Symmetric Moving Frames

ETIENNE CORMAN, Carnegie Mellon University / University of Toronto
KEENAN CRANE, Carnegie Mellon University

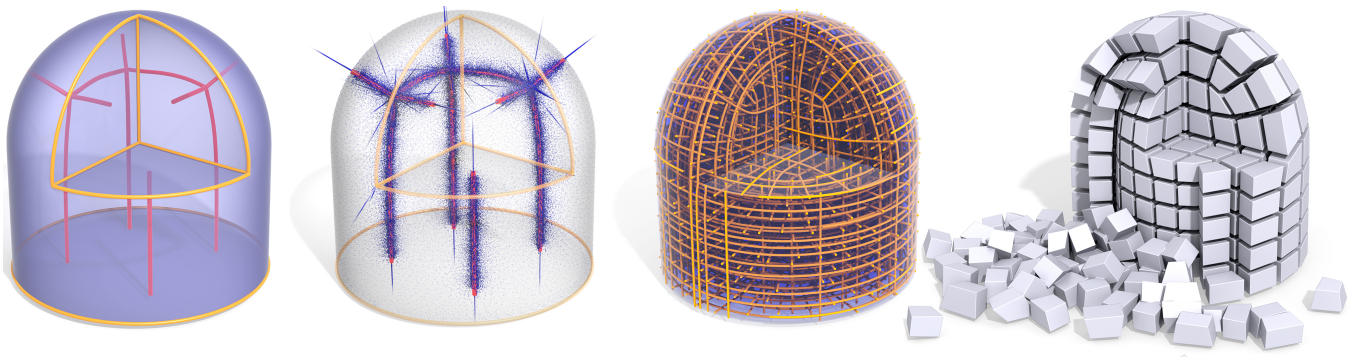


Fig. 1. Given a collection of singular and feature curves on a volumetric domain *(far left)*, we compute the smoothest rotational derivative that winds around these curves *(center left)*, and describes a symmetric 3D cross field *(center right)* which can be directly used for hexahedral meshing *(far right)*.

A basic challenge in field-guided hexahedral meshing is to find a spatially-varying frame that is adapted to the domain geometry and is continuous up to symmetries of the cube. We introduce a fundamentally new representation of such *3D cross fields* based on Cartan's method of moving frames. Our key observation is that cross fields and ordinary frame fields are locally characterized by identical conditions on their *Darboux derivative*. Hence, by using derivatives as the principal representation (and only later recovering the field itself), one avoids the need to explicitly account for symmetry during optimization. At the discrete level, derivatives are encoded by skew-symmetric matrices associated with the edges of a tetrahedral mesh; these matrices encode arbitrarily large rotations along each edge, and can robustly capture singular behavior even on coarse meshes. We apply this representation to compute 3D cross fields that are as smooth as possible everywhere but on a prescribed network of singular curves—since these fields are adapted to curve tangents, they can be directly used as input for field-guided mesh generation algorithms. Optimization amounts to an easy nonlinear least squares problem that behaves like a convex program in the sense that it always appears to produce the same result, independent of initialization. We study the numerical behavior of this procedure, and perform some preliminary experiments with mesh generation.

CCS Concepts: • **Mathematics of computing → Mesh generation**.

Additional Key Words and Phrases: cross fields, discrete differential geometry

ACM Reference Format:
Etienne Corman and Keenan Crane. 2019. Symmetric Moving Frames. *ACM Trans. Graph.* 38, 4, Article 87 (July 2019), 16 pages. https://doi.org/10.1145/3306346.3323029

Authors' addresses: Etienne Corman, Carnegie Mellon University / University of Toronto; Keenan Crane, Carnegie Mellon University, Pittsburgh, PA, 15213.

0730-0301/2019/7-ART87 $15.00
https://doi.org/10.1145/3306346.3323029

0 INTRODUCTION

A *hexahedral mesh* decomposes a solid region of three-dimensional space into six-sided cells; such meshes play an important role in numerical algorithms across geometry processing and scientific computing. An attractive approach to mesh generation is to first construct a *guidance field* oriented along features of interest, then extract a mesh aligned with this field. However, there are major open questions about how to even *represent* such fields in a way that is compatible with the demands of hexahedral meshing—the most elementary of which is how to identify frames that differ by rotational symmetries of the cube. These so-called *3D cross fields* allow one to encode networks of singular features (Fig. 1, far left), which are critical to achieving good element quality.

In differential geometry, *Cartan's method of moving frames* provides a rich theory for spatially-varying coordinate frames, but to date has not been used for hexahedral meshing—perhaps because, classically, it does not consider fields with local rotational symmetry (like cross fields). In this paper we show how the theory of moving frames can be naturally applied in the symmetric case, and how to incorporate constraints needed for hexahedral meshing, namely, adaptation to a network of *singular curves* which correspond to mesh edges of irregular degree. Specifically, we consider the following problem: given a domain and a valid singularity network, find the smoothest 3D cross field compatible with this network. Here, a *valid* network means one that is compatible with the global topology of some hexahedral mesh, as recently studied by Liu et al. [2018].

Computationally, our method amounts to solving an augmented version of *Cartan's second structure equation*

$$\boxed{d\omega - \omega \wedge \omega = 0.}$$

Much as the curl-free condition $\nabla \times X = 0$ characterizes vector fields X that can be locally expressed as the gradient of a scalar potential, the structure equation characterizes differential 1-forms ω which are the *Darboux derivative* of some spatially-varying frame field.

