2 - 18(q_3^2 + q_4^2) = 0 \\ c_2(\mathbf{q}) = & \sqrt{2} q_0 q_1 - 6 q_1 q_3 - 6 q_2 q_4 = 0 \\ c_3(\mathbf{q}) = & \sqrt{2} q_0 q_2 - 6 q_1 q_4 + 6 q_2 q_3 = 0 \end{cases} \tag{3.18}$$

Area of a frame. The optimization we perform on the frame field requires to normalize the integrability condition to make it independent of the size of the frame. We choose to do this normalization with the *area* of the frame, i.e., the product of the sizes $a = \lambda\mu$. This can be interpreted as the local area of the quad elements in the frame's neighborhood. One can show that it can be expressed only in terms of the circular harmonics coefficients $\mathbf{q}$:

$$a(\mathbf{q}) = \lambda\mu = \frac{1}{\pi}\left(\frac{8}{9} q_0^2 - (q_1^2 + q_2^2)\right). \tag{3.19}$$

We show later in Section 5 how this expression is used to normalize the integrability condition.

Isotropic case. A frame is *isotropic*, i.e., its vectors have equal length $\lambda = \mu$, if its degree 2 coefficients are zero: $q_1 = q_2 = 0$. The algebraic variety reduces to a single quadratic equation $c_1(\mathbf{q}) = 0$. When the degree 2 coefficients are zero, the tensor eigenvectors are always orthogonal, even if the tensor is not odeco. A range of isotropic tensors is illustrated on Figure 6. We see that in the general non-odeco case, this class has an additional degree of freedom depending on how close it is to a sphere. We explain in section 7 how this extra degree of freedom is beneficial as it allows to represent singularities.

Figure 6: Range of isotropic tensors, the middle one being odeco. All tensors have a set of 4 eigenvectors with the same eigenvalues.

4 Integrability condition

Since there is a one-to-one correspondence between frames (as defined in (3.8)) and orthogonally decomposable (odeco) tensors with positive eigenvalues (as de-

fined in (3.13)), one can express the integrability condition (2.7) solely in terms of the odeco tensor coefficients and their spatial derivatives. In this section we show how we derive the closed-form expression for the Lie bracket.

Consider a frame $\mathbf{F} = \{\mathbf{R}^k(\mathbf{u}, \mathbf{v}),\ k = 0, 1, 2, 3\}$, where $\mathbf{R}$ performs a 90° rotation as defined in (2.1) and illustrated in Figure 3. Every pair of vectors in $\mathbf{F}$ has the same Lie bracket, which we denote Lie($\mathbf{F}$):

$$\text{Lie}(\mathbf{F}) = [\mathbf{R}^k\mathbf{u}, \mathbf{R}^k\mathbf{v}], \quad k = 0, 1, 2, 3. \tag{4.20}$$

First we write the Lie bracket $[\mathbf{u}, \mathbf{v}]$ in index notation:

$$[\mathbf{u}, \mathbf{v}]_i = u_\alpha \frac{\partial v_i}{\partial x_\alpha} - v_\alpha \frac{\partial u_i}{\partial x_\alpha}. \tag{4.21}$$

Let $\mathbf{T}$ denote the odeco tensor corresponding to $\mathbf{F}$. We apply the chain rule to express the spatial derivatives on the tensor coefficients:

$$[\mathbf{u}, \mathbf{v}]_i = u_\alpha \frac{\partial v_i}{\partial T_{j_1 j_2 j_3 j_4}} \frac{\partial T_{j_1 j_2 j_3 j_4}}{\partial x_\alpha} - v_\alpha \frac{\partial u_i}{\partial T_{j_1 j_2 j_3 j_4}} \frac{\partial T_{j_1 j_2 j_3 j_4}}{\partial x_\alpha}. \tag{4.22}$$

In the remaining of the section we show how (a) the sensitivity terms $\partial v_i / \partial T_j$ and $\partial u_i / \partial T_j$ are derived, and (b) how the Lie bracket is completely expressed in terms of the tensor coefficients $\mathbf{T}$.

Eigenvalue sensitivity for tensors. Recall that $\mathbf{u}, \mathbf{v}$ are eigenvectors of $\mathbf{T}$ as defined by (3.10). Finding the sensitivity terms $\partial v_i / \partial T_j$ and $\partial u_i / \partial T_j$ can be done by analyzing how the eigenvectors $\mathbf{u}, \mathbf{v}$ change when adding an infinitesimal perturbation $\delta\mathbf{T}$ to the tensor. This analysis is well-known for matrices as the *eigenvalue perturbation problem*, but can be generalized to higher-order tensors, provided they are odeco. We leave the complete derivation in appendix A and retain the main result: if $(\lambda, \hat{\mathbf{w}})$ is an eigenpair of odeco tensor $\mathbf{T}$, then the sensitivity of $\mathbf{w} = \lambda\hat{\mathbf{w}}$ with respect to a perturbation $\delta\mathbf{T}$ is given by

