Eurographics Symposium on Geometry Processing 2026
M. Skouras and M. Livesu
(Guest Editors)

Volume 45 (2026), Number 5

Surface Quadrilateral Meshing from Integrable Odeco Fields

M. Couplet[†1], A. Chemin[2], D. Bommes[3], E. Chien[1]

[1]Boston University, [2]Université catholique de Louvain, [3]University of Bern

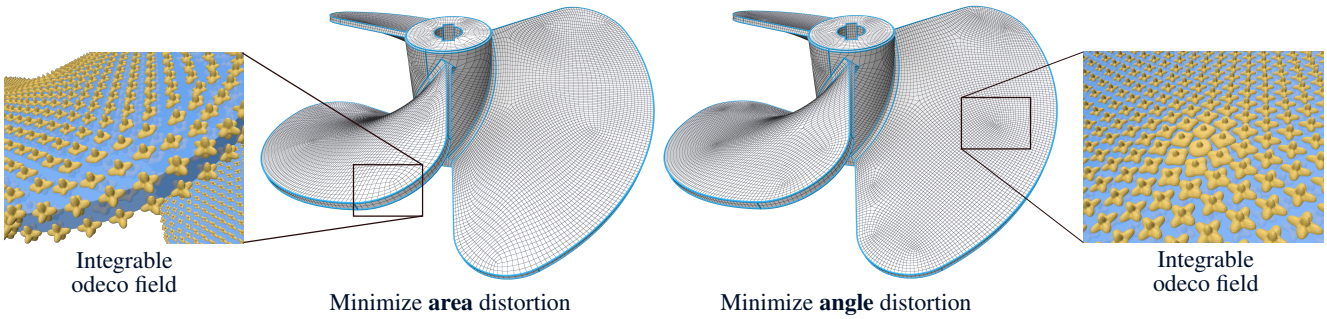


Figure 1: Our optimization framework utilizes normal-aligned 3D odeco tensors to produce integrable frame fields suitable for parameterization-based generation of anisotropic quad meshes. Our framework also accommodates area- and angle-distortion-miniziming energies. Our method jointly optimizes for singularity positions and integrability, allowing the frames to stray from odeco in the vicinity of naturally arising singularities.

Abstract

We present a method for generating orthogonal quadrilateral meshes subject to user-defined feature alignment and sizing constraints. The approach relies on computing integrable orthogonal frame fields, whose symmetries are implicitly represented using orthogonally decomposable (odeco) tensors. We extend the existing 2D odeco integrability formulation to the 3D setting, and define the useful energies in a finite element approach. Our frame fields are shear-free (orthogonal) by construction, and we provide terms to minimize area and/or angle distortion. The optimization naturally creates and places singularities to achieve integrability, obviating the need for user placement or greedy iterative methods. We validate the method on both smooth surfaces and feature-rich CAD models. Compared to previous works on integrable frame fields, we offer better performance in the presence of mesh sizing constraints and achieve lower distortion metrics.

1. Introduction

The problem of anisotropic quad meshing of curved surfaces has broad application and relevance to a wide number of use cases, e.g., FEM on rectangular meshes [ABF05], discrete nets and principal stress networks for physical construction [PAK07, PP18], and textile modelling via Chebyshev nets [SFCBCV19]. Meshes aligned to lines of principal curvature are also used to guide the creation of such meshes for the purposes of illustration and visualization [HZ00], and for subdivision modeling [Exo24].

Befitting such a central problem, there is an extensive literature aimed at generating semi-regular quad meshes in the presence of alignment and sizing constraints. The most popular sort of pipeline for producing such meshes involves two main steps: (1) a *seamless parameterization* is generated and (2) integer rounding and mesh extraction is performed to achieve the final mesh. For step (1), the classic approach is to first generate a smooth *cross field* and then modify this (unit-length) field to achieve an integrable result that can be used to produce a seamless parameterization. A key determination in this first step is the global topology of the quad mesh, consisting of mesh singularity placements. These placements are reflected in singularities of the initial cross field, whose placements may be suboptimal for a full seamless parameterization, in light of the fact that the field is non-integrable.

In our method, we present a new optimization formulation for step (1) that jointly optimizes for singularity positions and integra-

† Corresponding author

bility of an orthogonal *frame field* (with variable-length frame vectors) suitable for generation of a seamless parameterization. Such frame fields are represented with *orthogonally decomposable tensors* as introduced in [Rob16] and applied in [PBS20]. Our work extends the framework of [CCR26], which utilized 2D odeco tensors in the planar setting, to curved surfaces in 3D with the use of normal-aligned 3D odeco tensors. The 2D integrability energy present there is generalized to a 3D integrability energy that captures intrinsic curl on the 3D surface, and alignment and sizing of fields can again be imposed with simple linear constraints. As seen there, the combination of an integrability and "odeco-ness" energy allows for frame field singularities to arise naturally where violation of the latter allows for better global satisfaction of the former.

The use of normal-aligned odeco tensors also generalizes the use of normal-aligned octahedral frames in [ZVC*20] to generate cross fields on curved surfaces. The additional variables of frame vector lengths allow consideration of integrability, and our results also exhibit a natural alignment with sharp creases and principal curvature alignment in areas of high curvature. Lastly, we note that we also formulate effective area- and angle-preserving distortion energies within our tensor optimization framework.

To validate our approach, we run our method on a selection of feature-rich CAD models from the MAMBO dataset [Led20], as well as a handful of smooth models from the dataset of [MPZ14] (sans features). On the MAMBO models, we compare to the method of [DVPSH15], which is the only publicly available method we are aware of that also attempts to produce frame fields that are also integrable in a single optimization. Empirically, we find that our method leads to better global placement of singularities leading to more orthogonal quad meshes and better satisfaction of sizing constraints. We also compare to [CC25] on a small set of 3 challenging planar domains and one MAMBO model. Their base method assumes singularity placement as input, but we compare to their proposed automatic greedy insertion of singularities via iterative optimization. We find that our method achieves better distortion metrics and mesh symmetry at comparable singularity counts.

2. Related Work

The problems of quad meshing, frame field generation, and surface parameterization have been extensively studied, with an associated literature that is too large to comprehensively review here. Thus, we touch on the most relevant aspects and works below and refer readers to some excellent surveys on the topics [BLP*13, VCD*16, FH05, SPR07] for additional coverage.

Our method aims to generate integrable, orthogonal frame fields that integrate to locally injective seamless parameterizations. Moreover, the method handles both feature alignment and element sizing constraints, and jointly optimizes for both singularity placement and distortion measures suitable for generation of anisotropic quad meshes. Of the many works that consider frame/cross field design and seamless parameterization for quad meshing, most have some, but not all of these aspects. We restrict our discussion to methods that fall within the domain of training-free field-guided meshing, leaving out methods based on learning, e.g., [HRLL24], advancing fronts, e.g., [MGTG07], triangle merging, e.g., [RHCB*13], and principal curvature tracing, e.g., [ACSD*03], amongst others.

Singularity Placement. Most methods assume that singularities are given by those of an input cross field derived from smoothed principal curvature directions, e.g., [KNP07], or a field optimization that necessarily prioritizes smoothness over integrability (as the cross field is the same size everywhere), e.g., [BZK09]. Within the setting of conformal parameterization in the presence of cones (for generation of isotropic quad meshes), there have been many sophisticated works aimed at optimal cone placement to minimize area distortion and related isotropic distortion measures. Some are based on the use of Gaussian curvature as a signal [BCGB08], or via incremental flattening and concentration of this signal [MZ12, MZ13]. Others formulate more complicated optimizations, that are attacked with advanced optimization techniques like Fenchel-Rockafellar duality [SSC18] and Douglas-Rachford splitting [LFO*22]. Another recent method that incorporates singularity placement for cross fields is that of [PCS24], which selects smooth cross fields as minimal area sections of a circle bundle (but clearly does not incorporate an integrability criterion).

In contrast, our method achieves reasonable singularity placement while considering integrability and allowing for non-conformal distortion. In particular, our work allows for rectangular parameterizations, for which the coordinate axes get mapped to orthogonal directions. These were considered in the recent work [CC25] which does not jointly optimize for singularity positions. They do posit a greedy, iterative procedure reminiscent of [MZ12, MZ13] if automatic cone placement is desired. Also notable is the method of [DVPSH15], which achieves integrable PolyVector fields, and also allows for joint optimization of singularity positions and frame fields, though orthogonality of the frames is only enforced softly.

Integrability. As noted above, most methods optimize first for a non-integrable cross field and then modify the frames to achieve integrability through one of three approaches. Most common perhaps is the direct or indirect use of a Helmholtz-Hodge decomposition to extract the curl-free part of the initial field [KNP07, BZK09]. This is done implicitly in any method that utilizes a Poisson solve to find the parameterization that most closely matches the initial cross field. The second common approach is a rescaling of the cross field axes whether isotropic [RLL*06] or anisotropic [ZHLB10, SA24] that aims to achieve integrability. This method fixes the directionality of the input cross field and can lead to high distortion and inherit the poor singularity positioning of the initialization. Lastly, there are recent works that express integrability in other frameworks, like the moving frames setting [CC23, CC25] and PolyVector fields [DVPSH15, SFCBCV19]. Also notable is the work of [JCR24], which achieves integrable frame fields on surfaces, but is also constrained to the singularities of an input cross field.

Orthogonality and Distortion Control. There is a large body of work on sophisticated methods for optimizing distortion measures that favor isotropic (conformal) distortion, e.g., [SPSH*17, RPPSH17, CLW16], but there are few that use distortion measures that allow for anisotropic rectangular distortions. One exception to this is [Lev23b], which formulates a bespoke shear energy, but thus only enforces orthogonality in a soft fashion. Similarly, [DVPSH15] motivate orthogonality via a parameter $s \in [0, 1]$. The only method which maintains orthogonality in a strict fashion

is the work of [CC25], which satisfies it by construction in their optimization formulation. Our method achieves orthogonality via an odeco energy, which measures tensor deviation from the odeco variety, where frames are guaranteed to be orthogonal. Empirically, we achieve better orthogonality than [DVPSH15], and the relaxing of this hard orthogonality constraint allows for the joint optimization of singularity positions.

Feature Alignment and Sizing Control. Many works allow for hard or soft feature alignment via an aligned input field, but few allow for the capability to explicitly constrain sizing on boundary constraints. Some notable exceptions are the wave-based [ZHLB10], which only enforce the sizing softly, and metric modification approaches [KMZ10, JFH*15], which do not incorporate integrability criteria into their frame field generation.

Local Injectivity. Many works also focus on criteria for and guarantees of local injectivity through conformal [GSC21, CCS*21, FSZ*21, CZ24, CSH*25], harmonic [Flo03, BCW17], or more general mappings [Lip12, GKK*21, DKZ*22]. The vast majority of these do not target local injectivity of rectangular anisotropic mappings. There are also several notable works that focus on topological guarantees of local injectivity via explicit construction of locally-injective seamless maps [SZC*22, CSZZ19, Lev23a]. However, these all assume specified singularity positions, and usually are wildly distorted and non-orthogonal, relying on subsequent optimization to smooth the mappings.

At a continuous level, our method enforces positive lengths and integrability for our frames, so is guaranteed to be locally-injective. However, as with [CC25], it may fail to be injective at a discrete level, and so we use a Poisson method to recover a mapping that is mostly injective. As there, one could also consider giving our frames to a robust parameterization method that can maintain local injectivity, e.g., [MPZ14].

3. Background

3.1. Seamless parametrization

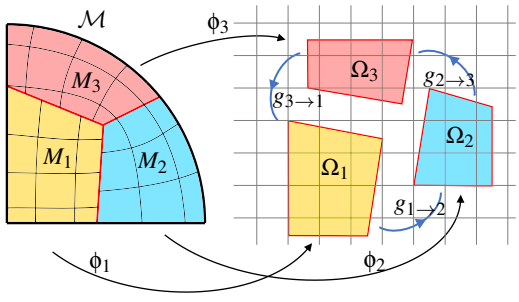


Figure 2: Domain $\mathcal{M}$ is mapped onto a parametric plane by a seamless parametrization ϕ, allowing to map a grid back onto $\mathcal{M}$.

Let $\mathcal{M} \subset \mathbb{R}^3$ be a two-dimensional manifold representing the surface to be meshed. A *parametrization* $\phi : \mathcal{M} \to \mathbb{R}^2$ maps the surface to a Cartesian parametric plane; we sometimes write it as a pair of maps (ϕ^u, ϕ^v). By pulling back the integer grid $(\mathbb{R}\times\mathbb{Z}) \cup (\mathbb{Z}\times\mathbb{R})$ from the parametric plane to the surface, one can locally tesselate the surface into quadrilateral elements. Because of the topology of the surface or the presence of desired cone points, it is usually impossible to generate a globally continuous parametrization, and instead one defines a collection of *charts* $\{\phi_i\}$ homeomorphically mapping a partition $M_i \subset \mathcal{M}$ to subsets $\Omega_i \subset \mathbb{R}^2$ of the parametric plane. These maps need to respect a set of conditions to be appropriate for quadrilateral meshing purposes. First, each chart must be *locally injective* to avoid any inverted elements when pulling back the integer grid onto the surface. For this property to hold, the Jacobian matrix $\mathbf{J}_{\phi_i}$ must have positive determinant:

$$\det \mathbf{J}_\phi(p) = \det \begin{pmatrix} \nabla\phi^u \\ \nabla\phi^v \end{pmatrix} > 0 \quad \text{for } p \in \mathcal{M}. \tag{1}$$

ϕ^u and ϕ^v are the surface gradients of the respective scalar fields u and v. Charts are glued together by *transition functions*: if ϕ_i and ϕ_j are two charts, then on the intersection of their domains $M_i \cap M_j$ (a one-dimensional curve) there exists another homeomorphism $g_{i\to j} : \phi_i(M_i \cap M_j) \to \phi_j(M_i \cap M_j)$ with $g_{i\to j} = \phi_j \circ \phi_i^{-1}$. In order for the mesh to seamlessly stitch between charts, the transition functions need to preserve the Cartesian grid, which yields the *comformity* condition

$$g_{i\to j}(z) = \mathbf{R}_{i\to j} z + \boldsymbol{\tau}_{i\to j} \quad \text{for } z \in \Omega_i \cap \Omega_j, \tag{2}$$

where $\mathbf{R}_{i\to j}$ is any rotation multiple of $\pi/2$ – this multiple is denoted $r_{i\to j}$ and is often called a *matching* – and $\boldsymbol{\tau}_{i\to j} \in \mathbb{R}^2$ is a parametric translation, i.e., the offset in parameter space when transitioning from chart i to j.