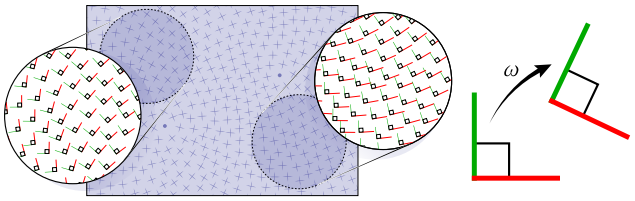


Fig. 2. Within small neighborhoods, a 2D or 3D cross field can be represented by an ordinary frame field. Its derivatives ω therefore obey standard *structure equations*, which provide the basic constraints for our method.

Our key insight is that—at least locally—*the derivative of a cross field looks no different from the derivative of an ordinary frame field.* To generate cross fields we can therefore optimize the derivatives, without having to encode explicit "jumps" or enumerate all possible rotations. The final field is recovered via local integration, which amounts to a simple breadth-first traversal of the domain. This field can in turn be used as direct input to parameterization-based meshing tools, yielding high-quality pure hexahedral meshes with precise control over singular features (Fig. 1, *far right*).

0.1 Related Work

This paper is concerned solely with the representation and generation of 3D cross fields. A discussion of broader hexahedral meshing is covered by several recent surveys [Armstrong et al. 2015; Yu et al. 2015]; the specific problem of finding meshable fields with prescribed singularities is nicely motivated by Liu et al. [2018].

0.1.1 Moving Frames. Familiar examples of moving frames include the *Frenet frame* of a space curve, and the *Darboux frame* of a surface patch; these so-called *adapted frames* arise naturally in applications ranging from elastic rod simulation [Bergou et al. 2008] to geometric design [Pan et al. 2015]. Richer elements of the theory have seen little use in computer graphics: Lipman et al. [2005, 2007] consider a surface representation similar in spirit to moving frames but do not directly discretize the structure equations; moreover, these methods have no reason to consider volumetric domains or singular cross fields, as are needed for hexahedral meshing. More broadly, specialized numerical treatments of moving frames have been applied sporadically to problems ranging from general relativity to integrable systems theory [Olver 2000; Frauendiener 2006; Mansfield et al. 2013], though none are suitable for the problem at hand.

0.1.2 Direction Field Representations. For surfaces, representation of symmetric direction fields is fairly well understood—see surveys by Vaxman et al. [2016] and de Goes et al. [2016]. However, due to non-commutativity of 3D rotations many of these representations do not easily generalize to volumes, or lead to optimization problems that are difficult to solve. Moreover, whereas singularities in a 2D cross field can always be realized as irregular vertices in a quad mesh, singularities in a 3D cross field cannot always be realized as irregular edges in a hexahedral mesh since the field direction may not be tangent to the singular curve (Fig. 3). Therefore, although there are many methods for generating *smooth* 3D cross fields, almost none produce fields directly suitable for meshing.

Periodic Functions. Early methods used periodic, sinusoidal functions to capture the 4-fold rotational symmetry of 2D cross fields [Hertzmann and Zorin 2000, Sec. 5]; likewise, several 3D methods use a function with cube symmetry expressed as a sum of low-frequency spherical harmonics [Huang et al. 2011], or equivalent polynomials [Li et al. 2012]. Ensuring that spherical harmonic coefficients correspond to *rotations* of this function entails high-degree nonlinear constraints in a large number of variables—moreover, since optimization can easily get stuck in local minima, practical success of such methods appears to depend strongly on careful initialization of the field [Li et al. 2012; Ray et al. 2016].

Representation Vectors. Symmetric fields can also be expressed as a set of vectors at each point; in 2D, one can identify all elements of this set with a single symmetric tensor [Palacios and Zhang 2007], or a single complex number via the identification $z \mapsto z^k$ [Knöppel et al. 2013]. In 3D, there does not appear to be any easy analogue—for instance, 3D symmetric tensor fields do not exhibit the symmetry needed for hex meshing [Palacios et al. 2017], and powers of *quaternions* identify rotations only around a single axis. Alternatively, one can retain the full set of vectors and enumerate all possible rotations during optimization, necessitating iterative local smoothing that easily gets trapped in local minima [Gao et al. 2017].

Period Jumps. In 2D, cross fields can be encoded via angles $\theta \in \mathbb{R}$, together with integer *period jumps* or *matchings* $n \in \mathbb{Z}$ which encode identifications between equivalent angles (*e.g.*, $\theta_1 = \theta_2 + n\pi/2$). Optimization then entails *mixed integer programming* [Bommes et al. 2009], which in general is NP-hard. Liu et al. [2018] develop the first such approach for 3D cross fields; like our method (and unlike all other methods discussed so far) they ensure that fields are compatible with the structure of a hexahedral mesh, providing direct control over singularities. To determine period jumps, the method solves a large system of nonlinear mixed integer equations; in the worst case, it resorts to exhaustive search over the entire solution tree. It then finds the smoothest cross field by relaxing a unit-norm constraint on individual quaternions to a principal eigenvector problem over the entire domain. Implementation involves intricate *merging* and *zippering* procedures; moreover, the eigenvector relaxation does not directly measure the smoothness of rotations (Sec. 4.4), and can in principle introduce new singularities (zeros) that were not part of the given network—see discussion of Vaxman et al. [2016, Figure 6].