$$\delta\mathbf{w} = (\delta\mathbf{T})\hat{\mathbf{w}}^3. \tag{4.23}$$

From this result we find that the partial derivatives are given by

$$\frac{\partial w_i}{\partial T_{j_1 j_2 j_3 j_4}} = \delta_{ij_1} \hat{w}_{j_2} \hat{w}_{j_3} \hat{w}_{j_4}, \tag{4.24}$$

where δ_{ij} is the Kronecker delta symbol. Note that an eigenvector's sensitivity only depends on itself and not on the other eigenvectors of the odeco tensor.

Lie bracket in terms of tensor representation. We now wish to find an expression for the Lie bracket (4.22) so that it only depends on the tensor coefficients $\mathbf{T}$ and not on the frame directions $\mathbf{u}, \mathbf{v}$. Plugging in the sensitivity result (4.24) yields

$$[\mathbf{u}, \mathbf{v}]_i = (\|\mathbf{u}\| \hat{u}_\alpha \hat{v}_{j_2} \hat{v}_{j_3} \hat{v}_{j_4} - \|\mathbf{v}\| \hat{v}_k \hat{u}_{j_2} \hat{u}_{j_3} \hat{u}_{j_4}) \frac{\partial T_{ij_2j_3j_4}}{\partial x_\alpha}. \tag{4.25}$$

We see that an expression close to the odeco tensor definition (3.13) starts to appear. The last ingredient is to note that $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$ are related by a 90° rotation, which can be written an $\hat{v}_i = \epsilon_{ij}\hat{u}_j$ and $\hat{u}_i = -\epsilon_{ij}\hat{v}_j$, where ϵ_{ij} is the two-dimensional Levi-Civita symbol. Plugging this into the Lie bracket expression and identifying the tensor coefficients yields

$$\text{Lie}(\mathbf{T})_i = \epsilon_{j_2k_2}\epsilon_{j_3k_3}\epsilon_{j_4k_4} T_{\alpha k_2 k_3 k_4} \frac{\partial T_{ij_2j_3j_4}}{\partial x_\alpha}. \tag{4.26}$$

As the tensor coefficients $\mathbf{T}$ are a linear transformation of the polynomial coefficients $\mathbf{q}$, one can express the Lie bracket in terms of $\mathbf{q}$ only:

$$\text{Lie}(\mathbf{q})_i = C_{ijk\alpha}\ q_k\ \frac{\partial q_j}{\partial x_\alpha}, \tag{4.27}$$

where the $C_{ijk\alpha}$ are known coefficients. This expression is a key result of this work as it allows to compute an integrability metric that does not involve the frame vectors, but instead a quadratic function of the implicit tensor representation.

5 Optimization

We design an optimization formulation that aims at making the Lie bracket (4.27) as close to zero as possible, with odeco tensors (3.18) having positive sizes $(\lambda, \mu > 0)$. To do so we define the following energy functionals on the domain Ω:

$$E_{\text{Lie}}[\mathbf{q}] = \int_\Omega \frac{\|\text{Lie}(\mathbf{q})\|^2}{a(\mathbf{q})^2}\ \mathrm{d}\mathbf{x}\,, \tag{5.28}$$

$$E_{\text{odeco}}[\mathbf{q}] = \sum_{i=1}^{3} \int_\Omega c_i(\mathbf{q})^2\ \mathrm{d}\mathbf{x}, \tag{5.29}$$

$$E_{\text{D}}[\mathbf{q}] = \sum_{j=1}^{5} \frac{1}{2} \int_\Omega \|\boldsymbol{\nabla} q_j\|^2\ \mathrm{d}\mathbf{x}\,. \tag{5.30}$$

The normalized Lie bracket energy E_{Lie} aims at making the frame field integrable, the odeco penalty E_{odeco} at keeping the tensors odeco, and the Dirichlet energy E_{D} strives for smoothness of the frame field (Palmer et al. [16] have shown that, since the L^2 distance on $\mathbf{q}$ corresponds to the L^2 distance between the polynomials $p_{\mathbf{T}}$, the Dirichlet energy is an adequate proxy for the smoothness of the frame field). The area expression $a(\mathbf{q})$ in the denominator of the Lie bracket energy acts

as a normalization: E_{Lie} does not depend on the global scale of the frame field, i.e., for a constant factor α, $E_{\text{Lie}}(\alpha \mathbf{q}) = E_{\text{Lie}}(\mathbf{q})$. It also acts as a barrier, preventing the frame sizes λ and μ from becoming zero or negative, provided the initial solution has positive sizes. The Lie bracket energy and the odeco penalty are assembled in a Ginzburg-Landau-like functional, and the Dirichlet energy acts as a regularizer:

(5.31)
$$\min_{\mathbf{q}} E_\kappa[\mathbf{q}] = (1-\kappa)\ E_{\text{Lie}}[\mathbf{q}] + \kappa\ E_{\text{D}}[\mathbf{q}] + \frac{1}{\epsilon^2} E_{\text{odeco}}[\mathbf{q}].$$

Parameter ϵ has units of length and, like in the Ginzburg-Landau functional, controls the size of the singularity neighborhoods where the frames drift away from the odeco variety. A schedule is set on κ such that it starts at 1, providing a smooth (but not integrable) frame field, and is gradually reduced to 0 to enforce integrability. As there is an infinite space of integrable solutions, we found that the Dirichlet regularization prevents the solver from drifting too far away from smooth solutions.

Discretization. The frame field is discretized on a standard triangle mesh over Ω. At each node we store the five circular harmonics coefficients $\mathbf{q} = (q_0, \dots, q_4)$. The coefficients are then linearly interpolated on the mesh using continuous P1 triangular finite elements. The energy functionals are then evaluated using a standard 3-point quadrature rule on the triangles. Note that we are using a continuous approximation for the frame field even though the actual field is not defined at the singularities and therefore cannot be represented by our discretization. This justifies our weak enforcement of the odeco constraint through a penalty term: it allows the creation of singularities in a way that does not blow up the integrability energy E_{Lie}, by using tensors which have no preferential directions, i.e., a ball as in Figure 5, left.

Behavior of E_{Lie} at singularities. Since we are using a continuous representation for the frame field, a legitimate concern is whether the Lie bracket energy E_{Lie} remains bounded around singularities. To analyze this we compute an exactly integrable frame field around a singularity of index i; this frame field is given in polar coordinates by

(5.32)
$$\mathbf{u}(r,\theta) = r^i\ (\cos((1-i)\theta)\ \hat{\mathbf{e}}_r - \sin((1-i)\theta)\ \hat{\mathbf{e}}_\theta),$$
$$\mathbf{v}(r,\theta) = r^i\ (\sin((1-i)\theta)\ \hat{\mathbf{e}}_r + \cos((1-i)\theta)\ \hat{\mathbf{e}}_\theta).$$

Notice that the size r^i vanishes at the singularity for $i > 0$ (valence 3 and less) and blows up for $i < 0$ (valence 5 and more). One can check that this frame field indeed has zero Lie bracket. We represent this frame field on finite element meshes of varying sizes and evaluate both the total energy E_{Lie} as well as the maximum value of the integrand; this is shown on Figure 7. We see that even though the value of the integrand blows up when refining the mesh, the energy remains bounded and converges to a fixed value for both singularities. The energy is higher for a valence 5 singularity than a valence 3 since the size is unbounded. This property makes it possible for our solver to introduce singularities whatever the mesh size, if they make the frame field more integrable.

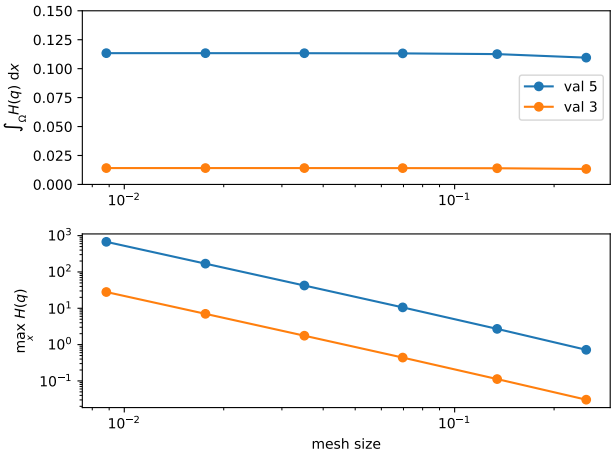

Figure 7: Convergence of the Lie bracket energy E_{Lie} around singularities of valence 5 and 3. $H(\mathbf{q})$ denotes the integrand of E_{Lie}: $H(\mathbf{q}) = \|\text{Lie}(\mathbf{q})\|^2 / a(\mathbf{q})^2$.

Recovering a frame field. The minimization of (5.31) provides a field of tensors that is odeco everywhere except in the vicinity of singularities. We recover a frame at every node of the mesh in two steps: first, the tensors are projected onto the odeco variety using the projector of [16], using a semidefinite relaxation of the exact projection problem. This operation results in a set of frames of which we only keep the direction vectors $\hat{\mathbf{u}}, \hat{\mathbf{v}}$. In a second step, the sizes of the direction vectors are computed by contracting the tensor 4 times, as if they were eigenvectors:

(5.33)
$$\lambda = \mathbf{T}\hat{\mathbf{u}}^4, \quad \mu = \mathbf{T}\hat{\mathbf{v}}^4.$$

We found that this approach for recovering sizes provided better results than using the sizes given by the projector; this is because the Lie bracket is expressed in terms of the eigenvectors and eigenvalues of the odeco tensor field.