In the neighborhood of a point p, if there is a counterclockwise cycle of charts $\phi_1, \phi_2, \ldots, \phi_N = \phi_1$ about p (as in Fig. 2) for which the sum of rotation multiples $r_p^{\circlearrowleft} := r_{1\to2} + r_{2\to3} + \ldots + r_{(N-1)\to1}$ is nonzero, then p is a *singularity* of index $r_p^{\circlearrowleft}/4$. Upon quadrangulation such a singular point will become a valence $(4 - r_p^{\circlearrowleft})$ mesh vertex. The point in the center of Fig. 2 is of index $+1/4$ and would result in a valence 3 vertex.

A parametrization $\{\phi_i\}$ is said to be *seamless* if it is both locally injective and conforming, verifying Equations (1) and (2). In the presence of feature or boundary curves that the mesh should align to, we additionally ask that one of the parametric functions ϕ^u or ϕ^v be constant along the curve. This is imposed by ensuring that one of the parametrization gradients is orthogonal to the curve; if $\mathbf{e}_t$ is a vector tangent to the curve then

$$\nabla\phi^u \cdot \mathbf{e}_t = 0 \quad \text{or} \quad \nabla\phi^v \cdot \mathbf{e}_t = 0. \tag{3}$$

Lastly, for extraction of a quad mesh by pulling back the integer grid, we must require that the parameterization be *integer seamless*. In particular, all singular points p map to integer points, all feature curve points q map into the integer grid, and all parametric translations are integer:

$$\phi_i(p) \in \mathbb{Z}\times\mathbb{Z} \quad \text{and} \quad \phi_i(q) \in (\mathbb{R}\times\mathbb{Z}) \cup (\mathbb{Z}\times\mathbb{R}) \tag{4}$$

$$\text{and} \quad \tau_{i\to j} \in \mathbb{Z}^2. \tag{5}$$

3.2. Integrable frame fields

The conditions defining a seamless parametrization, Eqs. (1) to (3), can all be expressed as conditions on the parametrization Jaco-

bian $\mathbf{J}_\phi$. This is already the case for local injectivity and feature alignment. For conformity, on the chart boundary we have that $\phi_j = g_{i\to j} \circ \phi_i$, and taking the Jacobian on both sides gives

$$\mathbf{J}_{\phi_j}(p) = \mathbf{R}_{i\to j}\mathbf{J}_{\phi_i}(p), \quad \text{for } p \in M_i \cap M_j. \tag{6}$$

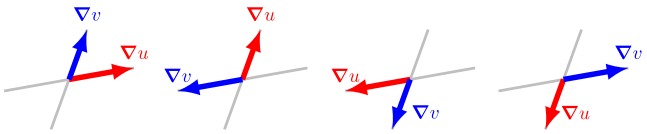


Figure 3: Equivalent gradient pairs $(\nabla u, \nabla v)$ across the charts of a global seamless parametrization.

Because $\mathbf{R}_{i\to j}$ is a rotation by a multiple of $\pi/2$, it acts on the rows of $\mathbf{J}_{\phi_i}$ via permutation (with some signs). Thus, the gradient vectors (rows of $\mathbf{J}_{\phi_i}$) are not actually rotated, but rather are permuted as shown in Fig. 3. The Jacobian is thus discontinuous across charts where it undergoes a rotation of a multiple of $\pi/2$. To remove this discontinuity, we can define the equivalence class of $\mathbf{J}_\phi$ under this group:

$$\left[\mathbf{J}_\phi\right] \coloneqq \left\{\mathbf{R}^k \mathbf{J}_\phi \mid k \in \mathbb{Z}\right\}, \tag{7}$$

where $\mathbf{R}$ performs a $\pi/2$ rotation. This equivalence class forms our notion of *frame*. The corresponding *frame field* $\left[\mathbf{J}_\phi\right](p)$ is then continuous on the whole parametrization, except at singularities (note that the discontinuity is at the singular point, as sizes of the frame vectors blow up when approaching singularities of valence 3). This observation justifies computing parametrizations through continuous frame fields, which do not require integer constraints or cutting the domain into charts. For a frame field to induce a parametrization, it must be *integrable*. On any simply-connected domain, a vector field is integrable to a scalar potential if it is *curl-free*. Let $\mathbf{F}(p)$ be a frame field as defined by the equivalence class in Eq. (7). At a nonsingular point p, a *lifting* of $\mathbf{F}$ in a neighborhood of p is a pair of continuous vector fields $\mathbf{u}(p), \mathbf{v}(p)$, each being part of frame $\mathbf{F}(p)$. Frame field $\mathbf{F}$ is curl-free at p if any lifting in a neighborhood $\mathcal{N}(p)$ is curl-free:

$$\nabla \times \mathbf{u}(p') = \nabla \times \mathbf{v}(p') = \mathbf{0}, \quad \text{for } p' \in \mathcal{N}(p). \tag{8}$$

If $\mathbf{F}$ is curl-free, it is thus the Jacobian of a parametrization ϕ_i on any simply-connected chart M_i. Stitching the charts together through the permutations of the frame representation, $\mathbf{F}$ is the Jacobian of a global seamless parametrization on the whole domain.

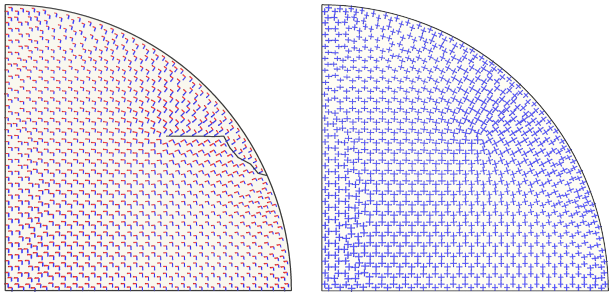


Figure 4: Gradients of a parametrization and corresponding integrable frame field.

3.3. Intrinsic vector field curl from extrinsic frames

Given a vector field $\mathbf{v}$ tangent to a smooth embedded surface $\mathcal{S} \subset \mathbb{R}^3$, it is important to distinguish the intrinsic scalar surface curl $\nabla_\mathcal{S} \times \mathbf{v}$ from the extrinsic 3D vorticity vector $\nabla \times \tilde{\mathbf{v}}$ (where $\tilde{\mathbf{v}}$ is an ambient extension of $\mathbf{v}$ defined in some neighborhood of $\mathcal{S}$). If $\mathbf{n}$ is the surface unit normal, these quantities are related by

$$\nabla_\mathcal{S} \times \mathbf{v} = \mathbf{n} \cdot (\nabla \times \tilde{\mathbf{v}}). \tag{9}$$

This result requires that $\mathbf{v} \cdot \mathbf{n} = 0$ ($\mathbf{v}$ is a tangent vector field) and is independent of the particular extension $\tilde{\mathbf{v}}$. This fundamental fact tells us that the normal component of the vorticity is an intrinsic quantity, and is discussed in many classic references, e.g., [AMR88, Fra11]. Thus, we say that $\mathbf{v}$ is curl-free and integrable on the surface if its 3D vorticity is orthogonal to the surface normal $\mathbf{n}$.

We also note here that our use of an extrinsic representation via normal-aligned 3D odeco tensors, as described in Sections 3.4 and 3.5.2, takes inspiration from the works of [JTPSH15, ZVC*20], both of which use a similar extrinsic frame representation for curved surfaces. As noted in both of these works, the use of such an extrinsic representation allows for natural alignment to sharp creases and principal curvatures in regions of high sectional curvature. Examples of this may be seen in Figure 5 and the last two models of Figure 7 (e.g., see arm regions). Another advantage of our extrinsic formulation is that surface curl is computed from 3D curl directly, as shown later in Section 4.1. We obviate the need to connect tangent spaces and explicitly account for surface curvature, making the formulation more straightforward.

3.4. Odeco representation

Orthogonally decomposable (*odeco*) tensors have been applied in [PBS20] as a representation of three-dimensional orthogonal frames. We review their definition and some of their important properties, before presenting in Section 4 how they are exploited in this work.

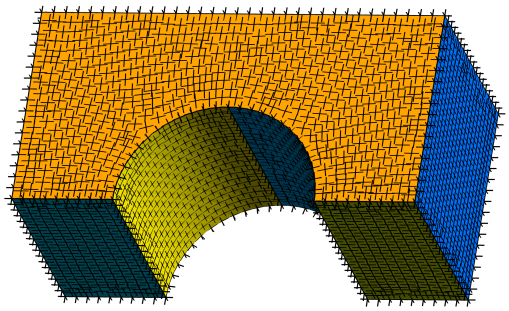


Figure 5: Even with *no explicit feature alignment constraints*, we still achieve natural frame alignment to sharp creases and principal curvature directions in this result on the B0 model from the MAMBO dataset [Led20].

When looking for a representation of an n-dimensional *orthogonal* frame $(\mathbf{v}_1, \ldots, \mathbf{v}_n)$ that is invariant under permutation and sign changes, it is tempting to simply consider the n-by-n symmetric positive definite matrix

$$\mathbf{P} \coloneqq \sum_m \lambda_m \mathbf{u}_m \mathbf{u}_m^\mathsf{T}, \tag{10}$$

where $\lambda_m \coloneqq \|\mathbf{v}_m\|$ and $\mathbf{u}_m \coloneqq \mathbf{v}_m/\lambda_m$ is the normalized frame vector. This choice is problematic when some frame vectors have identical lengths, as they form an eigenspace of $\mathbf{P}$ of dimension greater than 1 and the information on their individual orientations is lost. This problem is obviated by turning to a fourth-order tensor representation

$$\mathbf{T} \coloneqq \sum_{m=1}^{n} \lambda_m \mathbf{u}_m^{\otimes 4}. \tag{11}$$

An even order is necessary to ensure invariance under sign changes in the vectors. This tensor is *fully symmetric*, i.e., T_{ijkl} is invariant under permutation of its 4 indices. The notion of eigenvalue and eigenvector can be generalized to higher-order tensors as done in [Lim05, Qi07]. A unit vector $\mathbf{u} \in \mathbb{R}^n$ is an eigenvector of $\mathbf{T}$ with eigenvalue λ if contracting the tensor three times with $\mathbf{u}$ results in a scalar multiple:

$$\mathbf{T} \cdot \mathbf{u}^3 = \lambda \mathbf{u}. \tag{12}$$

Hence, the frame vectors and their magnitudes can be interpreted as eigenpairs of their odeco tensor.

Note that these fourth-order tensor representations do not suffer the same eigenspace degeneracy problems as the second-order symmetric matrix representation Eq. (10). In particular, if $\mathbf{u}$ and $\mathbf{v}$ are both eigenvectors of $\mathbf{T}$ with eigenvalue λ:

$$\begin{aligned} \mathbf{T} \cdot (\mathbf{u}+\mathbf{v})^3 &= \mathbf{T} \cdot \mathbf{u}^3 + 3\mathbf{T} \cdot \mathbf{u}^2\mathbf{v} + 3\mathbf{T} \cdot \mathbf{u}\mathbf{v}^2 + \mathbf{T} \cdot \mathbf{v}^3 \\ &= \lambda(\mathbf{u}+\mathbf{v}) + 3\mathbf{T} \cdot (\mathbf{u}^2\mathbf{v} + \mathbf{u}\mathbf{v}^2), \end{aligned} \tag{13}$$

and the latter term in Eq. (13) does not generally vanish. This is reflective of the fact that the notion of eigenspaces is replaced by eigenvarieties (typically sets of discrete points) for higher-order symmetric tensors.

3.4.1. Representing symmetric tensors: monomials and spherical harmonics.

As done in [PBS20], we represent such symmetric tensors as spherical polynomials expressed in a basis of spherical harmonics. Note first that symmetric order-d tensors $\mathbf{T}$ on $\mathbb{R}^n$ are determined fully by their values $\mathbf{T} \cdot \mathbf{x}^d$ for unit vectors $\mathbf{x} \in \mathbb{S}^{n-1}$. This holds by a generalized polarization identity and generalizes the correspondence between symmetric bilinear forms and quadratic forms (for $d = 2$). In particular, if $\mathbf{x} = (x_1, \ldots, x_n)$, then:

$$\begin{aligned} \mathbf{T} \cdot \mathbf{x}^d &= \sum_{j_1,\ldots,j_d=1}^{n} T_{j_1 \ldots j_d} x_{j_1} \ldots x_{j_d} \\ &= \sum_{i_1+\cdots+i_n=d} \underbrace{\binom{d}{i_1, \ldots, i_n}}_{u_{i_1 \ldots i_n}} T_{\underbrace{1\ldots1}_{i_1 \text{ times}} \ldots \underbrace{n\ldots n}_{i_n \text{ times}}} x_1^{i_1} \ldots x_n^{i_n} \coloneqq p_{\mathbf{T}}(x_1, \ldots, x_n). \end{aligned} \tag{14}$$

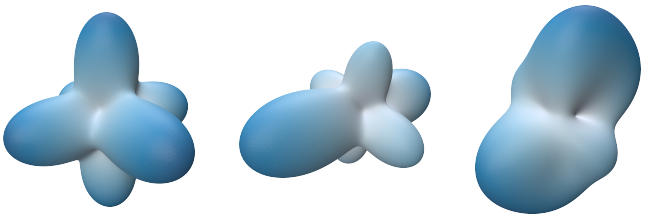


Figure 6: Visualization of tensor polynomials $p_{\mathbf{T}}(\mathbf{x})$. Left: an isotropic odeco tensor with equal eigenvalues. Center: an anisotropic odeco tensor with distinct eigenvalues. Right: a non-odeco tensor that does not decompose into orthogonal rank-1 terms.

In the above, we see that a homogeneous degree d polynomial $p_{\mathbf{T}}$ results, with coefficients denoted by $u_{i_1 \ldots i_n}$. This expression takes into account the symmetries of the tensor coefficients, and shows the equivalence of such symmetric tensors to homogeneous degree-d polynomials in $\mathbb{R}^n$. It also clearly shows the dimensionality of the space, which is equal to the number of multinomial coefficients $\binom{d}{i_1,\ldots,i_n}$. Relevant to us are the case of order-4 symmetric tensors in 2D and 3D, which form 5- and 15-dimensional vector spaces, respectively. Figure 6 shows graphs of such tensor polynomials restricted to $\mathbb{S}^2$ for odeco and non-odeco tensors.