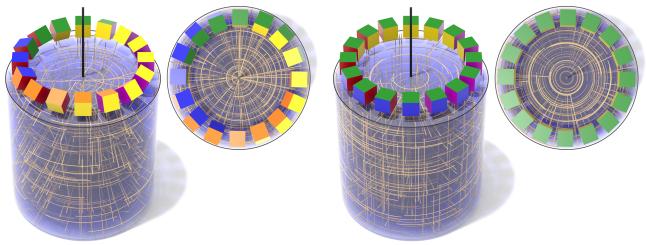


Fig. 3. Control over the behavior of singularities is essential, since even extremely smooth fields *(left)* may not be meshable. Using symmetric moving frames, we can ensure that one frame axis is always tangent to a given singular curve *(right)*, without having to determine this axis *a priori*.

Differential Representations. Our representation naturally generalizes Crane et al. [2010], who optimize the *derivative* ω of a 2D cross field rather than the field itself. This approach avoids the need to explicitly identify equivalent frames during optimization, leading to a convex problem easily solved via a sparse linear system. The symmetric nature of the field arises purely from the fact that the derivative may describe only *quarter* turns around closed loops (Fig. 4). The only challenge is ensuring that ω is *integrable*, *i.e.*, that it really is the derivative of some cross field. In 2D, integrability is enforced via a simple linear structure equation where $\omega \wedge \omega = 0$; in 3D we must discretize the full structure equation, and consider singularities which are now curves rather than isolated points (Sec. 3).

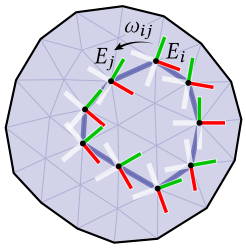


Fig. 4. A 2D cross field encoded by the change in angle ω across each edge.

0.1.3 Contributions. Our main contributions are to (i) cast the problem of symmetric 3D cross field generation in the language of moving frames, and to (ii) develop a principled discretization of moving frames suitable for hexahedral meshing. In doing so, we build a bridge between a rich body of knowledge from differential geometry and the difficult computational challenge of mesh generation; connections to established partial differential equations (PDEs) enable us to build on principled numerical foundations for discrete differential forms [Hirani 2003; Desbrun et al. 2006]. Ultimately, we obtain a simple, practical representation where cross fields are encoded by an axis and angle of rotation across each edge. This representation captures arbitrarily large rotations even on coarse meshes (Fig. 9), and leads to a natural notion of field smoothness that considers only orientation, rather than magnitude (Sec. 3.1).

Our main application is computing 3D cross fields adapted to a given singularity network; preserving these singularities is critical for ensuring that the field can actually be meshed. Computational cost is dominated by a sparse nonlinear least squares problem arising from equations that are at most quadratic and have no integer variables; such problems are easily solved using a small number of linear solves, or scalable iterative solvers [DeVito et al. 2017]. In practice this problem behaves like a convex program: it produces the same result independent of initialization, and yields only the requested singularities (Sec. 4.2). Representations that encode both direction *and* magnitude are unattractive for this task since magnitudes may go to zero (yielding unwanted singularities), or may get stuck in local minima that do not exhibit the desired singularities. Moreover, while Liu et al. [2018] must make all topological decisions *a priori* (*e.g.*, total torsion around closed loops), our formulation allows these choices to emerge naturally from the optimization of a simple geometric energy (Sec. 3.2.2). See Sec. 4.4 for further comparisons.

0.2 Overview

Our algorithm can be broken down into two major steps:

- first find a 2D cross field on the domain boundary compatible with the prescribed singularity network;
- then find a 3D cross field on the interior adapted to both the singularities and the boundary normal.

Note that, as in Crane et al. [2010], we do not aim to solve the problem of *finding* singularities, but assume that a valid, "meshable" network is provided as input (in the sense of Liu et al. [2018]). Since both steps amount to solving very similar *structure equations*, we begin with a unified treatment of 2D and 3D discretization in Sec. 1 before describing the boundary (2D) and volume (3D) optimization problems (Secs. 2 and 3). App. A motivates this algorithm from the smooth perspective; numerical experiments can be found in Sec. 4.

0.3 Background

Throughout we use $\mathrm{SO}(n)$ to denote the collection of $n \times n$ rotation matrices $\mathsf{Q}^\mathsf{T}\mathsf{Q} = \mathsf{Q}\mathsf{Q}^\mathsf{T} = \mathsf{I}$, $\det(\mathsf{Q}) > 0$, where I is the identity. We use $\mathfrak{so}(n)$ to denote $n \times n$ skew-symmetric matrices $\mathsf{A}^\mathsf{T} = -\mathsf{A}$, whose nonzero components describe the axis and magnitude of a rotation. The corresponding rotation matrix is obtained via the *exponential map* $\exp : \mathfrak{so}(n) \to \mathrm{SO}(n)$. For instance, every $\mathsf{A} \in \mathfrak{so}(2)$ is determined by a single angle θ, and we have the relationship

$$\begin{bmatrix} 0 & \theta \\ -\theta & 0 \end{bmatrix} \overset{\exp}{\longmapsto} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}. \tag{1}$$

In 3D, a unit axis $u = (u_1, u_2, u_3) \in \mathbb{R}^3$ and angle $\theta \in \mathbb{R}$ determines a skew-symmetric matrix $\mathsf{A} = \theta\hat{u}$, where

$$\hat{u} := \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix}. \tag{2}$$

The exponential map can then be evaluated via *Rodrigues' formula*

$$\exp(\mathsf{A}) = \mathsf{I} + \sin\theta\hat{u} + (1 - \cos\theta)\hat{u}^2.$$

Importantly, the exponential map is not one-to-one: as the angle increases, exp will return to the same rotation many times. For a given $\mathsf{Q} \in \mathrm{SO}(n)$, the *logarithmic map* $\log : \mathrm{SO}(n) \to \mathfrak{so}(n)$ gives the *smallest* matrix A such that $\exp(\mathsf{A}) = \mathsf{Q}$, and can be evaluated via the matrix logarithm. Throughout we will use the notation $\hat{\theta}$ and $\hat{u}$ to identify angles and vectors with skew-symmetric matrices, as in Eqn. 1 and 2 (*resp.*); we will use $|\mathsf{M}|^2 := \sum_{i,j=1}^{n} \mathsf{M}_{ij}^2$ to denote the (squared) *Frobenius norm* of any $n \times n$ matrix M.