The optimization procedure to compute an integrable frame field is summarized in Algorithm 1.

Algorithm 1 Integrable frame field solver

Input: A triangle mesh on a planar geometry, with size and/or orientation constraints on boundaries
Output: A smooth integrable frame field

On boundaries, assign frames respecting constraints.
In interior, assign zero frames.
Compute $\mathbf{T}$ and $\mathbf{q}$ using (3.13).
for κ in $[1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 0]$ **do**
  $\mathbf{q} \leftarrow \text{argmin}_{\mathbf{q}} E_\kappa[\mathbf{q}]$ using L-BFGS. (5.31)
end for
for each node **do**
  Recover directions $\hat{\mathbf{u}}, \hat{\mathbf{v}}$ using projection of [16].
  Recover frame sizes λ, μ using (5.33).
end for

6 Frame field guided parametrization and meshing

We briefly review how a seamless parametrization is computed from the frame field $\mathbf{F}(\mathbf{x})$ obtained through the optimization of (5.31); for a more detailed treatment we refer to, e.g., [2].

We compute a field-guided seamless parametrization with the following steps:

Singularity detection. Singular triangles are identified by calculating the turning number of the frames around a given triangle; this number is $+1/4$ or $-1/4$ for a singularity of valence 3 and 5, respectively.

Cut graph. The triangle mesh is "cut open" such that a closed curve cannot turn around a singularity without crossing the cut graph. The singular triangles also belong to the cut graph (the parametrization is not defined on them).

Frame vectors assignment. Thanks to the cut graph, two vectors $\mathbf{u}$ and $\mathbf{v}$ can be chosen from each frame to form two continuous vector fields $\mathbf{u}(\mathbf{x})$ and $\mathbf{v}(\mathbf{x})$.

Integration. The vector fields are integrated to scalar potentials $u(\mathbf{x})$ and $v(\mathbf{x})$ by solving the least-squares problem

(6.34)
$$\min_{\substack{u(\mathbf{x}) \\ v(\mathbf{x})}} \int_\Omega \left(\left\| \boldsymbol{\nabla} u - \frac{\mathbf{u}}{\|\mathbf{u}\|^2} \right\|^2 + \left\| \boldsymbol{\nabla} v - \frac{\mathbf{v}}{\|\mathbf{v}\|^2} \right\|^2 \right) \mathrm{d}\mathbf{x}\,.$$

During this optimization, the parametrization is constrained to be seamless, meaning that (i) on the boundary, one of the potentials u or v is constant, and (ii) along each cut, the matching potential gradients are equal (which amounts to impose (2.3)). These seamlessness constraints can be written as linear constraint on the unknowns which are straightforward to impose.

We call the residual of the optimization objective (6.34) the *integration error*; it quantifies how integrable the frame field was, and in general how well the parametrization aligns to the prescribed frame field. Achieving a small integration error means we can adequately control the size and orientation constraints in the seamless parametrization process.

Different approaches, reviewed earlier in section 1, exist to generate a fully quadrilateral mesh from a seamless parametrization. In this work we use the quasi-structured quadrilateral mesher of [19], which consists in a frontal point insertion coupled with cavity remeshing and mesh smoothing.

7 Results and Discussion

To demonstrate the validity of our method, we compare smooth frame fields against integrable frame fields for a set of geometries where the frame orientations and sizes are fixed on the boundaries. For each frame field we compute a parametrization as described in the previous section, and measure the integration error. For the integrable frame field we also generate a quadrilateral mesh using the previously described frontal mesher.

Parameter settings. For each test case we use a fixed odeco parameter ϵ that has the order of magnitude of the maximum mesh size. The initial solution of the optimization is a smooth field of tensors that are not constrained to be odeco, i.e., $\kappa = 1$ and $\epsilon = \infty$. Then a schedule is started on the reguarization parameter κ, using the following sequence: $(1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 0)$. The first iteration ($\kappa = 1$) corresponds to a pure Ginzburg-Landau functional (this is the solution we use for smooth frame fields), and at the last iteration ($\kappa = 0$) the regularizer is removed to obtain the most integrable frame field. To solve each optimization problem (5.31) we use a standard quasi-Newton L-BFGS solver. The integral calculations are done in parallel over the triangles. For the following results, the triangular meshes have between 1000 and 4000 nodes. All results have been obtained in less than 3 minutes on a 2 GHz Intel Core i5 CPU on 4 threads.