Rather than use a basis of monomials as suggested by Equation (14), we can restrict $p_{\mathbf{T}}$ to $\mathbb{S}^{n-1}$ and use a basis of *spherical harmonics*. Note that the restriction of the tensor polynomials to $\mathbb{S}^{n-1}$ also allows for a natural L^2 inner product (and thus metric) to be defined:

$$\langle p_{\mathbf{T}_1}, p_{\mathbf{T}_2} \rangle = \int_{\mathbb{S}^{n-1}} p_{\mathbf{T}_1}(\mathbf{x}) p_{\mathbf{T}_2}(\mathbf{x}) \, \mathrm{d}\mathbf{x}. \tag{15}$$

Let us concentrate on the $n = 3$-dimensional case first. *Spherical harmonics* are a special set of functions on $\mathbb{S}^{n-1}$ that are orthonormal with respect to this L^2 inner product:

$$Y_l^m(\mathbf{x}), \quad l = 0, 1, \ldots, \quad m = -l, \ldots, l, \tag{16}$$

where l is the *band* of the harmonic and m its *order*. The space of homogeneous degree 4 polynomials (restricted to $\mathbb{S}^2$) can be spanned by the spherical harmonics of bands $l = 0, 2, 4$, which we write compactly as

$$p_{\mathbf{T}}(\mathbf{x}) = \sum_{l \in \{0,2,4\}} \sum_{m=-l}^{l} (q_{\mathbf{T}})_l^m Y_l^m(\mathbf{x}). \tag{17}$$

Note that the number of terms in each band indeed sum to $1 + 5 + 9 = 15$, and the spherical harmonics form a basis. As they are orthonormal, the inner product and distance between tensors $\mathbf{T}_1$ and $\mathbf{T}_2$ can be calculated directly in terms of these coefficients:

$$\langle p_{\mathbf{T}_1}, p_{\mathbf{T}_2} \rangle = \langle \mathbf{q}_{\mathbf{T}_1}, \mathbf{q}_{\mathbf{T}_2} \rangle \quad \text{and} \quad d(\mathbf{T}_1, \mathbf{T}_2) = \|\mathbf{q}_{\mathbf{T}_2} - \mathbf{q}_{\mathbf{T}_1}\|, \tag{18}$$

where the $(q_{\mathbf{T}})_l^m$ have been gathered into vectors $\mathbf{q}_{\mathbf{T}} \in \mathbb{R}^{15}$.

For the $n = 2$ case, spherical harmonics have a 2D counterpart, sometimes called *circular harmonics*, which correspond to the basis functions of a *Fourier series*:

$$\underbrace{1}_{\text{band } 0}, \underbrace{\cos(2\theta), \sin(2\theta)}_{\text{band } 2}, \underbrace{\cos(4\theta), \sin(4\theta)}_{\text{band } 4}. \tag{19}$$

3.5. Odeco variety and band interpretations

With the spaces of symmetric tensors parameterized, we must also be able to specify the subset of odeco tensors Equation (11). In 3D, these form a variety defined by 27 quadratic relations on the monomial coefficients u_{i_1,i_2,i_3} (Theorem 4 of [BDHR17]):

$$c_i(\mathbf{u}) \coloneqq \mathbf{u}^\mathsf{T} A_i \mathbf{u} = 0, \tag{20}$$

where coefficients have been gathered into a vector $\mathbf{u} \in \mathbb{R}^{15}$. These will be used to define an odeco constraint energy in Section 4.2.2. The specific matrices A_i may be found in various sources, e.g., the supplementary material of [PBS20].

[PBS20] have also shown that the 0th and 2nd bands of spherical harmonics have interpretable meanings for odeco tensors. Band 0 encodes the tensor *size*, since

$$(q_\mathbf{T})_0 = C_0 \sum_m \lambda_m, \tag{21}$$

for a constant C_0. Band 2 encodes the tensor *anisotropy*, since

$$\|(\mathbf{q_T})_2\|^2 = C_2 \sum_{m=1}^{n} (\lambda_{m+1} - \lambda_m)^2 \quad (m \in \mathbb{Z}\backslash n\mathbb{Z}), \tag{22}$$

for a constant C_2. These relations and interpretations hold for both 2D and 3D odeco tensors. In particular, we note that a tensor is isotropic (eigenvalues all equal) if and only if $\|\mathbf{q}_2\| = 0$, a fact used in Section 4.2.3 to define an angle distortion energy.

3.5.1. Second-order part of odeco tensors.

A very useful tool when manipulating odeco tensors is to look at its *second-order part* $\mathbf{M}$, defined as the second-order tensor (or $n \times n$ matrix)

$$M_{ij} = T_{ijkk}, \tag{23}$$

obtained by contracting two of its indices (from here on, we use the Einstein summation convention). Note that, thanks to the symmetry of $\mathbf{T}$, it does not matter on which two indices the contraction is done, and $\mathbf{M}$ is a symmetric matrix. Plugging in the odeco definition, Eq. (11), yields

$$M_{ij} = T_{ijkk} = \sum_{m=1}^{n} \lambda_m \left(\sum_{k=1}^{n} u_i^m u_j^m u_k^m u_k^m \right) = \sum_{m=1}^{n} \lambda_m u_i^m u_j^m, \tag{24}$$

$$\text{and} \quad \mathbf{M} = \sum_{m=1}^{n} \lambda_m (\mathbf{u}_m)^{\otimes 2}, \tag{25}$$

where we used the fact that vectors $\mathbf{u}_m = \sum_i u_i^m \mathbf{e_i}$ have unit norm. In other words, the second-order part of a fourth-order odeco tensor is a matrix that shares eigenvalues and eigenvectors $\lambda_1 \mathbf{u}_1, \ldots, \lambda_n \mathbf{u}_n$ with the tensor. This matrix is very useful since it allows us to scale the eigenvectors of odeco tensor $\mathbf{T}$. First notice that taking matrix $\mathbf{M}$ to the power p changes its eigenvalues accordingly:

$$\mathbf{M}^p = \sum_{m=1}^{n} \lambda_m^p \mathbf{u}_m^{\otimes 2}. \tag{26}$$

And contracting the tensor (once) with this matrix yields a fourth-order tensor with modified sizes

$$\mathbf{T}^{p+1} \triangleq \mathbf{T} \cdot \mathbf{M}^p = \left(\sum_{m=1}^{n} \lambda_m^p \mathbf{u}_m^{\otimes 2} \right) \cdot \left(\sum_{m=1}^{n} \lambda_m \mathbf{u}_m^{\otimes 4} \right) = \sum_{m=1}^{n} \lambda_m^{p+1} \mathbf{u}_m^{\otimes 4}. \tag{27}$$

Integer power p can be chosen arbitrarily and matrix $\mathbf{M}^p$ can be computed using standard matrix algorithms; in particular, $p = -1$ gives a unitary (or *octahedral*) frame tensor $\mathbf{T}^0$ and $p = -2$ inverts the lengths of the frame vectors. We demonstrate in Section 4.1 how this tool enables us to properly normalize the integrability energy.

3.5.2. Surface/feature alignment, sizing, and isotropy.

It is crucial to be able to impose alignment of a frame field to the surface or feature curves, as well as sizing of frame axes along these directions; we explain how it is done in the odeco tensor setting.

As noted in [PBS20], rotation of a spherical polynomial may be done by rotating its spherical harmonics coefficients in $\mathbb{R}^{15}$. These 15-dimensional rotation matrices are the *Wigner D-matrices*, an essential tool of quantum physics for the study of angular momentum. Consider the space of odeco frames $\mathbf{q}$ that are aligned to axis $\hat{\mathbf{e}}_z$, and have a length of 1 in this direction. This space can be parametrized by the affine transformation

$$\mathbf{q} = \mathbf{q}_z + \mathbf{B}_z \mathbf{s}, \tag{28}$$

where $\mathbf{s} \in \mathbb{R}^5$ represents a 2D odeco tensor, and $\mathbf{q}_z \in \mathbb{R}^{15}$ and $\mathbf{B}_z \in \mathbb{R}^{15\times 5}$ are constants (see Section 5.1 of [PBS20]). Having a length/sizing constraint l different from 1 merely scales the vector $\mathbf{q}_z$ in the expression above. Then, the space of odeco frames that are aligned to an arbitrary surface normal $\mathbf{n}$, is obtained by applying any rotation matrix $\mathbf{R_n} \in \mathbb{R}^{15\times 15}$ that brings $\hat{\mathbf{e}}_z$ to $\mathbf{n}$:

$$\mathbf{q} = \mathbf{R_n}(\mathbf{q}_z + \mathbf{B}_z \mathbf{s}) = \mathbf{q_n} + \mathbf{B_n} \mathbf{s}. \tag{29}$$

For ease of implementation, instead of such an affine parametrization, we are interested in finding an affine equation

$$\mathbf{A_n q} + \mathbf{b_n} = \mathbf{0} \tag{30}$$

that is satisfied by any $\mathbf{q}$ respecting the alignment constraint. We propose a simple procedure that does not require figuring out $\mathbf{R_n}$ nor $\mathbf{B}_z$ (or even its dimension), and can be adapted to any kind of alignment constraint (e.g., feature edge or corner constraints). First, one forms a matrix $\mathbf{Q}$ that contains as columns a batch of random tensors respecting the *linear part* of the alignment constraint; in this case, frames aligned to $\mathbf{n}$ but having zero length in that direction. Let $\mathbf{N}$ have columns that form a basis of the null space of $\mathbf{Q}^\mathsf{T}$, such that $\mathbf{Q}^\mathsf{T}\mathbf{N} = \mathbf{0}$; we then have $\mathbf{A_n} = \mathbf{N}^\mathsf{T}$. Now let $\mathbf{q}_0$ be any tensor respecting the affine constraint; then $\mathbf{b_n} = -\mathbf{A_n q}_0$. This procedure gives us the affine equation, Eq. (30).

Isotropy of a tensor, i.e., all eigenvalues being equal, amounts to a linear constraint $(\mathbf{q_T})_2 = \mathbf{0}$, as shown by Eq. (22). Given an $\mathbf{n}$-aligned tensor parametrized by Eq. (29), isotropy in the plane orthogonal to $\mathbf{n}$ thus amounts to zeroing out these coefficients for the planar 2D odeco tensor $\mathbf{s}$, leaving an affine space

$$\mathbf{q} = \mathbf{q_n} + \mathbf{B_n^{iso}} \mathbf{s}^{\text{iso}} \tag{31}$$

of dimension 3, i.e., $\mathbf{B_n^{iso}} \in \mathbb{R}^{15\times 3}$ and $\mathbf{s}^{\text{iso}} \in \mathbb{R}^3$. We denote the corresponding affine constraint equations

$$\mathbf{A_n^{iso}} \mathbf{q} + \mathbf{b_n^{iso}} = \mathbf{0}. \tag{32}$$

3.5.3. Eigenvalue sensitivity for tensors.

A last tool we add to our "tensor toolbox" is a result on eigenvalue sensitivity. It answers the question: "*Given an odeco tensor, how do*

its eigenvectors change when slightly perturbing its coefficients?" This tool will help us express integrability in terms of tensor coefficients, as it can transform spatial derivatives on the frame vectors into spatial derivatives on the tensor.

This *eigenvalue perturbation problem* is well-known for matrices, but is easily generalized to higher-order tensors, provided they are odeco. Consider an odeco tensor $\mathbf{T} = \sum_{m=1^n} \lambda_m \mathbf{u}_m^{\otimes 4}$, or, in index notation,

$$T_{j_1 j_2 j_3 j_4} = \sum_{m=1}^{n} \lambda_m u^m_{j_1} u^m_{j_2} u^m_{j_3} u^m_{j_4}. \tag{33}$$

Consider the contraction of $\mathbf{T}$ with one of its eigenvectors, $\mathbf{u}_k$:

$$\begin{aligned} T_{j_1 j_2 j_3 j_4} u^k_{j_4} &= \sum_{m=1}^{n} \lambda_m u^m_{j_1} u^m_{j_2} u^m_{j_3} u^m_{j_4} u^k_{j_4} \\ &= \sum_{m=1}^{n} \lambda_m u^m_{j_1} u^m_{j_2} u^m_{j_3} \delta_{mk} \\ &= \lambda_k u^k_{j_1} u^k_{j_2} u^k_{j_3}, \end{aligned} \tag{34}$$

or, written compactly, $\mathbf{T} \cdot \mathbf{u}_k = \lambda_k \mathbf{u}_k^{\otimes 3}$. Contracting again and following the same procedure we find that $\mathbf{T} \cdot \mathbf{u}_k^{\otimes 2} = \lambda_k \mathbf{u}_k^{\otimes 2}$ and $\mathbf{T} \cdot \mathbf{u}_k^{\otimes 3} = \lambda_k \mathbf{u}_k$, 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 $\mathbf{w}$ has unit norm, it is orthogonal to its variation:

$$(\mathbf{w} + \delta\mathbf{w}) \cdot (\mathbf{w} + \delta\mathbf{w}) = 1 \quad \Rightarrow \quad \mathbf{w} \cdot \delta\mathbf{w} = 0, \tag{35}$$

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

$$\begin{aligned} &(\mathbf{T} + \delta\mathbf{T})(\mathbf{w} + \delta\mathbf{w})^{\otimes 3} = \lambda\mathbf{w} + \delta(\lambda\mathbf{w}), \\ &\cancel{\mathbf{T}\mathbf{w}^3} + 3\mathbf{T}\mathbf{w}^2\delta\mathbf{w} + \delta\mathbf{T}\mathbf{w}^3 = \cancel{\lambda\mathbf{w}} + \delta(\lambda\mathbf{w}), \quad \text{(eigenvector definition)} \\ &\cancel{3\lambda\mathbf{w}^2\delta\mathbf{w}} + \delta\mathbf{T}\mathbf{w}^3 = \delta(\lambda\mathbf{w}), \quad (\mathbf{T}\mathbf{w}^2 = \lambda\mathbf{w}^2, \mathbf{w} \cdot \delta\mathbf{w} = 0). \end{aligned} \tag{36}$$

We find the final eigenvalue sensitivity result: $\delta(\lambda\mathbf{w}) = \delta\mathbf{T}\mathbf{w}^3$; the variation in a scaled eigenvector is obtained by contracting the tensor perturbation three times with the unit eigenvector (note that an eigenvector's sensitivity only depends on itself and not explicitely on the other eigenvectors). From this result we find the partial derivatives

$$\frac{\partial(\lambda w_i)}{\partial T_{j_1 j_2 j_3 j_4}} = w_{j_1} w_{j_2} w_{j_3} \delta_{i j_4}. \tag{37}$$

Section 4.1 will show how this result is put to use to derive an integrability expression.