1 DISCRETIZATION

Our main object is a (2D or 3D) cross field E and its *Darboux derivative* ω, which encodes the change in the field from one point to another. In 2D, integrable Darboux derivatives are characterized by linear equations describing the consistency of rotations around closed loops [Crane et al. 2010]. In 3D, the chief difficulty is that exponentiation of rotations no longer obeys the familiar relationship

$$\exp(A)\exp(B) = \exp(A + B), \tag{3}$$

which means we can no longer convert statements about products of rotations into corresponding linear equations. To obtain efficient algorithms, we instead approximate exact but nonlinear discrete integrability conditions by a truncated series expansion (Sec. 3.1). Remarkably enough, this approximation turns out to be *identical* to a principled discretization of the structure equation via discrete exterior calculus (App. A.4), which in practice exhibits highly accurate (second-order) enforcement of integrability (Sec. 4.2).

1.1 Domain

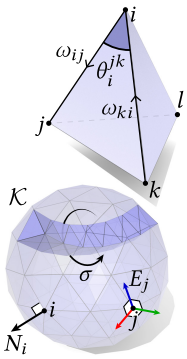


Fig. 5. Quantities associated with the tetrahedral mesh $\mathcal{K}$.

The domain is represented by a connected, manifold tetrahedral mesh $\mathcal{K}$ embedded in $\mathbb{R}^3$. We use $\mathcal{K}_k$ to denote the k-simplices of $\mathcal{K}$ (*e.g.*, $\mathcal{K}_0$ is the set of vertices). We likewise use $\partial\mathcal{K}$ to denote the boundary surface, and $\partial\mathcal{K}_k$ to denote the k-simplices contained in $\partial\mathcal{K}$. Ordered lists of vertex indices denote oriented simplices, *e.g.*, $ij \in \mathcal{K}_1$ is an edge from vertex i to vertex j, and ji is the same edge but with opposite orientation. Sums are implicitly restricted to simplices containing vertices that appear on both the left- and right-hand side of an expression—for instance, $A_i := \frac{1}{3}\sum_{ijk\in\partial\mathcal{K}_2} A_{ijk}$ defines the barycentric dual area obtained by taking one-third the area of triangles ijk containing vertex i. We use N_i to denote the unit area-weighted vertex normal at any boundary vertex $i \in \partial\mathcal{K}_0$, θ_i^{jk} to denote the interior angle at vertex i of triangle $ijk \in \mathcal{K}_2$, and ℓ_{ij} to denote the length of edge $ij \in \mathcal{K}_1$.

1.1.1 Singularity Tubes. In 2D, cross fields can have isolated *singular points* where the direction is undefined, and around which the field "spins" at a prescribed rate; 3D fields can likewise have networks of *singular curves* that form closed loops or terminate at the boundary. To represent such networks, we use a mesh $\mathcal{K}$ with a special structure: singular curves are represented by tubes of triangular prisms (Fig. 5, *bottom*) which terminate at boundary triangles or meet at interior tetrahedra (see Sec. 4.1 for details).

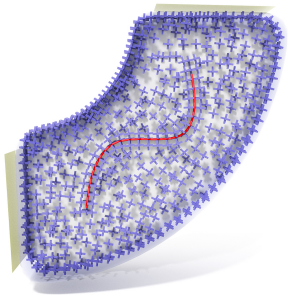


Fig. 6. Field with a *feature curve* (red) and boundary constraints (yellow).

Each curve has an *index* $\sigma \in \mathbb{R}$ which determines how many times the field rotates as it goes around the tube (Fig. 9); meshable cross fields can have fractional indices $\sigma = \pm n/4$ for $n \in \mathbb{Z}$. An index $\sigma = 0$ specifies a *feature curve*, along which the cross field is tangent but not singular (Fig. 6). In order to be meshable, indices must at least satisfy a condition analogous to *Poincaré-Hopf*, given in Liu et al. [2018, Equation 2], and interior nodes must exhibit configurations described in Liu et al. [2018, Sec. 3] (who note that global necessary *and* sufficient conditions remain difficult to establish.)

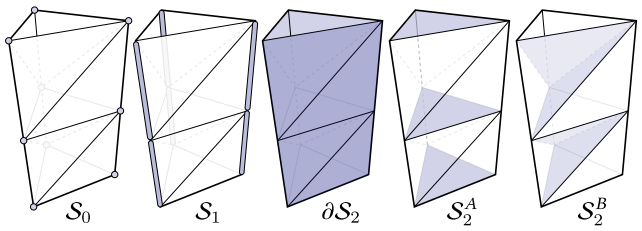


Fig. 7. Notation used to refer to elements of the singularity tubes.