Comparison. The results of this comparison are illustrated in Figure 8. On average, the integrable frame field achieves an integration error 8.6 times smaller than the smooth frame field. The smooth frame fields do not have the correct singularity configuration to perform the required size transition. This causes the corresponding

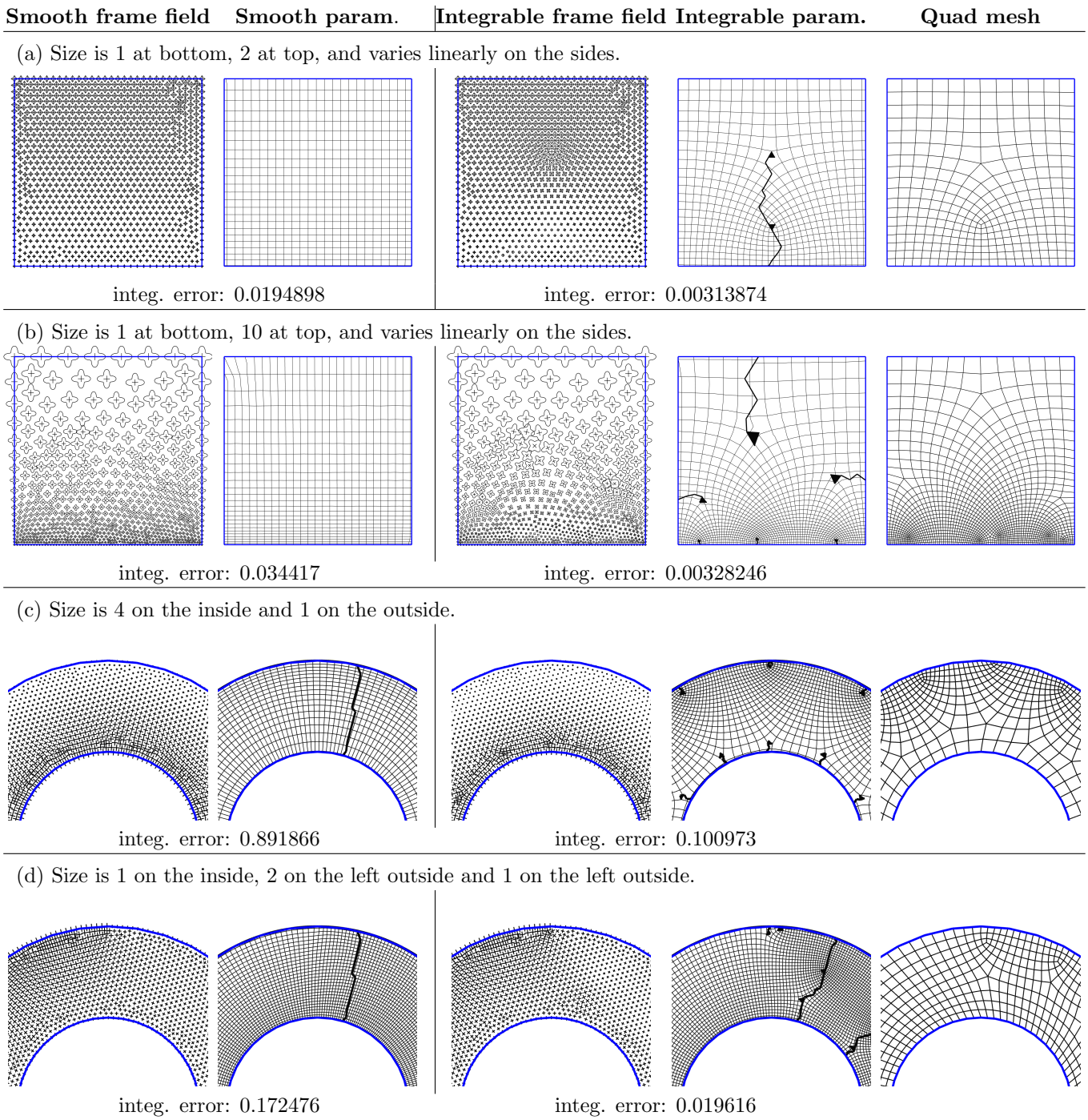

Figure 8: Comparison of smooth frame fields and integrable frame fields when sizes are imposed on boundaries. For each frame field the corresponding parametrization is illustrated, and a quad mesh is computed using guidance from the integrable frame field. All frame fields are isotropic. For cases (a) and (b) we also illustrate the polynomials $p_{\mathbf{T}}(\theta)$ corresponding to the frames.