4. Method

Our method optimizes for a symmetric tensor field that is odeco and integrable nearly everywhere, with sparse violations naturally arising to achieve singularities. A novel integrability energy is first derived in Section 4.1, discretization details are provided in Section 4.2.1, and an "odeco-ness" energy and distortion energies are provided in Sections 4.2.2 and 4.2.3. Lastly, the explicit solver used is described in Section 4.2.6 and frame field recovery is described in Section 4.2.7.

4.1. Integrable odeco fields

On a surface $\mathcal{M} \subset \mathbb{R}^3$ we define a three-dimensional orthogonal frame field $\mathbf{F}(\mathbf{x})$ that is aligned with the surface normal $\mathbf{n}(\mathbf{x})$. At each non-singular point $\mathbf{x} \in \mathcal{M}$, a local lifting of frame vectors $\mathbf{u}(\mathbf{x}'), \mathbf{v}(\mathbf{x}'), \mathbf{n}(\mathbf{x}')$ can be defined on a small neighborhood $\mathcal{N}$ of $\mathbf{x}$. We assume the frame field respects the local injectivity property: $\det(\mathbf{u} \quad \mathbf{v} \quad \mathbf{n}) > 0$; the frame being orthogonal, this means that none of the frame vectors can vanish. As stated by Eq. (9), the frame field is *integrable* if its frame vectors $\mathbf{u}$ and $\mathbf{v}$ have their curl tangent to the surface:

$$(\nabla \times \mathbf{u}) \cdot \mathbf{n} = (\nabla \times \mathbf{v}) \cdot \mathbf{n} = 0. \tag{38}$$

This section constructs an integrability criterion $h_{\mathbf{T}}(\mathbf{x})$, only depending on the tensor field and its derivatives, that is zero when the curl-free condition is satisfied. Let $\mathbf{u} = \lambda\hat{\mathbf{u}}$ be an eigenvector of tensor $\mathbf{T}$. We develop its curl:

$$\begin{aligned} (\nabla \times \mathbf{u})_j &= \epsilon_{jab} \frac{\partial u_b}{\partial x_a} && (\epsilon\text{: Levi-Civita symbol}) \\ &= \epsilon_{jab} \frac{\partial u_b}{\partial T_{k_1 k_2 k_3 k_4}} \frac{\partial T_{k_1 k_2 k_3 k_4}}{\partial x_a} && \text{(chain rule)} \\ &= \epsilon_{jab} \hat{u}_{k_1} \hat{u}_{k_2} \hat{u}_{k_3} \delta_{k_4 b} \frac{\partial T_{k_1 k_2 k_3 k_4}}{\partial x_a} && \text{(sensitivity Eq. (37))} \\ &= \hat{u}_{k_1} \hat{u}_{k_2} \hat{u}_{k_3} \epsilon_{jab} \frac{\partial T_{k_1 k_2 k_3 b}}{\partial x_a} && \text{(sum on } k_4\text{)} \\ &= \hat{u}_{k_1} \hat{u}_{k_2} \hat{u}_{k_3} (\nabla \times \mathbf{T}_{k_1 k_2 k_3})_j && \text{(curl definition),} \end{aligned} \tag{39}$$

where $\mathbf{T}_{k_1 k_2 k_3} \in \mathbb{R}^3$ is the vector with components $(T_{k_1 k_2 k_3 1}, T_{k_1 k_2 k_3 2}, T_{k_1 k_2 k_3 3})$. We wish to make the frame vectors u_k disappear on the right-hand side to obtain an expression that only depends on the tensor coefficients instead. To do so we multiply the expression by the i-th component of $\mathbf{u}$ (note that i and j are free indices here):

$$u_i (\nabla \times \mathbf{u})_j = \lambda_u \hat{u}_{k_1} \hat{u}_{k_2} \hat{u}_{k_3} \hat{u}_i (\nabla \times \mathbf{T}_{k_1 k_2 k_3})_j. \tag{40}$$

Summing this expression for each of the frame vectors $\mathbf{u}, \mathbf{v}, \mathbf{n}$ gives

$$u_i (\nabla \times \mathbf{u})_j + v_i (\nabla \times \mathbf{v})_j + n_i (\nabla \times \mathbf{n})_j = T_{k_1 k_2 k_3 i} (\nabla \times \mathbf{T}_{k_1 k_2 k_3})_j. \tag{41}$$

Both sides of this identity can be seen as a 3-by-3 matrix. We dot-product them with the surface normal $\mathbf{n}$, which gives a scalar equation for each i:

$$\begin{aligned} u_i (\nabla \times \mathbf{u}) \cdot \mathbf{n} + v_i (\nabla \times \mathbf{v}) \cdot \mathbf{n} + n_i \overbrace{(\nabla \times \mathbf{n}) \cdot \mathbf{n}}^{=0} \\ = T_{k_1 k_2 k_3 i} (\nabla \times \mathbf{T}_{k_1 k_2 k_3}) \cdot \mathbf{n}. \end{aligned} \tag{42}$$

The third term is zero as the curl of the surface normal is always tangent to the surface. Taking the L^2 vector norm on both sides we

finally obtain

$$\lambda_u^2((\nabla\times\mathbf{u})\cdot\mathbf{n})^2+\lambda_v^2((\nabla\times\mathbf{v})\cdot\mathbf{n})^2 = \left(\sum_{k_1,k_2,k_3=1}^{3}\mathbf{T}_{k_1k_2k_3}(\nabla\times\mathbf{T}_{k_1k_2k_3})\cdot\mathbf{n}\right)^2. \quad (43)$$

This expression is purely in terms of tensor entries and may be considered for general symmetric tensors, even if they are not odeco.

In [CCR26] (Figs. 10 and 11), it was shown that normalization of this kind of base expression, by a reciprocal squared power of eigenvector lengths, is required to have an energy that is independent of mesh resolution. Around singularities of index $k/4$, frame lengths grow at a rate of $r^{-k/4}$, where r is the radial distance from the singularity. This means that index $+1/4$ (valence 3) singularities blow up, and index $-1/4$ (valence 5) vanish. The normalization does not prevent either from appearing, but introduces a fixed discretization error that does not favor either singularity type. Empirically, the normalization also acts as a barrier preventing frame lengths to become negative. For the normalization in this setting: take Eq. (42), multiply both sides by $\mathbf{M}^{-2}=\sum_m\lambda_m^{-2}\hat{\mathbf{u}}_m^{\otimes 2}$ and we get

$$\frac{u_i}{\lambda_u^2}(\nabla\times\mathbf{u})\cdot\mathbf{n}+\frac{v_i}{\lambda_v^2}(\nabla\times\mathbf{v})\cdot\mathbf{n}=T^{-1}_{k_1k_2k_3i}(\nabla\times\mathbf{T}_{k_1k_2k_3})\cdot\mathbf{n}. \quad (44)$$

Taking the L^2 vector norm again gives

$$\left(\frac{(\nabla\times\mathbf{u})\cdot\mathbf{n}}{\lambda_u}\right)^2+\left(\frac{(\nabla\times\mathbf{v})\cdot\mathbf{n}}{\lambda_v}\right)^2 = \left(\sum_{k_1,k_2,k_3=1}^{3}\mathbf{T}^{-1}_{k_1k_2k_3}(\nabla\times\mathbf{T}_{k_1k_2k_3})\cdot\mathbf{n}\right)^2 =: h_{\mathbf{T}}. \quad (45)$$

This finalizes our expression for our normalized integrability energy $h_{\mathbf{T}}$ used within our optimization.

4.2. Discretization and Optimization

We formulate the search for an integrable frame field as an optimization problem striving to minimize the integral of the integrability measure $h_{\mathbf{T}}$ over the domain Ω,

$$H[\mathbf{T}]=\int_\Omega h_{\mathbf{T}}(\mathbf{x})\,\mathrm{d}\mathbf{x}, \quad (46)$$

which we call the *curl energy*. We start by explaining how this integral is computed in practice, then we present the optimization approach.

4.2.1. Discretization and integration.

The computations are supported by a standard triangle mesh in 2D. At each mesh vertex we store the tensors as their spherical harmonics coefficients $\mathbf{q}\in\mathbb{R}^{15}$, with a normal alignment condition enforced at each vertex. These coefficients are interpolated throughout the domain by a continuous function $\mathbf{q}(\mathbf{x})$ that is piecewise linear on the mesh elements. Consider a triangle $\mathcal{T}$ with vertex positions $\mathbf{x}^1,\mathbf{x}^2,\mathbf{x}^3$, vertex values $\mathbf{q}^1,\mathbf{q}^2,\mathbf{q}^3$ and corresponding linear shape functions $\psi^1(\mathbf{x}),\psi^2(\mathbf{x}),\psi^3(\mathbf{x})$ (such that $\psi^i(\mathbf{x}^j)=\delta_{ij}$). The interpolant over the triangle is defined by

$$\mathbf{q}(\mathbf{x})=\sum_{i=1}^{3}\mathbf{q}^i\psi^i(\mathbf{x}). \quad (47)$$

Note while vertex tensors $\mathbf{q}^i$ are constrained to be normal-aligned and motivated to be odeco, these constraints/energies are not imposed on the interpolated tensors interior to triangles. Computing the tensor curl $h_{\mathbf{T}}$ requires evaluating the spatial derivatives of $\mathbf{q}(\mathbf{x})$; this is done by taking the gradient on both sides:

$$\nabla\mathbf{q}(\mathbf{x})=\sum_{i=1}^{3}\mathbf{q}^i\nabla\psi^i(\mathbf{x}). \quad (48)$$

Note that this gradient is constant per element since the shape fuctions ψ^i are linear. In order to numerically integrate a function $f(\mathbf{q})$ over $\mathcal{T}$ (for example, the integrability measure $h_{\mathbf{T}}$), we apply a 3-point Gaussian quadrature rule

$$\int_{\mathcal{T}}f(\mathbf{q}(\mathbf{x}))\,\mathrm{d}\mathbf{x}\approx\sum_{k=1}^{3}w_k f(\mathbf{q}(\mathbf{x}(\xi_k)))J_{\mathcal{T}}, \quad (49)$$

where ξ_i are the quadrature points on a reference triangle, w_i the corresponding quadrature weights, and $J_{\mathcal{T}}$ the determinant of the Jacobian of the transformation $\mathbf{x}(\xi)$ mapping the reference triangle to $\mathcal{T}$. This 3-point quadrature rule has a degree of precision of 2, meaning that it is exact for quadratic polynomials. The integrability measure $h_{\mathbf{T}}$ is not a polynomial but we found that this degree of precision is accurate enough for our purposes. The integral of f over the whole domain is then obtained by summing the integral over the triangles

$$\int_{\mathcal{S}}f(\mathbf{q}(\mathbf{x}))\,\mathrm{d}\mathbf{x}=\sum_i\int_{\mathcal{T}_i}f(\mathbf{q}(\mathbf{x}))\,\mathrm{d}\mathbf{x}. \quad (50)$$

This calculation is embarrassingly parallel, as the sum over the triangles can be distributed over a large number of CPUs.

Note that evaluating the integrability measure $h_{\mathbf{T}}$ at the quadrature points ξ_i requires an estimate of the surface normal $\mathbf{n}$. For this purpose we simply use the triangle normal $\mathbf{n}_{\mathcal{T}}$.

4.2.2. Relaxation of the odeco constraints.

At this point, it might be tempting to simply minimize $H[\mathbf{T}]$ under the constraint that the field is everywhere odeco. However, this would fail to introduce new singularities as it is often energetically cheaper to distribute the curl on the domain rather than to introduce the fixed discretization cost of singularities (this fixed cost comes from the fact that a piecewise linear representation cannot capture the frame vectors blowing up when nearing singularities). To address this, we relax the odeconess constraint and allow the solver to introduce non-odeco terms that reduce the integrability cost in the neighborhood of singularities. A natural choice would be to introduce an *odeco penalty*

$$C[\mathbf{T}]\triangleq\sum_i\int_\Omega c_i^2(\mathbf{T}(\mathbf{x}))\,\mathrm{d}\mathbf{x}, \quad (51)$$

where $c_i(\cdot)$ are the quadrics defining the odeco variety referred to in Eq. (20); this functional is thus a 4th-order polynomial in the tensor coefficients.