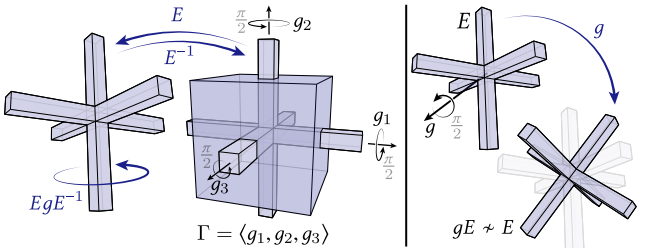


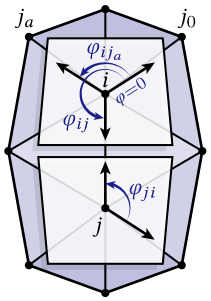
Fig. 8. *Left:* Two crosses are equivalent only if they differ by quarter rotations around their own axes. Such motions correspond to inverting the current rotation (E^{-1}), applying some symmetry of the standard cube (g), then applying the original rotation (E). *Right:* Simply applying a quarter rotation around an *arbitrary* axis generally yields a different cross.

Notation. We use $\mathcal{S}_0 \subset \mathcal{K}_0$ to denote the set of vertices contained in the singular tubes, $\mathcal{S}_1$ to denote edges running along their length, $\partial\mathcal{S}_2$ to denote triangles on tube boundaries, $\mathcal{S}_2^A$ to denote triangles at the top/bottom of triangular prisms, and $\mathcal{S}_2^B$ to denote all other interior triangles, which serve only to triangulate the tube (Fig. 7).

1.2 Discrete Cross Field

Let $\Gamma \subset \mathrm{SO}(n)$ denote the set of rotations that map the n-dimensional cube $[-1,1]^n \subset \mathbb{R}^n$ to itself. Two rotations $E_i, E_j \in \mathrm{SO}(n)$ are equivalent up to cube symmetry if their difference $E_jE_i^{-1}$ equals $E_igE_i^{-1}$ for some $g \in \Gamma$, as depicted in Fig. 8. A *cross* is then an equivalence class of rotations, and a *discrete cross field* is a cross at each vertex i, encoded by a representative rotation $E_i \in \mathrm{SO}(n)$. In 3D, these values encode rotations of the standard basis, and the cross axes are given by the columns (not the rows) of the rotation matrix. In 2D, they encode rotations of a canonical tangent frame at each vertex (Sec. 1.3). Since cubes and octahedra have the same symmetry, 3D cross fields are also sometimes called *octahedral fields*.

1.3 Boundary Coordinate Systems



To encode the boundary (2D) frame field we adopt the approach of Knöppel et al. [2013], who express tangent vectors in local polar coordinates (r, φ) relative to some local coordinate system at each vertex (see inset). We first define normalized interior angles

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

where $\Theta_i := \sum_{ijk\in\partial\mathcal{K}_2} \theta_i^{jk}$. At each vertex $i \in \partial\mathcal{K}_0$, we then assign the angle $\varphi = 0$ to a fixed reference edge $ij_0 \in \partial\mathcal{K}_1$. The angles of all other edges $ij_1, \ldots, ij_a$ are given by partial sums of the augmented angles $\tilde{\theta}$:

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

The augmented angles also provide a definition of Gaussian curvature per boundary triangle $ijk \in \partial\mathcal{K}_2$, given by the deviation from the angle sum of a standard Euclidean triangle:

$$K_{ijk} := \tilde{\theta}_i^{jk} + \tilde{\theta}_j^{ki} + \tilde{\theta}_k^{ij} - \pi. \tag{4}$$

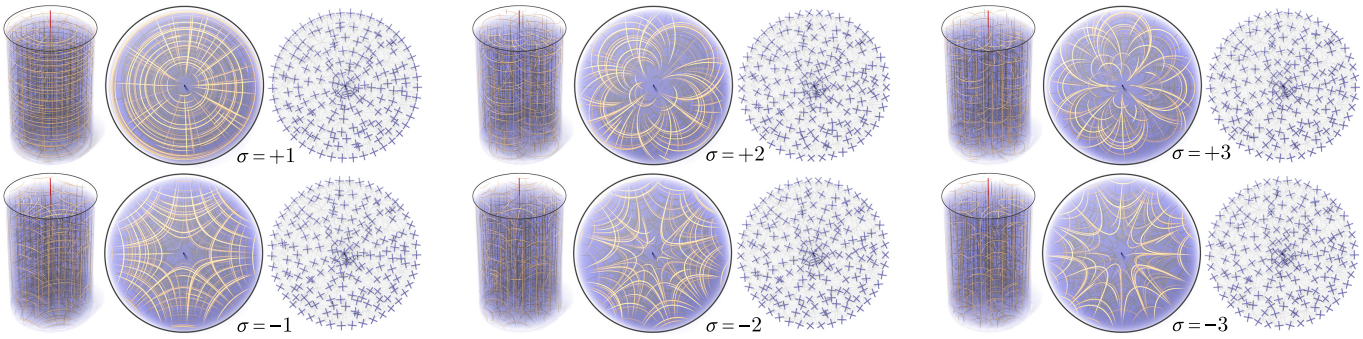


Fig. 9. The index σ determines how many times the field winds around a singular curve. Since we directly encode the angular change along each edge, we can robustly handle large rotations even on very coarse meshes.