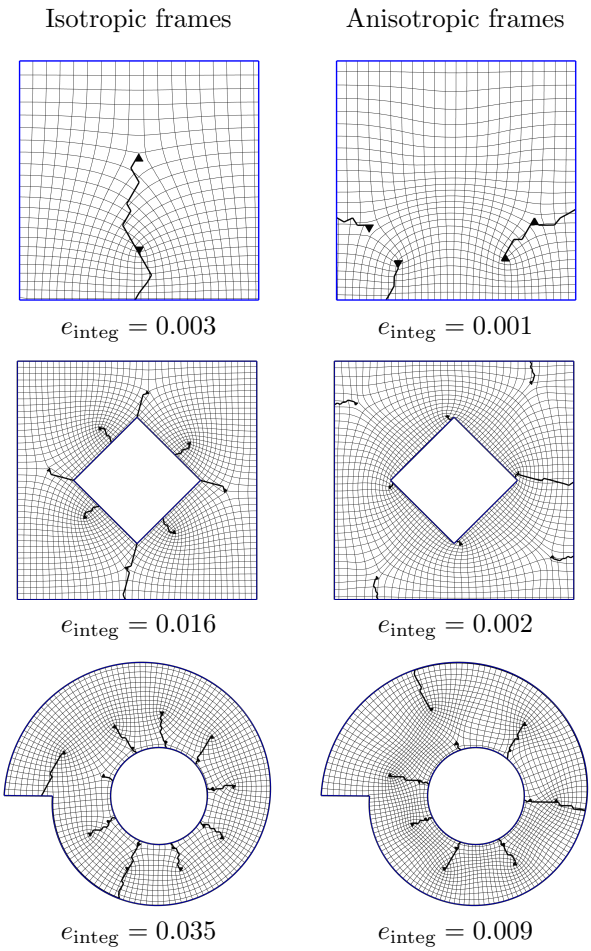

Figure 9: Seamless parametrizations computed for various models, including the *nautilus* (bottom) in the isotropic and anisotropic settings. e_{integ} denotes the integration error. Boundary size conditions are
(top) 1 at bottom and 2 at top,
(middle) 1 on inner boundary and 2 on outer boundary,
(bottom) 1 on inner and outer boundaries.

parametrizations to be very anisotropic, even though the frames are fully isotropic. On the other hand, the integrable frame fields have a set of singularities that ensure the size transitions, and this is reflected in the parametrizations where the isolines remain globally isotropic and have the correct sizes at the boundaries. We also see on testcase (d) that the solver is able to represent abrupt size transitions through a tight pair of singularities.

Singularities. These results also exhibit how the field of tensors behaves in the vicinity of singularities. As expected, the tensors become non-odeco as they narrow the singularities; this allows the solver to represent singularities in a way that is not too costly in terms of integrability energy. However it also brings a limitation: as the tensors are not odeco, the assumptions needed for the integrability energy fail, causing the frame field to not be exactly integrable close to singularities. In practice, one needs to tune parameter ϵ to achieve an appropriate trade-off between singularity cost and integrability.

Isotropic vs anisotropic frames. Figure 9 compares the isotropic and anisotropic frame fields obtained with our solvers on additional examples. The anisotropic solver produces a very distinct singularity configuration compared to the isotropic solver, and achieves a much smaller integration error thanks to the additional degrees of freedom.

Limit cycles. Conventional frame field-driven quad meshing methods fail due to the presence of *limit cycles* in the frame field, which are the consequence of a non-meshable topology. A standard test case where a limit cycle appears is the *nautilus* model, for which we show our results on Figure 9, bottom. Our method successfully produces singularity configurations the are free of limit cycles, and a valid quad mesh can be extracted.

Existence of integrable frame fields. Given a set of boundary conditions, an exactly integrable frame field does not exist in general; in our case, by imposing sizes strongly on the boundaries, an integrable isotropic frame field very often does not exist. We refer to [9] and the Abel-Jacobi framework for a complete treatment of these aspects. The non-existence of an integrable frame field can explain why a Lie bracket of zero is not achievable. Moreover, the smaller integration error achieved by allowing anisotropic frames (as illustrated in Figure 9) shows that the boundary conditions truly obstruct the integrability. This can be alleviated either by relaxing the boundary conditions or the isotropy constraint.

8 Conclusion and Future Work

In this work, we have shown how to leverage the theory of orthogonally decomposable tensors to produce integrable frame fields. We have studied the tensor eigenvalue perturbation problem and found simple expressions for the eigenvalue sensitivity of tensors. This result enables us to write a frame field's Lie bracket (and thus, the integrability optimization problem) solely in terms of its tensor representation. The integrable frame field can be integrated to a seamless parametrization of which we have full control over size and orientation through the frame field optimization; this is a feature that is often overlooked in field-based meshing methods. We believe this contribution is an important step towards a robust, flexible (in terms of size and orientation prescriptions) and optimal quadrilateral mesher.