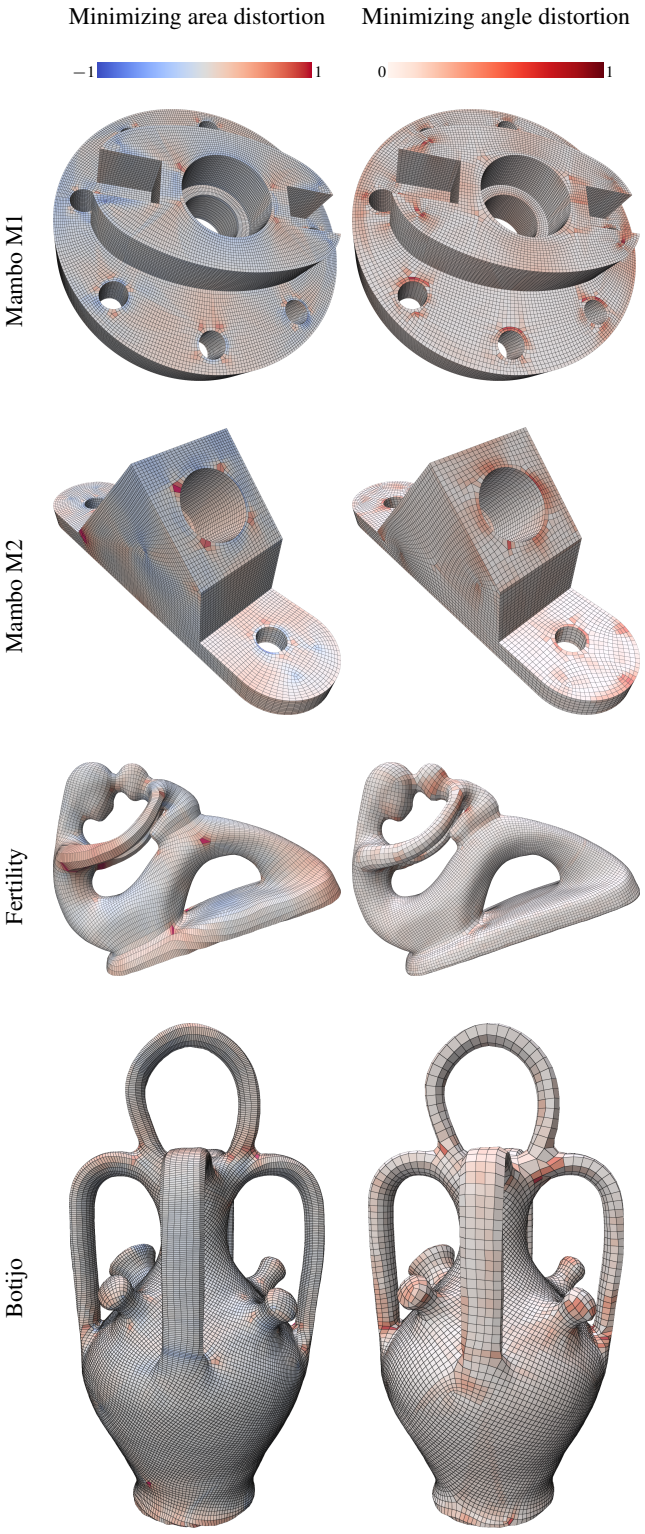


Figure 7: Quad mesh results with area- and angle-distortion metrics on select CAD models (feature alignment, but no sizing constraints) and smooth models (no features) using different distortion energies. Note the higher anisotropy in the area-preserving mode and the differing singularity placements between the different modes of the framework.

In practice, we introduce a *normalized odeco penalty* that applies an L^2 normalization before application of squared constraint equations:

$$\hat{C}[\mathbf{T}] \triangleq \sum_i \int_\Omega c_i^2 \left(\frac{\mathbf{T}}{\|\mathbf{T}\|} \right) \mathrm{d}\mathbf{x}, \tag{52}$$

Empirically, this prevents the shrinking of the tensors, as the standard odeco penalty $C[\mathbf{T}]$ can be minimized by setting $T = 0$.

Note that most results on tensor-based integrability assume a tensor field that is odeco everywhere. The relaxation breaks this property. Our core assumption is that notions of integrability and distortion still make sense when the field is not perfectly odeco, and this is validated by experiments.

4.2.3. Distortion metrics.

Lastly, we optionally include distortion metrics that aim to preserve the area of quad elements, or aim to minimize the angle distortion, promoting isotropic (conformal) quad elements. Recall that $\det \mathbf{M} = \Pi_m \lambda_m$, which is the inverse of frame volume or area (as the eigenvalues represent gradient norms).

$$\Phi_{\text{area}}[\mathbf{T}] = \int_\Omega \log^2 \left(A_0 \det \mathbf{M} \right) \mathrm{d}\mathbf{x}, \tag{53}$$

$$\Phi_{\text{angle}}[\mathbf{T}] = \int_\Omega \left\| \mathbf{b}_{\mathbf{n}}^{\text{iso}} - \mathbf{A}_{\mathbf{n}}^{\text{iso}} \mathbf{q} \right\|^2 \mathrm{d}\mathbf{x}. \tag{54}$$

Above, A_0 denotes the target area of quad elements, and $\mathbf{n}$ is the normal of the triangle we're integrating over. Recall that $\mathbf{A}_{\mathbf{n}}^{\text{iso}}$ and $\mathbf{b}_{\mathbf{n}}^{\text{iso}}$ were defined in Eq. (32). Rows of $\mathbf{A}_{\mathbf{n}}^{\text{iso}}$ are made orthonormal so that this objective measures actual distance to the affine space.

4.2.4. Full objective and initialization

The combination of the normalized integrability and odeco energies and the optional distortion energies leads us to our full objective function:

$$E[\mathbf{T}] \coloneqq H[\mathbf{T}] + \kappa_{\text{odeco}} \hat{C}[\mathbf{T}] + \kappa_{\text{area}} \Phi_{\text{area}}[\mathbf{T}] + \kappa_{\text{angle}} \Phi_{\text{angle}}[\mathbf{T}]. \tag{55}$$

The κ weights used in our results vary depending on the desired amount of anisotropy in the end mesh. The exact settings are described at the start of Section 5.

Note that if L is a characteristic size for the domain Ω, then the energies $\hat{C}$, Φ_{area} and Φ_{angle} scale with L^2 whereas H does not scale. This can be problematic when working with models at different scales. To achieve scale independence, we divide the integrands by the local triangle area $J_{\mathcal{T}}$. This ensures that the different energies remain adequately weighted regardless of scale.

As an initial solution, we solve for a smooth octahedral (i.e., odeco and unit norm) field. Smoothness is achieved by a Dirichlet energy Φ_{smooth} defined as in [PBS20]:

$$\Phi_{\text{smooth}}[\mathbf{T}] = \sum_j \frac{1}{2} \int_\Omega \left\| \nabla q_j \right\|^2 \mathrm{d}\mathbf{x}. \tag{56}$$

The octahedral variety is cut out from the odeco variety by an affine space $\mathbf{A}^{\text{octa}}\mathbf{q} + \mathbf{b}^{\text{octa}} = \mathbf{0}$, which can be computed in the same way

as presented in Section 3.5.2. The search for an initial smooth octahedral field can thus be formulated by minimizing

$$E_{\text{init}}[\mathbf{T}] := \Phi_{\text{smooth}}[\mathbf{T}] + \kappa_{\text{odeco}}\hat{C}[\mathbf{T}] \quad \text{s.t.} \quad \mathbf{A}^{\text{octa}}\mathbf{q} + \mathbf{b}^{\text{octa}} = \mathbf{0}. \tag{57}$$

For this optimization we consistently set $\kappa_{\text{odeco}} = 1$. Note that the initial solution may violate the sizing constraints we set in the actual optimization; in that case we first project the tensors onto their corresponding affine constraint space.

4.2.5. Boundary alignment.

As seen in Section 3.5.2, any alignment or sizing constraint on an odeco tensor can be written as an affine expression at each node

$$\mathbf{A}\mathbf{q} + \mathbf{b} = \mathbf{0}, \tag{58}$$

with $\mathbf{A} \in \mathbb{R}^{10\times 15}$ and $\mathbf{b} \in \mathbb{R}^{10}$. As a reminder, we determine this linear subspace by randomly generating a batch of (odeco) tensors respecting the constraint we want to impose, which allows us to compute $\mathbf{A}$ and $\mathbf{b}$. The 5-dimensional solution space reflect the 5 degrees of freedom afforded by the 2D tensor subspace orthogonal to the fixed frame along the alignment direction.

Such constraints are imposed at nearly every vertex of the mesh. On general surface points, we impose alignment to the surface normal, with sizing of 1. On feature curve points, alignment tangent to the curve is enforced, with user-specified sizing in that direction. Note that the normal direction alignment is not imposed and allowed to be free, as these features are usually sharp edges with ambiguous normal directions. On "corners" where multiple feature curves meet, no alignment conditions are imposed due to the even greater normal (and feature tangent) ambiguity.

4.2.6. Solver and gradient computation.

The optimization problem we pose (minimizing Eq. (55) subject to Eq. (58)) is a nonlinear and nonconvex problem with linear constraints. We address it using a quasi-Newton method, namely the limited-memory BFGS algorithm, which we modify to impose the linear constraints at each update. Let $\mathbf{q}$ be a tensor at a node that respects its alignment constraint $\mathbf{A}\mathbf{q} + \mathbf{b} = \mathbf{0}$, and let $\mathbf{g} \in \mathbb{R}^N$ be the gradient of the objective function with respect to $\mathbf{q}$. To ensure that any L-BFGS update does not violate the constraint, we project $\mathbf{g}$ onto the space normal to the rows of $\mathbf{A}$, effectively ensuring $\mathbf{A}\mathbf{g} = \mathbf{0}$. Recalling that $\mathbf{A}$ was constructed to have orthonormal rows, this projection is simply done by iteratively subtracting from $\mathbf{g}$ its component along row $\mathbf{a}_i$:

$$\text{for } i = 1,\dots,M: \quad \mathbf{g} := \mathbf{g} - (\mathbf{g}\cdot\mathbf{a}_i)\mathbf{a}_i. \tag{59}$$

We use the L-BFGS implementation of ALGLIB [Boc26]. Analytical gradients are computed via automatic differentiation, using the TinyAD library [SBB*22].

4.2.7. Frame field recovery.

The minimization of (55) provides a field of tensors that is odeco everywhere except in the vicinity of singularities. For the following mesh generation step, we recover a frame on every triangle face by projecting the linearly-interpolated tensors onto the odeco variety using the semidefinite projector described in [PBS20]. Let us denote the recovered field on each face t:

$$G_t := [\nabla u | \nabla v] \in \mathbb{R}^{3\times 2}, \tag{60}$$

where u, v refer to putative components of a seamless parameterization.

4.3. Mesh generation

There is a one-to-one correspondence between quad meshes and *integer-grid maps* (IGM) [BCE*13], which are discrete realizations of the integer seamless maps described in Section 3.1. Consequently, in our context the goal of the mesh generation stage consists of determining an IGM ϕ_{IGM}, which is as similar as possible to the integrable odeco field G, i.e., a piecewise linear map ϕ_{IGM} minimizing

$$E_{IGM} = \int_\Omega \left\| J_{\phi_{IGM}} - G \right\|_2^2 \, dA \tag{61}$$

subject to IGM constraints, including local injectivity, and integer-quantization of transition functions, singularities, and feature curves (cf. [BCE*13]). Please note that up to discretization errors, the integrable odeco field G already coincides to a seamless map ϕ_S such that the main deviation to ϕ_{IGM} is induced by additional integer-quantization constraints, not being considered in the integrable odeco field optimization. We employ an established approach, conceptually identical to [LCBK19] Figure 4, and the robust pipeline described in [CSH*25] Section 8. The key idea is to decompose the challenging IGM generation task into a series of independent sub-steps, for which robust and fast algorithms are available. In a nutshell our mesh generation algorithm consists of (i) generating a locally injective seamless map ϕ_S closely resembling the integrable odeco field G by optimizing a QP induced by Eqn.(61) in conjunction with the linearized local injectivity constraints of [BCE*13], (ii) resolving numerical imprecisions of the seamless map constraints with [MC19], (iii) generating a T-mesh partitioning by tracing the motorcycle complex [CBK15], (iv) integer-quantization of T-mesh arcs [HWB23], (v) T-mesh per patch parametrization to the integer-quantized rectangles determined in the previous step resulting in an initial IGM ϕ_{IGM}, (vi) global relaxation of ϕ_{IGM} by minimizing the symmetric Dirichlet distortion energy [SS15] w.r.t. deviation from ϕ_S, while respecting the quantization constraints, (vii) extracting the quad mesh induced by ϕ_{IGM} with the robust algorithm of [EBCK13].

4.4. Overview of pipeline and capabilities

Putting everything together, our end-to-end meshing pipeline proceeds in the following main steps:

1. Compute a smooth *octahedral* (i.e., unit odeco) field $\mathbf{T}^{\text{smooth}}$ with alignment constraints, Eq. (57).
2. Using $\mathbf{T}^{\text{smooth}}$ as initialization, compute an integrable *odeco* field $\mathbf{T}^{\text{integ}}$ with alignment and sizing constraints and optional distortion objective, Eq. (55).
3. Recover frame field G by linearly interpolating odeco tensors to every triangle face, and projecting onto the odeco variety using the semidefinite projector of [PBS20] (Section 4.2.7).
4. Compute integer-grid map and extract quad mesh (Section 4.3).

We summarize the constraints that can be prescribed by our algorithm:

Alignment: can be set strongly or weakly through linear constraints on the tensor coefficients (Section 3.5.2). The optimization empirically aligns to principal curvature directions, and the user can enforce stricter curvature alignment constraints if desired.

Sizing: can be prescribed in two ways:

- *impose size together with a direction*, either strongly or weakly; this is done by augmenting the alignment constraint, making it affine instead of linear.
- *impose local area* in a weak sense through the area distortion energy, Eq. (53).

Anisotropy: can be controlled in a weak sense through the angle distortion energy, Eq. (53).

Future work will explore user prescription of singularity positions.

5. Results

We validate our method on two subsets: the 9 "Medium" and 1 "Simple" feature-rich CAD models from the MAMBO dataset [Led20], and 4 smooth models from the dataset of [MPZ14], sans feature curves. MAMBO is split into three categories of increasing difficulty: Basic, Simple, and Medium, so our choice reflects the most challenging set of models. All meshes obtained in our results and comparisons below are included as part of the supplementary materials with our submission. If this submission is accepted, we will release an open-source version of the code.

Computations were done on an Apple M3 Pro with 12 threads parallelized by OpenMP. We stop the L-BFGS iterations when the relative objective improvement is $< 10^{-5}$. We used the following sets of weighting coefficients for the optimization unless otherwise stated:

- for *area* distortion minimization on *CAD models* we set $\kappa_{\text{odeco}} = 10$ and $\kappa_{\text{area}} = 0.1$ (and $\kappa_{\text{angle}} = 0$),
- for *area* distortion minimization on *smooth models* we set $\kappa_{\text{odeco}} = 10$, $\kappa_{\text{area}} = 0.1$, and $\kappa_{\text{angle}} = 0.0001$; the addition of a small amount of Φ_{angle} prevents the solver from completely collapsing frames along one dimension,
- for *angle* distortion minimization we set $\kappa_{\text{odeco}} = 1$ and $\kappa_{\text{angle}} = 0.01$ (and $\kappa_{\text{area}} = 0$),
- for sizing constraints only (no distortion energies) we set $\kappa_{\text{odeco}} = 1$.