1.4 Parallel Transport

To compare frames at neighboring vertices, we use matrices $R_{ij} \in \mathrm{SO}(n)$ that encode the change in local coordinates as we go from i to j. For the volume (3D) field, all rotations are expressed in the same basis, and hence $R_{ij} = \mathsf{I}$. For the 2D (boundary) field, let

$$\rho_{ij} := (\varphi_{ji} + \pi) - \varphi_{ij} \tag{5}$$

be the difference between the two angles encoding the shared edge ij. The rotation $R_{ij} = \exp(\hat{\rho}_{ij})$ then describes the process of *parallel transport*, *i.e.*, moving along ij without unnecessary "twisting." (Note that $\rho_{ji} = -\rho_{ij}$, and hence $R_{ji} = R_{ij}^{-1}$.) In general, parallel transport of a frame from i to j can be expressed as $E_i \mapsto R_{ij}E_i$.

An important relationship between curvature and parallel transport is nicely preserved by the discretization from Sec. 1.3, namely, the net rotation around any triangle $ijk \in \partial\mathcal{K}_2$ is determined by its total Gaussian curvature:

$$R_{ki}R_{jk}R_{ij} = \exp(\widehat{K}_{ijk}). \tag{6}$$

This relationship, and a corresponding index theorem (discussed carefully in Knöppel et al. [2013, Appendix B]), will enable us to formulate a precise version of the *trivial connections* algorithm of Crane et al. [2010] with frames at vertices rather than faces (Sec. 2).

1.5 Discrete Darboux Derivative

A discrete frame field is determined up to global rotation by the *change* across each edge. Inspired by the theory of moving frames, we will express this change *relative* to the frame itself. In particular, we define the *(discrete) Darboux derivative* along edge ij as

$$\omega_{ij} := \log(E_j(R_{ij}E_i)^{-1}), \tag{7}$$

i.e., as the (smallest) "axis-angle" representation of the change from E_i to E_j, taking parallel transport into account (see also App. A.4). For a cross field, we let E_j be the representative rotation closest to $R_{ij}E_i$. Although we use the *smallest* difference when taking the derivative of a given field E, in general we will allow ω_{ij} to have any magnitude, permitting very large rotations (Fig. 9).

1.5.1 Discrete Integrability. The Darboux derivative ω describes how a given frame E changes across each edge. We can also ask the opposite question: given values $\omega_{ij} \in \mathfrak{so}(n)$ at edges, do there exist frames E_i at vertices whose Darboux derivative is equal to ω? Any such frame is called a *development* of ω. One can clearly develop ω along any simple open path $\gamma = (i_0, \ldots, i_N)$: start with some initial frame E_{i_0}, and use parallel transport to obtain the development

$$E_{i_{p+1}} = \exp(\omega_{i_p, i_{p+1}})R_{i_p, i_{p+1}}E_{i_p}. \tag{8}$$

However, if γ is a closed loop, there is no reason the final frame must be equal to the initial one. In this case, ω does not describe a well-defined frame field, no matter how we pick E_{i_0}. More generally, for a field to be well-defined over the whole mesh, ω must be consistent around *every* closed loop of edges. The *(discrete) monodromy* Φ_ω quantifies the failure of this condition around a given loop γ:

$$\Phi_\omega(\gamma) = \exp(\omega_{i_N, i_0})R_{i_N, i_0}E_N E_0^{-1} \tag{9}$$

(where E_N is defined by Eqn. 8). The values ω then describe an ordinary frame field if and only if $\Phi_\omega(\gamma) = \mathsf{I}$ for all closed loops γ.

1.5.2 Monodromy of Cross Fields. In a 2D or 3D cross field, the total rotation around a closed loop no longer needs to be equal to the identity: instead, it can look like a symmetry of the square or cube (*resp.*). More precisely, let $E_i \in \mathrm{SO}(3)$ be any rotation representing a cross at vertex i, and let γ be a closed loop based at i. In order for ω to be the Darboux derivative of a cross field, the monodromy around γ must be *conjugate* to a cube symmetry, *i.e.*,

$$\Phi_\omega(\gamma) = E_i g E_i^{-1} \tag{10}$$

for some $g \in \Gamma$. If this condition holds, we say that ω has *trivial (Γ)-monodromy* around γ, with respect to E_i.

In 2D, Eqn. 10 is equivalent to simply asking that the monodromy is an element of Γ, since here rotations commute and $E_i g E_i^{-1} = E_i E_i^{-1} g = g$. But in 3D, merely asking that monodromy be an *element* of Γ is not the right condition, as illustrated in Fig. 8: a rotation that preserves a cross must be around the axes of the cross itself, not the axes of the canonical cube. From here it is easy to show that if Eqn. 10 is satisfied for some loop around each triangle (for some fixed choice of cross field), then it is automatically satisfied around all contractible loops; if it also satisfied around a collection of generators for the first fundamental group, then it is satisfied around all closed loops. This observation provides a discrete analogue of the fundamental theorem discussed in App. A.2.1 and A.3.

2 BOUNDARY CROSS FIELD (2D)

The first step of our algorithm is to solve an optimization problem for a 2D cross field on the boundary surface, which provides boundary data for our 3D problem (Sec. 3). The algorithm is essentially the one described by Crane et al. [2010], with two important modifications: first, we store frames at vertices rather than faces; second, singularities on the boundary are determined by the singular and feature curves of our 3D problem.