Integrable non-odeco fields. An important limitation of our approach is that the tensor field is assumed to be odeco for the Lie bracket expression to be valid. This causes the frame field to not be exactly integrable near singularities. A possible solution for this is to remove the odeco assumption when writing out the eigenvalue perturbation problem. This makes it possible to write a Lie bracket that remains valid even if the tensor field is non-odeco (at least in the isotropic case). The price to pay is that the integrability condition becomes a more complex, rational expression of the tensor coefficients. We leave the investigation of this approach for future work.

The 3D case. The major advantage of our methodology is that it offers a natural extension to compute 3D integrable frame fields; indeed, the theory of odeco tensors, as well as the results on eigenvalue sensitivity, remain valid in arbitrary dimensions. The only obstacle remaining is to express the integrability condition in terms of the algebraic frame representation We are confident that a solution can be found in future work.

Acknowledgments

Mattéo Couplet is a Fellow of the Belgian Fund for Scientific Research (F.R.S.-FNRS).

Appendix A Eigenvalue sensitivity for odeco tensors

Let $\mathbf{T}$ be a fully symmetric fourth-order orthogonally decomposable tensor of dimension n, i.e., $\mathbf{T} = \sum_{i=1}^{n} \lambda_i \hat{\mathbf{v}}_i^4$, or, in index notation,

$$T_{j_1 j_2 j_3 j_4} = \lambda_i \hat{v}_{i,j_1} \hat{v}_{i,j_2} \hat{v}_{i,j_3} \hat{v}_{i,j_4}, \tag{1.35}$$

with orthogonal unit vectors $\hat{\mathbf{v}}_i$. Consider the contraction of $\mathbf{T}$ with one of its eigenvectors, $\hat{\mathbf{v}}_k$:

$$\begin{aligned} T_{j_1 j_2 j_3 j_4} \hat{v}_{k,j_4} &= \lambda_i \hat{v}_{i,j_1} \hat{v}_{i,j_2} \hat{v}_{i,j_3} \hat{v}_{i,j_4} \hat{v}_{k,j_4}, \\ &= \lambda_i \hat{v}_{i,j_1} \hat{v}_{i,j_2} \hat{v}_{i,j_3} \delta_{ik} \\ &= \lambda_k \hat{v}_{k,j_1} \hat{v}_{k,j_2} \hat{v}_{k,j_3}, \end{aligned} \tag{1.36}$$

or, written compactly, $\mathbf{T}\hat{\mathbf{v}}_k = \lambda_k \hat{\mathbf{v}}_k^3$. Contracting again and following the same procedure we find that $\mathbf{T}\hat{\mathbf{v}}_k^2 = \lambda_k \hat{\mathbf{v}}_k^2$ and $\mathbf{T}\hat{\mathbf{v}}_k^3 = \lambda_k \hat{\mathbf{v}}_k$, the latter being the definition of a tensor eigenvector. Consider now a specific eigenpair $(\lambda, \mathbf{w})$, with $\mathbf{w}$ having unit norm. We wish to express the variation of the eigenpair $(\delta\lambda, \delta\mathbf{w})$ given some perturbation of the tensor $\delta\mathbf{T}$. We write the remainder of the proof in compact notation for brevity but the same steps can be performed in index notation. Since the eigenvectors have unit norm, any eigenvector is orthogonal to its variation:

$$(\mathbf{w} + \delta\mathbf{w}) \cdot (\mathbf{w} + \delta\mathbf{w}) = 1 \;\Rightarrow\; \mathbf{w} \cdot \delta\mathbf{w} = 0, \tag{1.37}$$

neglecting the higher-order terms $(\delta\mathbf{w} \cdot \delta\mathbf{w})$. Writing the eigenvector definition $\mathbf{T}\mathbf{w}^3 = \lambda\mathbf{w}$ for the new eigenpair, we find

$$\begin{aligned} (\mathbf{T} + \delta\mathbf{T})(\mathbf{w} + \delta\mathbf{w})^3 &= \lambda\mathbf{w} + \delta(\lambda\mathbf{w}) \\ \cancel{\mathbf{T}\mathbf{w}^3} + 3\mathbf{T}\mathbf{w}^2\delta\mathbf{w} + \delta\mathbf{T}\mathbf{w}^3 &= \cancel{\lambda\mathbf{w}} + \delta(\lambda\mathbf{w}) \\ \cancel{3\lambda\mathbf{w}^2\delta\mathbf{w}} + \delta\mathbf{T}\mathbf{w}^3 &= \delta(\lambda\mathbf{w}) \end{aligned} \tag{1.38}$$

We have used, respectively, the eigenvector definition, the contraction property $\mathbf{T}\mathbf{w}^2 = \lambda\mathbf{w}^2$ and the orthogonality property $\mathbf{w} \cdot \delta\mathbf{w} = 0$. This gives the desired sensitivity expression.