Empirically, we found that the higher degree of anisotropy that is typical in area-preserving schemes required a higher $\kappa_{\text{odeco}} = 10$ for best performance.

5.1. Demonstration of Distortion Energies

We demonstrate the efficacy of our Φ_{area} and Φ_{angle} distortion energies in Figure 7 on two CAD models, M1 and M4 from the MAMBO dataset, and two smooth models from the dataset of [MPZ14], Fertility and Botijo. Two additional smooth models,

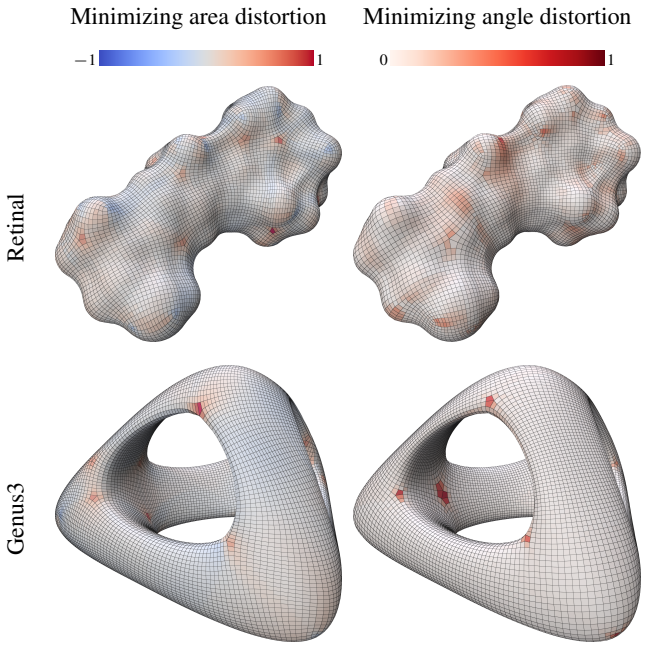


Figure 8: Quad mesh results on two smooth models from [MPZ14] with area- and angle-distortion minimization.

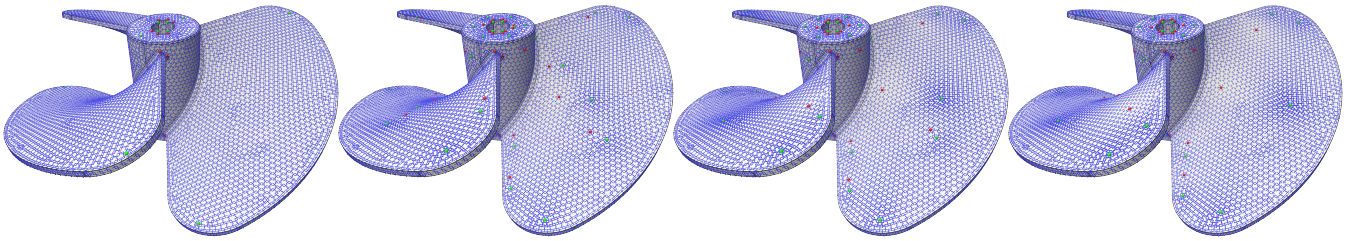
Retinal and Genus3 are shown in Figure 8. The weighting parameters used are those listed above. Clear qualitative differences in the anisotropy of the quads can be seen, and note also that the global singularity positions differ between the two modes of the framework (e.g., on the side of M2 and on Fertility). Another qualitative comment is that one can see rough principal curvature alignment along regions of highest sectional curvature, e.g., the arms of Fertility and Botijo.

Also displayed via colorbars are area and angle distortion metrics. The area disortion measure is the log of the quad area over the target area. As our quads are all nearly planar, we simply measure area by splitting along a diagonal and summing the resulting triangle areas. The angle distortion metric is log of max dimension length over min dimension length. These "height" and "width" dimension lengths are estimated by averaging the length of opposing sides of the quads. As can be seen, the distortion is mostly centered around singularities, trade-offs that the method deems necessary for reducing distortion more globally on the model.

Figure 9 shows the evolution of the integrable frame field optimization (step 2 of the pipeline, Section 4.4), along with the singularity counts and positions. The solver introduces and places new singularities to achieve the desired sizing and distortion objectives.

The size prescription capability allows users to achieve size transitions in specific regions. This is illustrated in Fig. 10 where we perform a 1-2 and a 1-5 size transition. The solver inserts pairs of 3-5 singularities to achieve the desired element sizing.

In some cases, our integrable solver can take as input a non-meshable smooth frame field and make it meshable. Figure 11 shows an example where a smooth frame field is not meshable; this



Iter. 0, $n_3 = 22$, $n_5 = 22$ | Iter. 250, $n_3 = 73$, $n_5 = 73$ | Iter. 750, $n_3 = 64$, $n_5 = 64$ | Iter. 1395, $n_3 = 59$, $n_5 = 59$

Figure 9: Evolution of the integrable frame field optimization producing the mesh of Fig. 1 (size is set to 1 along feature curves, and we minimize angle distortion) on MAMBO model M8. Green and red dots represent singularities of valence 3 and 5, respectively. Note the balanced singularity counts n_3 and n_5, which is due to the torus topology (Euler characteristic is zero).

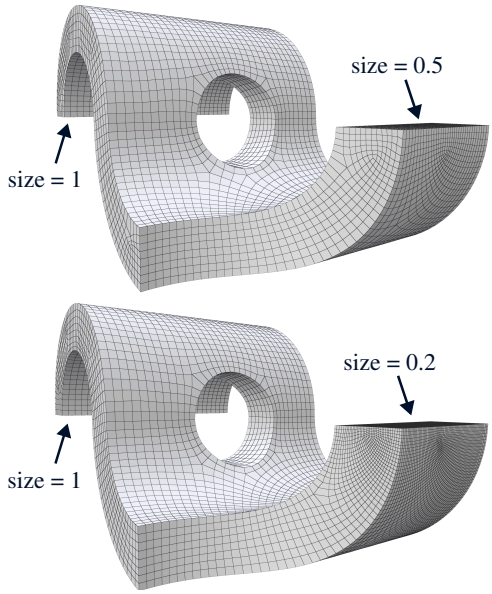


Figure 10: Example of size transitions with angle distortion minimization on MAMBO model B1. Sizes are set on the boundaries of the rectangular front and back faces designated by the arrows.

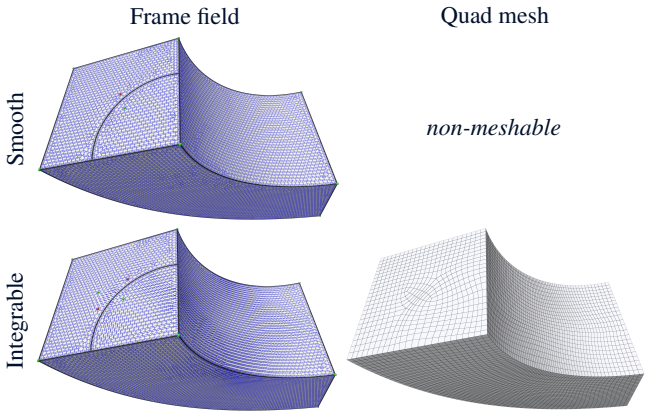


Figure 11: In the presence of feature curves, smooth frame fields can be unmeshable because of limit cycles (top). Our solver inserts new singularities that make the frame field meshable (bottom). Here area distortion is minimized ($\kappa_{\text{odeco}} = 1, \kappa_{\text{area}} = 0.1$), with no size prescription.

is due to an invalid frame field topology where separatrices emanating from singularities form limit cycles. Our solution (with area distortion minimization) produces new singularities that make the frame field globally meshable.

5.2. Comparison with Integrable PolyVector Fields

We compared with the work of [DVPSH15], which we abbreviate IPV. This was the only publicly available alternate approach that jointly optimizes for singularity placement and integrability of a frame field together in the setting of orthogonal anisotropic quad meshes. Their method actually does not require strict orthogonality, but it can be strongly motivated by setting a parameter s close to 1 in their "geometric order" term (see Eq. 22 and preceding equations in their work). We set $s = 0.9$ and $w_b = 10$ (the weighting parameter in front of the geometric order energy) for our comparisons. We used the implementation available as part of Version 2.0.0 of the Directional [VCD*16] library with default parameters otherwise. A past version of the library was used as an implementation of IPV is not available as part of the current version.

A conceptual difference between IPV and our method (besides different algebraic representations: complex vs. odeco polynomials) is that our optimization variables are the odeco polynomial coefficients, and not the explicit frame vectors. The benefit of this is that some energies become convex (e.g. Φ_{smooth} and Φ_{angle}), and it obviates the need for explicit objective terms to preserve geometric order of frame vectors. This conceptual advantage is made possible by the existing theory of odeco tensors and the description of the variety by quadratic polynomials.

The models considered in our comparison were all of the "Medium" models (the hardest of the three categories) in the MAMBO dataset. These CAD models come with feature curves on the boundaries of parametric patches, and we imposed alignment and tangential sizing of 1 on all such curves. No distortion energies were imposed as IPV does not directly accommodate these in their formulation. Over these models, we measured the *skew* of

each quad, which is the largest absolute difference from 90° of the four corners. Our results, reported in Table 1, were consistently better in terms of orthogonality, with a mean skew of 3.05° with standard deviation 0.88°, while IPV's mean skew was 5.07° with standard deviation 2.17°.

The above averages were calculated over models where the frame field results of both methods could be used to successfully generate a quad mesh (with the procedure outlined in Section 4.3). IPV suffered 4 failures over the 9 models, while our method suffered 1 failure on M3 (shared with IPV). The lack of robustness for both methods is often due to the challenging "thin corner" cases that often arise in CAD models, e.g., see Figure 13.

Results on models M4 and M8 are shown in Figure 12. Our results not only have more orthogonal elements, but also show better satisfaction of boundary sizing constraints. This is due in large part to different singularity configurations, with more numerous pairs of valence 3 and 5 singularities that allow for graceful and effective size transitions. This is quite visible on the fan blades of M8.

5.3. Comparison with Rectangular Surface Parameterization

We compare here with the work of [CC25], which we abbreviate RSP. In particular, we consider the greedy, iterative automatic placement of singularities proposed in Section 5.3.2 of their paper. Without a public implementation, we asked the authors to run their method on four meshing scenarios shown in Figure 14 utilizing the Φ_{iso}, Φ_{area}, and Φ_{angle} objectives described in Section 5.1.1 of their work. For a fair comparison of distortion metrics, we compare across a similar number of singularities for each scenario.

The scenarios presented are three challenging planar ones: SquareLine, SquareCurve, and Uturn; and one surface model, S4 from the "Simple" (2nd hardest) category from the MAMBO dataset. On all of these models, we impose feature alignment with tangent sizing constraints of 1. The distortion metrics visualized

Table 1: Skewness and runtimes comparison with IPV [DVPSH15] on "Medium" MAMBO CAD models. On average, our method achieves 3.05° mean skewness while IPV achieves 5.07°, and is more robust over this challenging dataset. The improvements come at the price of an average runtime that is 2.17× longer than IPV.

Model	Mean Skewness (°)		Runtime (m:ss)	
	IPV	Ours	IPV	Ours
M1	4.15°	3.00°	4:03	11:54
M2	5.78°	2.39°	1:58	7:31
M3	—	—	5:24	12:18
M4	3.32°	3.22°	3:28	7:01
M5	3.13°	3.29°	8:08	17:14
M6	—	1.83°	5:43	6:04
M7	—	4.85°	11:05	9:23
M8	8.98°	3.58°	3:24	8:58
M9	—	2.27°	3:36	6:29
Mean	5.07°	3.05°		
Std	2.17°	0.88°		

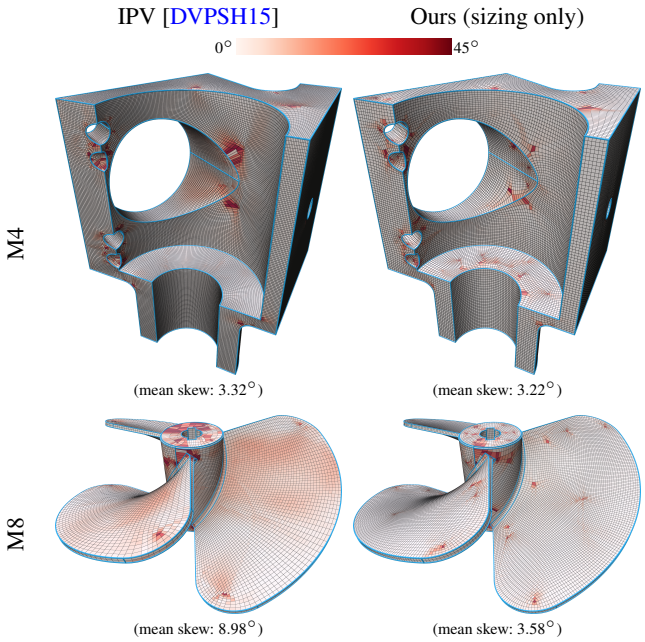


Figure 12: A skewness comparison with IPV [DVPSH15] on MAMBO models M4 and M8 with uniform tangent sizes on feature curves and no distortion energies. Also note the better satisfaction of sizing constraints via differing singularity configurations,e.g., on the fan blades of M8.

over the meshes are the same as those described in Section 5.1 for Figure 7. The reported parenthetical numbers are mean values, derived as follows: for area distortion, we use the log squared of the ratio between quad area and target area; for angle distortion, we use the log squared of the width to height ratios.

Across the planar scenarios, we see that our method produces qualitatively different singularity placements and superior distortion metrics. Also notable are the almost exact symmetry of our singularity placements, relative to those of RSP. In both SquareLine and SquareCurve, one can see differences in positioning relative to the central feature curves and to other singularities in the RSP results, violating the 180° rotational symmetry of the scenarios. This is perhaps not unexpected given the greedy, iterative placement strategy of RSP. Lastly, on S4, we compare the use of Φ_{iso} for their method with weightings of $\kappa_{\text{area}} = 0.1$ and $\kappa_{\text{angle}} = 0.0001$ (and $\kappa_{\text{odeco}} = 10$) for our method. We do not have an explicitly isometry-promoting energy, so we chose this combination of weights. No distortion metrics are visualized on these S4 meshes, but mean values are reported. Perhaps as expected, the results show a clear improvement upon RSP's result in area distortion, and slightly worse performance on angle distortion.