2.1 Objective (2D)

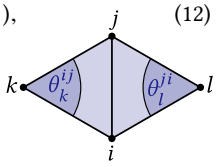
In 2D, the only objective term is the squared norm

$$||\omega||^2 := \sum_{ij \in \partial\mathcal{K}_1} w_{ij} |\omega_{ij}|^2. \tag{11}$$

Since ω encodes the deviation from ordinary parallel transport, Eqn. 11 encourages the field to be "as parallel as possible." The values w_{ij} are the standard *cotan weights*

$$w_{ij} := \tfrac{1}{2}(\cot \theta_k^{ij} + \cot \theta_l^{ji}), \tag{12}$$

where k, l are the vertices opposite edge ij on the boundary mesh $\partial\mathcal{K}$. Eqn. 11 discretizes an SO(2)-valued *Dirichlet energy*—see App. A.4 for further discussion.



2.2 Constraints (2D)

As discussed in Sec. 1.5.2, ω encodes a field E as long as it has trivial monodromy around all closed loops. This condition is enforced via linear constraints mirroring those from Crane et al. [2010, Sec. 3.3].

Local Integrability. Recall that parallel transport around a triangle $ijk \in \partial\mathcal{K}_2$ yields a change in angle determined by the Gaussian curvature K_{ijk} (Eqn. 6). To consistently describe an ordinary frame field on ijk, ω must *cancel* this deviation, *i.e.*, we must have

$$\omega_{ij} + \omega_{jk} + \omega_{ki} = -\widehat{K}_{ijk} + 2\pi\widehat{\sigma}_{ijk}, \tag{13}$$

for some integer $\sigma_{ijk} \in \mathbb{Z}$. This condition also permits some number of whole turns $\Omega_{ijk} := 2\pi\sigma_{ijk}$, corresponding to a singularity at ijk. For cross fields, σ_{ijk} can be a multiple of $\pi/2$ (describing quarter turns) rather than a whole integer—any cross transported around a contractible loop γ will then be indistinguishable from the initial cross (Fig. 4). The only requirement is that the prescribed indices satisfy a discrete *Poincaré-Hopf condition* $\sum_{ijk \in \partial\mathcal{K}_2} \sigma_{ijk} = \chi$, where χ is the Euler characteristic of the boundary surface.

Nonsimply-Connected Surfaces. Let be $\eta_1, \ldots, \eta_r$ a collection of generating cycles for the fundamental group (as depicted in the inset). To ensure that ω has trivial monodromy around non-contractible loops, we apply linear constraints

$$\sum_{ij \in \eta_p} \omega_{ij} = -\Phi_0(\eta_p) \tag{14}$$

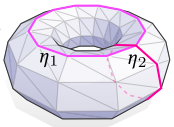


Fig. 10. Generators on a torus.

which cancel the monodromy $\Phi_0(\eta_p)$ due to parallel transport (*i.e.*, just the sum of values $\hat{\rho}_{ij}$ along η_p). The only change from Crane et al. [2010, Sec. 2.1] is that these generators are now paths along ordinary (primal) edges; in practice we use *tunnel and handle loops* computed via [Dey et al. 2013].

2.3 Optimization Problem (2D)

Overall, we obtain an optimization problem for the smoothest 2D cross field with prescribed singularities (and generator monodromy):

$$\begin{aligned} &\min_{\omega : \partial\mathcal{K}_1 \to \mathfrak{so}(2)} ||\omega||^2 \\ &\text{s.t. } \omega_{ij} + \omega_{jk} + \omega_{ki} = -\widehat{K}_{ijk} + \widehat{\Omega}_{ijk}, \qquad \forall ijk \in \partial\mathcal{K}_2, \\ &\quad \textstyle\sum_{ij \in \eta_p} \omega_{ij} = -\Phi_0(\eta_p), \qquad p \in \{1, \ldots, r\}. \end{aligned} \tag{15}$$

In practice we encode all values by real angles, yielding a convex quadratic program whose solution is described by a linear system (see [Crane et al. 2010, Sec. 2.4] and [Crane et al. 2013, Sec. 8.4.1]).

Singular Points and Sharp Features. To ensure the 2D field is compatible with the 3D curve network, we set $\Omega_{ijk} = 2\pi\sigma_{ijk}$ for any singularity tube of index σ terminating at a boundary triangle $ijk \in \partial\mathcal{K}_2$. For domains with sharp features (such as the edge of a cube), one can also specify a graph of sharp edges—this graph can be interpreted as the skeleton of a surface where all faces are axis-aligned, and all nonzero dihedral angles are equal to $\pm\pi/2$. The value of Ω_{ijk} at any vertex of this graph is then the angle defect of the axis-aligned surface; since we put singularities at triangles, sharp corners are replaced with a small singular triangle (Fig. 11). Singular curves that do not touch the boundary (*e.g.*, a loop around a solid torus) have no impact on boundary singularities, and all other values of Ω are set to zero.

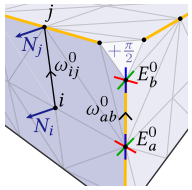


Fig. 11. Data along sharp (yellow) feature curves.

2.4 Field Integration (2D)

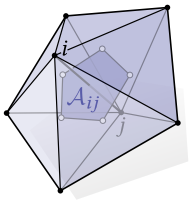
To obtain the final frames we perform a breadth-first traversal: starting at any vertex $i_0 \in \partial\mathcal{K}_0$, we transport some initial frame $E_i \in \mathrm{SO}(2)$ to all other boundary vertices via Eqn. 8. (The particular choice of frame has no effect on our 3D problem, which only uses the frame derivatives.) The constraints in Eqn. 15 ensure that the change in the resulting field E across any edge ij exactly agrees with ω_{ij}, independent of the starting point i_0. In this sense, the 2D theory is "exact": any ω satisfying our constraints exactly describes a 2D cross field, up to a global rotation.