References

[1] Pierre-Alexandre Beaufort, Jonathan Lambrechts, François Henrotte, Christophe Geuzaine, and Jean-François Remacle. Computing cross fields A PDE approach based on the Ginzburg-Landau theory. *Procedia Engineering*, 203:219–231, 2017.

[2] David Bommes, Henrik Zimmer, and Leif Kobbelt. Mixed-integer quadrangulation. *ACM Transactions on Graphics*, 28(3):77:1–77:10, July 2009.

[3] Marcel Campen, David Bommes, and Leif Kobbelt. Quantized global parametrization. *ACM Transactions on Graphics*, 34(6):1–12, November 2015.

[4] Alexandre Chemin, François Henrotte, Jean-François Remacle, and Jean Van Schaftingen. Representing Three-Dimensional Cross Fields Using Fourth Order Tensors. In Xevi Roca and Adrien Loseille, editors, *27th International Meshing Roundtable*, volume 127, pages 89–108. Springer International Publishing, Cham, 2019.

[5] Keenan Crane, Mathieu Desbrun, and Peter Schröder. Trivial Connections on Discrete Surfaces. *Computer Graphics Forum*, 29(5):1525–1533, September 2010.

[6] Olga Diamanti, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. Designing N -PolyVector Fields with Complex Polynomials. *Computer Graphics Forum*, 33(5):1–11, August 2014.

[7] Olga Diamanti, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. Integrable PolyVector fields. *ACM Transactions on Graphics*, 34(4):1–12, July 2015.

[8] Jin Huang, Yiying Tong, Hongyu Wei, and Hujun Bao. Boundary aligned smooth 3D cross-frame field. *ACM Transactions on Graphics*, 30(6):1–8, December 2011.

[9] Jovana Jezdimirović, Alexandre Chemin, and Jean-François Remacle. Integrable cross-field generation based on imposed singularity configuration – the 2D manifold case –, September 2022.

[10] Felix Kälberer, Matthias Nieser, and Konrad Polthier. QuadCover - Surface Parameterization using Branched Coverings. *Computer Graphics Forum*, 26(3):375–384, 2007.

[11] Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. Globally optimal direction fields. *ACM Transactions on Graphics*, 32(4):1–10, July 2013.

[12] Nicolas Kowalski, Franck Ledoux, and Pascal Frey. A PDE Based Approach to Multidomain Partitioning and Quadrilateral Meshing. In Xiangmin Jiao and Jean-Christophe Weill, editors, *Proceedings of the 21st International Meshing Roundtable*, pages 137–154. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[13] M. Lyon, M. Campen, and L. Kobbelt. Quad Layouts via Constrained T-Mesh Quantization. *Computer Graphics Forum*, 40(2):305–314, May 2021.

[14] Ashish Myles, Nico Pietroni, and Denis Zorin. Robust field-aligned global parametrization. *ACM Transactions on Graphics*, 33(4):1–14, July 2014.

[15] Jonathan Palacios and Eugene Zhang. Rotational symmetry field design on surfaces. *ACM Transactions on Graphics*, 26(3):55, July 2007.

[16] David Palmer, David Bommes, and Justin Solomon. Algebraic Representations for Volumetric Frame Fields. *ACM Transactions on Graphics*, 39(2):1–17, April 2020.

[17] Nicolas Ray, Wan Chiu Li, Bruno Lévy, Alla Sheffer, and Pierre Alliez. Periodic global parameterization. *ACM Transactions on Graphics*, 25(4):1460–1485, October 2006.

[18] Nicolas Ray, Bruno Vallet, Wan Chiu Li, and Bruno Lévy. N-symmetry direction field design. *ACM Transactions on Graphics*, 27(2):1–13, April 2008.

[19] Maxence Reberol, Christos Georgiadis, and Jean-François Remacle. Quasi-structured quadrilateral meshing in Gmsh – a robust pipeline for complex CAD models. *arXiv:2103.04652 [cs]*, March 2021.

[20] Elina Robeva. Orthogonal Decomposition of Symmetric Tensors. *SIAM Journal on Matrix Analysis and Applications*, 37(1):86–102, January 2016.

[21] Andrew O. Sageman-Furnas, Albert Chern, Mirela Ben-Chen, and Amir Vaxman. Chebyshev nets from commuting PolyVector fields. *ACM Transactions on Graphics*, 38(6):1–16, December 2019.

[22] Amir Vaxman, Marcel Campen, Olga Diamanti, Daniele Panozzo, David Bommes, Klaus Hildebrandt, and Mirela Ben-Chen. Directional Field Synthesis, Design, and Processing. *Computer Graphics Forum*, 35(2):545–572, May 2016.

[23] Ryan Viertel and Braxton Osting. An Approach to Quad Meshing Based on Harmonic Cross-Valued Maps and the Ginzburg–Landau Theory. *SIAM Journal on Scientific Computing*, 41(1):A452–A479, January 2019.