5.4. Runtimes and Computational Cost

In Table 2, we report the runtimes and mesh statistics for all results that are present in figures in the paper. Our optimization is nonconvex and high-dimensional, with 15 variables per vertex and 27 quadrics being used to define the nonconvex odeco energy. Much

of the computational cost derives from these factors, and the time for convergence is naturally hard to predict.

In Table 1, we compare runtimes with IPV on the Medium models of the MAMBO dataset. As can be seen, we are usually 1.5-3x slower, though are sometimes comparable and achieve superior skewness.

Table 2: Runtime and mesh statistics for the results shown in this paper.

Fig.	Model	#T	#Q	Runtime (m:ss)
7	M1 (area)	37766	42279	10:57
	M1 (angle)	37766	34454	3:25
	M2 (area)	22133	16942	12:06
	M2 (angle)	22133	11468	8:42
	Fertility (area)	27954	13352	9:13
	Fertility (angle)	27954	17498	4:42
	Botijo (area)	82332	33550	39:23
	Botijo (angle)	82332	14378	26:28
8	Retinal (area)	7282	10906	3:58
	Retinal (angle)	7282	9635	2:11
	Genus3 (area)	13312	19581	3:58
	Genus3 (angle)	13312	17938	8:26
10	B1 (0.5)	12928	6869	5:38
	B1 (0.2)	12928	12842	6:54
11	B18	27749	6846	13:30
12	M4	38928	33324	7:01
	M8	26637	46492	8:58
14	SquareLine (area)	8678	9601	3:34
	SquareLine (angle)	8678	10484	4:33
	SquareCurve (area)	8775	9984	1:05
	SquareCurve (angle)	8775	10444	0:47
	Uturn (area)	9462	5229	5:31
	Uturn (angle)	9462	5765	4:55
	S4	30988	10180	11:26

6. Conclusion

We present and validate a framework for joint optimization of singularity positions and integrability of a frame field for the purposes of anisotropic surface quad meshing. The framework adapts normal-aligned 3D odeco tensors and nontrivially extends the 2D integrability energy of [CCR26] to that of intrinsic vector curl on embedded, curved surfaces in 3D. We show superior orthogonality and size constraint satisfaction when compared to Integrable PolyVector Fields [DVPSH15] on a dataset of challenging CAD models. We also show improved performance with respect to area and angle distortion metrics on a small set of examples when compared to the greedy, iterative automatic placement strategy of [CC25]. Lastly, on a few smooth models from the [MPZ14] dataset, we show the efficacy of the method in the absence of features and we see empirical alignment to principal curvature directions in regions of highest sectional curvature.

6.1. Limitations and Future Work

Our optimization framework does present several opportunities for improvement and for natural extension of the ideas within. One notable issue is that the method is not fully robust, partially due to the presence of very challenging CAD corners, like the one shown in Fig. 13. It is likely that a local manual fix at such regions should help to make the method even more robust. Additionally, one could consider feeding our more optimal singularity placements to a method like Rectangular Surface Parameterization [CC25] to even further improve the orthogonality of our end meshes.

As noted in Section 5.4, the optimization problem at hand is large and nonconvex, so there is significant computational cost associated with the method. One could attempt alternate optimization strategies, e.g., Gauss-Newton methods, coarser approximation strategies, e.g., one-point quadrature rules, or GPU parallelization of some steps. One notable thought is to attempt an intrinsic odeco tensor approach, dropping the variable count significantly. However, this may lose the extrinsic advantages of natural alignment at sharp creases and regions of high sectional curvature.

We have also begun preliminary exploration into the exciting prospect of generating integrable volumetric frame fields for hexahedral meshing. The integrability energy is easily adapted to this setting, and it is an enticing prospect to see if the framework can generate reasonable singularity graphs for the analogous parameterization-based hexahedral mesh generation pipeline.

Lastly, it would be interesting to try to extend the framework to non-orthogonal (non-odeco) frame fields and develop an integrability energy in this setting. The integrability analysis (Section 4.1) and eigenvalue sensitivity (Section 3.5.3) are only valid under the odeco assumption. These could be extended to the non-odeco setting, but the resulting integrability energy becomes more complex: a rational expression of the tensor coefficients.

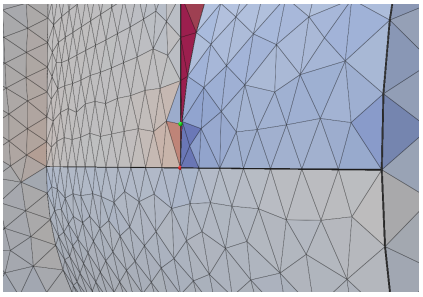


Figure 13: Challenging CAD corners can lead to non-meshable singularity configurations.

Acknowledgments

We would like to thank Étienne Corman for providing us with additional results from the RSP method. This work is supported by Wallonie-Bruxelles International and the European Research Council (grant no. 101 071 255).

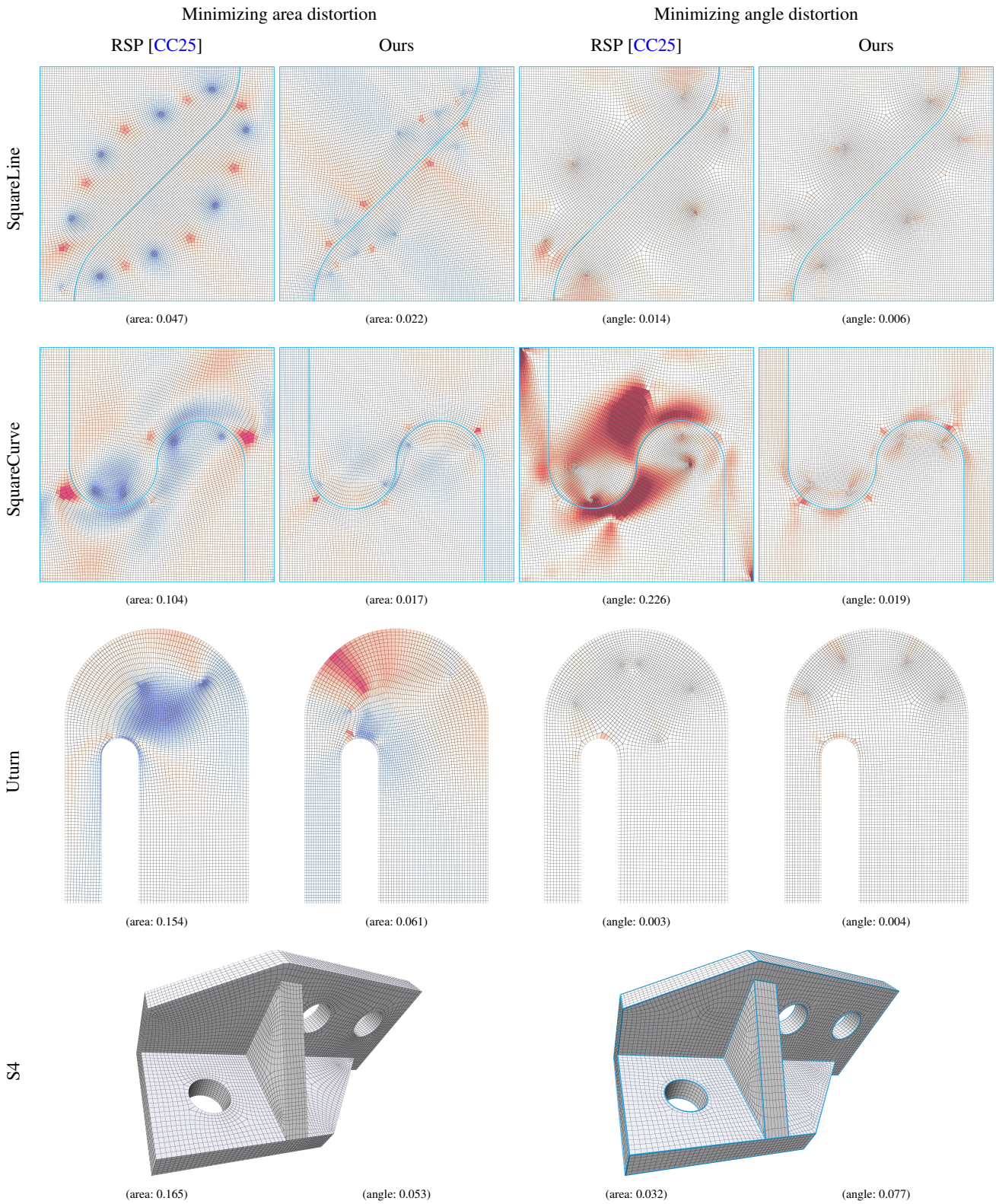


Figure 14: A comparison of minimizing area, angle, and isometric distortion on three challenging planar scenarios and one CAD model. Superior performance at similar singularity counts is seen in the planar models, as well as superior symmetry. On the CAD model, an isometric result for RSP is compared against a mostly area-preserving result from our method.

References

[ABF05] ARNOLD D. N., BOFFI D., FALK R. S.: Approximation by quadrilateral finite elements. *SIAM Journal on Numerical Analysis 43*, 2 (2005), 585–614. 1

[ACSD*03] ALLIEZ P., COHEN-STEINER D., DEVILLERS O., LÉVY B., DESBRUN M.: Anisotropic polygonal remeshing. *ACM Trans. Graph. 22*, 3 (July 2003), 485–493. URL: https://doi.org/10.1145/882262.882296, doi:10.1145/882262.882296. 2

[AMR88] ABRAHAM R., MARSDEN J. E., RATIU T.: *Manifolds, Tensor Analysis, and Applications*, 2 ed., vol. 75 of *Applied Mathematical Sciences*. Springer, New York, 1988. 4

[BCE*13] BOMMES D., CAMPEN M., EBKE H.-C., ALLIEZ P., KOBBELT L.: Integer-grid maps for reliable quad meshing. *ACM Trans. Graph. 32*, 4 (July 2013). URL: https://doi.org/10.1145/2461912.2462014, doi:10.1145/2461912.2462014. 10

[BCGB08] BEN-CHEN M., GOTSMAN C., BUNIN G.: Conformal flattening by curvature prescription and metric scaling. *Computer Graphics Forum 27*, 2 (2008), 449–458. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2008.01142.x, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2008.01142.x, doi:https://doi.org/10.1111/j.1467-8659.2008.01142.x. 2

[BCW17] BRIGHT A., CHIEN E., WEBER O.: Harmonic global parametrization with rational holonomy. *ACM Trans. Graph. 36*, 4 (July 2017). URL: https://doi.org/10.1145/3072959.3073646, doi:10.1145/3072959.3073646. 3

[BDHR17] BORALEVI A., DRAISMA J., HOROBEŢ E., ROBEVA E.: Orthogonal and unitary tensor decomposition from an algebraic perspective. *Israel Journal of Mathematics 222*, 1 (Oct. 2017), 223–260. URL: https://doi.org/10.1007/s11856-017-1588-6, doi:10.1007/s11856-017-1588-6. 6

[BLP*13] BOMMES D., LÉVY B., PIETRONI N., PUPPO E., SILVA C., TARINI M., ZORIN D.: Quad-mesh generation and processing: A survey. *Computer Graphics Forum 32*, 6 (2013), 51–76. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12014, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12014, doi:https://doi.org/10.1111/cgf.12014. 2

[Boc26] BOCHKANOV S.: ALGLIB. https://alglib.net, 1999–2026. Numerical analysis and data processing library. 10

[BZK09] BOMMES D., ZIMMER H., KOBBELT L.: Mixed-integer quadrangulation. *ACM Trans. Graph. 28*, 3 (July 2009). URL: https://doi.org/10.1145/1531326.1531383, doi:10.1145/1531326.1531383. 2

[CBK15] CAMPEN M., BOMMES D., KOBBELT L.: Quantized global parametrization. *ACM Trans. Graph. 34*, 6 (Nov. 2015). URL: https://doi.org/10.1145/2816795.2818140, doi:10.1145/2816795.2818140. 10

[CC23] COIFFIER G., CORMAN E.: The method of moving frames for surface global parametrization. *ACM Trans. Graph. 42*, 5 (Sept. 2023). URL: https://doi.org/10.1145/3604282, doi:10.1145/3604282. 2

[CC25] CORMAN E., CRANE K.: Rectangular surface parameterization. *ACM Trans. Graph. 44*, 4 (July 2025). URL: https://doi.org/10.1145/3731176, doi:10.1145/3731176. 2, 3, 13, 14, 15

[CCR26] COUPLET M., CHEMIN A., REMACLE J.-F.: Size-controlled quadrilateral meshing using integrable odeco fields. *Computer-Aided Design 190* (2026), 103974. URL: https://www.sciencedirect.com/science/article/pii/S0010448525001356, doi:https://doi.org/10.1016/j.cad.2025.103974. 2, 8, 14

[CCS*21] CAMPEN M., CAPOUELLEZ R., SHEN H., ZHU L., PANOZZO D., ZORIN D.: Efficient and robust discrete conformal equivalence with boundary. *ACM Trans. Graph. 40*, 6 (Dec. 2021). URL: https://doi.org/10.1145/3478513.3480557, doi:10.1145/3478513.3480557. 3

[CLW16] CHIEN E., LEVI Z., WEBER O.: Bounded distortion parametrization in the space of metrics. *ACM Trans. Graph. 35*, 6 (Dec. 2016). URL: https://doi.org/10.1145/2980179.2982426, doi:10.1145/2980179.2982426. 2

[CSH*25] CAPOUELLEZ R., SINGH R., HEISTERMANN M., BOMMES D., ZORIN D.: Feature-aligned parametrization in penner coordinates. *ACM Trans. Graph. 44*, 4 (July 2025). URL: https://doi.org/10.1145/3731216, doi:10.1145/3731216. 3, 10

[CSZZ19] CAMPEN M., SHEN H., ZHOU J., ZORIN D.: Seamless parametrization with arbitrary cones for arbitrary genus. *ACM Trans. Graph. 39*, 1 (Dec. 2019). URL: https://doi.org/10.1145/3360511, doi:10.1145/3360511. 3

[CZ24] CAPOUELLEZ R., ZORIN D.: Seamless parametrization in penner coordinates. *ACM Trans. Graph. 43*, 4 (July 2024). URL: https://doi.org/10.1145/3658202, doi:10.1145/3658202. 3

[DKZ*22] DU X., KAUFMAN D. M., ZHOU Q., KOVALSKY S., YAN Y., AIGERMAN N., JU T.: Isometric energies for recovering injectivity in constrained mapping. In *SIGGRAPH Asia 2022 Conference Papers* (New York, NY, USA, 2022), SA '22, Association for Computing Machinery. URL: https://doi.org/10.1145/3550469.3555419, doi:10.1145/3550469.3555419. 3

[DVPSH15] DIAMANTI O., VAXMAN A., PANOZZO D., SORKINE-HORNUNG O.: Integrable polyvector fields. *ACM Trans. Graph. 34*, 4 (July 2015). URL: https://doi.org/10.1145/2766906, doi:10.1145/2766906. 2, 3, 12, 13, 14

[EBCK13] EBKE H.-C., BOMMES D., CAMPEN M., KOBBELT L.: Qex: robust quad mesh extraction. *ACM Trans. Graph. 32*, 6 (Nov. 2013). URL: https://doi.org/10.1145/2508363.2508372, doi:10.1145/2508363.2508372. 10

[Exo24] EXOSIDE: Quad Remesher by exoside. https://exoside.com/, 2024. Software Tool. URL: https://exoside.com/. 1

[FH05] FLOATER M. S., HORMANN K.: Surface parameterization: a tutorial and survey. In *Advances in multiresolution for geometric modelling*, Dodgson N. A., Floater M. S., Sabin M. A., (Eds.). Springer Verlag, 2005, pp. 157–186. URL: http://vcg-legacy.isti.cnr.it. 2

[Flo03] FLOATER M. S.: One-to-one piecewise linear mappings over triangulations. *Mathematics of Computation 72*, 242 (2003), 685–696. 3

[Fra11] FRANKEL T.: *The Geometry of Physics: An Introduction*, 3 ed. Cambridge University Press, Cambridge, 2011. 4

[FSZ*21] FU X.-M., SU J.-P., ZHAO Z.-Y., FANG Q., YE C., LIU L.: Inversion-free geometric mapping construction: A survey. *Computational Visual Media 7*, 3 (Sept. 2021), 289–318. URL: https://doi.org/10.1007/s41095-021-0233-9, doi:10.1007/s41095-021-0233-9. 3

[GKK*21] GARANZHA V., KAPORIN I., KUDRYAVTSEVA L., PROTAIS F., RAY N., SOKOLOV D.: Foldover-free maps in 50 lines of code. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 102:1–102:16. doi:10.1145/3450626.3459847. 3

[GSC21] GILLESPIE M., SPRINGBORN B., CRANE K.: Discrete conformal equivalence of polyhedral surfaces. *ACM Trans. Graph. 40*, 4 (July 2021). URL: https://doi.org/10.1145/3450626.3459763, doi:10.1145/3450626.3459763. 3

[HRLL24] HAO Z., ROMERO D. W., LIN T.-Y., LIU M.-Y.: Meshtron: High-fidelity, artist-like 3d mesh generation at scale. *arXiv abs/2412.09548* (2024). URL: https://arxiv.org/abs/2412.09548. 2

[HWB23] HEISTERMANN M., WARNETT J., BOMMES D.: Min-deviation-flow in bi-directed graphs for t-mesh quantization. *ACM Trans.*

Graph. 42, 4 (July 2023). URL: https://doi.org/10.1145/3592437, doi:10.1145/3592437. 10

[HZ00] HERTZMANN A., ZORIN D.: Illustrating smooth surfaces. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., p. 517–526. URL: https://doi.org/10.1145/344779.345074, doi:10.1145/344779.345074. 1

[JCR24] JEZDIMIROVIĆ J., CHEMIN A., REMACLE J.-F.: Integrable cross-field generation based on imposed singularity configuration—the 2d manifold case. In *SIAM International Meshing Roundtable 2023* (Cham, 2024), Ruiz-Gironés E., Sevilla R., Moxey D., (Eds.), Springer Nature Switzerland, pp. 343–369. 2

[JFH*15] JIANG T., FANG X., HUANG J., BAO H., TONG Y., DESBRUN M.: Frame field generation through metric customization. *ACM Trans. Graph. 34*, 4 (July 2015). URL: https://doi.org/10.1145/2766927, doi:10.1145/2766927. 3

[JTPSH15] JAKOB W., TARINI M., PANOZZO D., SORKINE-HORNUNG O.: Instant field-aligned meshes. *ACM Transactions on Graphics (Proceedings of SIGGRAPH ASIA) 34*, 6 (Nov. 2015). doi:10.1145/2816795.2818078. 4

[KMZ10] KOVACS D., MYLES A., ZORIN D.: Anisotropic quadrangulation. In *Proceedings of the 14th ACM Symposium on Solid and Physical Modeling* (New York, NY, USA, 2010), SPM '10, Association for Computing Machinery, p. 137–146. URL: https://doi.org/10.1145/1839778.1839797, doi:10.1145/1839778.1839797. 3

[KNP07] KÄLBERER F., NIESER M., POLTHIER K.: Quad-cover - surface parameterization using branched coverings. *Computer Graphics Forum 26*, 3 (2007), 375–384. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2007.01060.x, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2007.01060.x, doi:https://doi.org/10.1111/j.1467-8659.2007.01060.x. 2

[LCBK19] LYON M., CAMPEN M., BOMMES D., KOBBELT L.: Parametrization quantization with free boundaries for trimmed quad meshing. *ACM Trans. Graph. 38*, 4 (July 2019). URL: https://doi.org/10.1145/3306346.3323019, doi:10.1145/3306346.3323019. 10

[Led20] LEDOUX F.: The mambo dataset. https://gitlab.com/franck.ledoux/mambo, 2020. 2, 4, 11

[Lev23a] LEVI Z.: Seamless parametrization with cone and partial loop control. *ACM Transactions on Graphics 42*, 4 (aug 2023). doi:10.1145/3600087. 3

[Lev23b] LEVI Z.: Shear-reduced seamless parametrization. *Comput. Aided Geom. Des. 101*, C (Apr. 2023). URL: https://doi.org/10.1016/j.cagd.2023.102179, doi:10.1016/j.cagd.2023.102179. 2

[LFO*22] LI M., FANG Q., OUYANG W., LIU L., FU X.-M.: Computing sparse integer-constrained cones for conformal parameterizations. *ACM Trans. Graph. 41*, 4 (jul 2022). URL: https://doi.org, doi:10.1145/3528223.3530118. 2

[Lim05] LIM L.-H.: Singular values and eigenvalues of tensors: a variational approach. In *1st IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP'05)* (2005), vol. 1, IEEE, pp. 129–132. 5

[Lip12] LIPMAN Y.: Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph. 31*, 4 (July 2012). URL: https://doi.org/10.1145/2185520.2185604, doi:10.1145/2185520.2185604. 3

[MC19] MANDAD M., CAMPEN M.: Exact constraint satisfaction for truly seamless parametrization. *Computer Graphics Forum 38*, 2 (2019), 135–145. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13625, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13625, doi:https://doi.org/10.1111/cgf.13625. 10

[MGTG07] MERHOF D., GROSSO R., TREMEL U., GREINER G.: Anisotropic quadrilateral mesh generation: An indirect approach. *Advances in Engineering Software 38*, 11 (2007), 860–867. Engineering Computational Technology. URL: https://www.sciencedirect.com/science/article/pii/S0965997806002158, doi:https://doi.org/10.1016/j.advengsoft.2006.08.036. 2

[MPZ14] MYLES A., PIETRONI N., ZORIN D.: Robust field-aligned global parametrization. *ACM Transactions on Graphics (TOG) 33*, 4 (jul 2014), 135:1–135:14. Proceedings of SIGGRAPH 2014. doi:10.1145/2601097.2601154. 2, 3, 11, 14

[MZ12] MYLES A., ZORIN D.: Global parametrization by incremental flattening. *ACM Trans. Graph. 31*, 4 (July 2012). URL: https://doi.org/10.1145/2185520.2185605, doi:10.1145/2185520.2185605. 2

[MZ13] MYLES A., ZORIN D.: Controlled-distortion constrained global parametrization. *ACM Trans. Graph. 32*, 4 (July 2013). URL: https://doi.org/10.1145/2461912.2461970, doi:10.1145/2461912.2461970. 2

[PAK07] POTTMANN H., ASPERL A., KILILAN A.: *Architectural Geometry*. Bentley Institute Press, Philadelphia, PA, 2007. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781934493045, arXiv:https://epubs.siam.org/doi/pdf/10.1137/1.9781934493045, doi:10.1137/1.9781934493045. 1

[PBS20] PALMER D., BOMMES D., SOLOMON J.: Algebraic representations for volumetric frame fields. *ACM Trans. Graph. 39*, 2 (Apr. 2020). URL: https://doi.org/10.1145/3366786, doi:10.1145/3366786. 2, 4, 5, 6, 9, 10

[PCS24] PALMER D., CHERN A., SOLOMON J.: Lifting directional fields to minimal sections. *ACM Trans. Graph. 43*, 4 (July 2024). URL: https://doi.org/10.1145/3658198, doi:10.1145/3658198. 2

[PP18] PELLIS D., POTTMANN H.: Aligning principal stress and curvature directions. In *Advances in Architectural Geometry* (2018). URL: https://api.semanticscholar.org/CorpusID:54078258. 1

[Qi07] QI L.: Eigenvalues and invariants of tensors. *Journal of Mathematical Analysis and Applications 325*, 2 (2007), 1363–1377. 5

[RHCB*13] REMACLE J.-F., HENROTTE F., CARRIER-BAUDOUIN T., BÉCHET E., MARCHANDISE E., GEUZAINE C., MOUTON T.: A frontal delaunay quad mesh generator using the l-infinity norm. *International Journal for Numerical Methods in Engineering 94*, 5 (2013), 494–512. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.4458, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.4458, doi:https://doi.org/10.1002/nme.4458. 2

[RLL*06] RAY N., LI W. C., LÉVY B., SHEFFER A., ALLIEZ P.: Periodic global parameterization. *ACM Trans. Graph. 25*, 4 (Oct. 2006), 1460–1485. URL: https://doi.org/10.1145/1183287.1183297, doi:10.1145/1183287.1183297. 2

[Rob16] ROBEVA E.: Orthogonal decomposition of symmetric tensors. *SIAM Journal on Matrix Analysis and Applications 37*, 1 (2016), 86–102. URL: https://doi.org/10.1137/140989340, arXiv:https://doi.org/10.1137/140989340, doi:10.1137/140989340. 2

[RPPSH17] RABINOVICH M., PORANNE R., PANOZZO D., SORKINE-HORNUNG O.: Scalable locally injective mappings. *ACM Trans. Graph. 36*, 2 (Apr. 2017). URL: https://doi.org/10.1145/2983621, doi:10.1145/2983621. 2

[SA24] SIMONS L., AMENTA N.: Anisotropy and cross fields.

Computer Graphics Forum 43, 5 (2024), e15132. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.15132, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.15132, doi:https://doi.org/10.1111/cgf.15132. 2

[SBB*22] SCHMIDT P., BORN J., BOMMES D., CAMPEN M., KOBBELT L.: TinyAD: Automatic differentiation in geometry processing made simple. *Computer Graphics Forum 41*, 5 (2022). doi:10.1111/cgf.14607. 10

[SFCBCV19] SAGEMAN-FURNAS A. O., CHERN A., BEN-CHEN M., VAXMAN A.: Chebyshev nets from commuting polyvector fields. *ACM Trans. Graph. 38*, 6 (Nov. 2019). URL: https://doi.org/10.1145/3355089.3356564, doi:10.1145/3355089.3356564. 1, 2

[SPR07] SHEFFER A., PRAUN E., ROSE K.: Mesh Parameterization Methods and Their Applications. *Foundations and Trends in Computer Graphics and Vision 2*, 2 (Feb. 2007), 105–171. _eprint: https://www.emerald.com/ftcgv/article-pdf/2/2/105/10878731/0600000011en.pdf. URL: https://doi.org/10.1561/0600000011, doi:10.1561/0600000011. 2

[SPSH*17] SHTENGEL A., PORANNE R., SORKINE-HORNUNG O., KOVALSKY S. Z., LIPMAN Y.: Geometric optimization via composite majorization. *ACM Trans. Graph. 36*, 4 (July 2017). URL: https://doi.org/10.1145/3072959.3073618, doi:10.1145/3072959.3073618. 2

[SS15] SMITH J., SCHAEFER S.: Bijective parameterization with free boundaries. *ACM Trans. Graph. 34*, 4 (July 2015). URL: https://doi.org/10.1145/2766947, doi:10.1145/2766947. 10

[SSC18] SOLIMAN Y., SLEPČEV D., CRANE K.: Optimal cone singularities for conformal flattening. *ACM Trans. Graph. 37*, 4 (2018). 2

[SZC*22] SHEN H., ZHU L., CAPOUELLEZ R., PANOZZO D., CAMPEN M., ZORIN D.: Which cross fields can be quadrangulated? global parameterization from prescribed holonomy signatures. *ACM Transactions on Graphics (TOG) 41*, 4 (jul 2022), 1–12. Proceedings of SIGGRAPH 2022. doi:10.1145/3528223.3530187. 3

[VCD*16] VAXMAN A., CAMPEN M., DIAMANTI O., PANOZZO D., BOMMES D., HILDEBRANDT K., BEN-CHEN M.: Directional field synthesis, design, and processing. *Computer Graphics Forum 35*, 2 (2016), 545–572. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12864, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12864, doi:https://doi.org/10.1111/cgf.12864. 2, 12

[ZHLB10] ZHANG M., HUANG J., LIU X., BAO H.: A wave-based anisotropic quadrangulation method. *ACM Trans. Graph. 29*, 4 (July 2010). URL: https://doi.org/10.1145/1778765.1778855, doi:10.1145/1778765.1778855. 2, 3

[ZVC*20] ZHANG P., VEKHTER J., CHIEN E., BOMMES D., VOUGA E., SOLOMON J.: Octahedral frames for feature-aligned cross fields. *ACM Trans. Graph. 39*, 3 (Apr. 2020). URL: https://doi.org/10.1145/3374209, doi:10.1145/3374209. 2, 4