Extrinsic Field. For the 3D problem, we will need an *extrinsic* version of the 2D field, *i.e.*, an element $E_i^0 \in \mathrm{SO}(3)$ for each boundary vertex $i \in \partial\mathcal{K}_0$, which we obtain by projecting each 2D frame onto the plane of the vertex normal N_i, and using N_i to complete the orthonormal basis. We also store the Darboux derivative ω^0 of the extrinsic field on each edge $ij \in \partial\mathcal{K}_1$. Evaluating the discrete Darboux derivative $\log(E_j^0 (E_i^0)^{-1})$ directly is not satisfactory since (i) it may exhibit spurious quarter rotations across edges not in the breadth-first tree, and (ii) the log map may not properly account for large rotations. Instead, we construct the smallest rotation $Q_{ij} \in \mathrm{SO}(3)$ from N_i to N_j, then set $\omega_{ij}^0 = \log(Q_{ij}) + \omega_{ij} N_i$ (using the triangle rather than vertex normal for edges with one endpoint on a sharp feature—see edge ij in Fig. 11). This value encodes a twist-free change of tangent plane, plus a (potentially very large) rotation $\omega_{ij} N_i$ around the normal. The final values of E^0 and ω^0 are the only data we need for the 3D stage of the algorithm.

3 VOLUME CROSS FIELD (3D)

To obtain the 3D cross field, we minimize an energy that measures (i) the smoothness of values $\omega : \mathcal{K}_1 \to \mathfrak{so}(3)$ and (ii) their failure to be integrable, subject to linear constraints that adapt the field to the boundary and the singular curve network. Note that we do not adapt *all three* directions of the 3D frame to the given (2D) boundary frame—we ask only that it preserve the 2D singularities (Sec. 3.2.1).

3.1 Objective (3D)



Field Smoothness. As in 2D, smoothness is quantified via

$$||\omega||^2 := \sum_{ij \in \mathcal{K}_1} w_{ij} |\omega_{ij}|^2, \qquad (16)$$

which measures the Dirichlet energy of the field (App. A.4). The weights are now given by $w_{ij} = \mathcal{A}_{ij}/\ell_{ij}$, where ℓ_{ij} is the length of edge ij, and $\mathcal{A}_{ij}$ is the area of its circumcentric dual face (see inset). This energy is particularly appropriate for field-guided meshing since it considers only smoothness in orientation and not magnitude.

Local Integrability. The 3D analogue of Eqn. 13 is given by the *discrete structure equation*

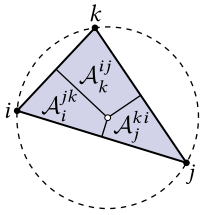
$$(d\omega)_{ijk} = (\omega \wedge \omega)_{ijk} + \Omega_{ijk}, \qquad (17)$$

Here d denotes the *discrete exterior derivative*

$$(d\omega)_{ijk} := \omega_{ij} + \omega_{jk} + \omega_{ki},$$

and the symbol $\wedge$ denotes the *discrete wedge product*

$$(\alpha \wedge \beta)_{ijk} := \frac{1}{6\mathcal{A}_{ijk}} \sum_{pqr \in S^+_{ijk}} \mathcal{A}^{qr}_p (\alpha_{pq}\beta_{rp} - \beta_{rp}\alpha_{pq}), \qquad (18)$$



where S^+_{ijk} are the three even permutations of ijk, $\mathcal{A}_{ijk} \in \mathbb{R}$ is the area of triangle ijk, and $\mathcal{A}^{jk}_i \in \mathbb{R}$ are the (unsigned) *Voronoi areas* obtained by connecting the circumcenter of ijk to its edge midpoints (see inset). In 3D, the values $\Omega_{ijk} \in \mathfrak{so}(3)$ now describe both the speed *and* axis of rotation around singular curves (see Sec. 3.2.2).

There are two ways to derive Eqn. 17: either discretize a smooth structure equation (App. A.4), or expand the monodromy around triangle ijk (*i.e.*, $\exp(\omega_{ki})\exp(\omega_{jk})\exp(\omega_{ij})$) via the *Baker-Campbell-Hausdorff formula*. The first-order terms yield the discrete exterior derivative; the second-order terms yield the discrete wedge product. Since higher-order terms are omitted, values ω satisfying Eqn. 17 do not *exactly* characterize a discrete frame—rather than a hard constraint, we therefore use a penalty

$$R(\omega) := \sum_{ijk \in \mathcal{K}^*_2} |(d\omega)_{ijk} - (\omega \wedge \omega)_{ijk} + \Omega_{ijk}|^2. \qquad (19)$$

Here $\mathcal{K}^*_2 := \mathcal{K}_2 \setminus (\partial\mathcal{K}_2 \cup \mathcal{S}^B_2)$ denotes the set of triangles in $\mathcal{K}$ that are neither on the domain boundary, nor on the interior of singularity tubes—for these triangles, integrability of ω will be encoded by linear constraints in Secs. 3.2.1 and 3.2.2, *resp.* In practice this penalty yields values ω that are extremely close to integrable (even on very coarse meshes), as demonstrated in Sec. 4.2.

3.2 Constraints (3D)

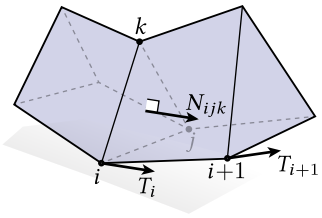
3.2.1 Boundary Adaptation. Along the domain boundary, the field must agree with singular curves at their endpoints; for hex meshing it should also be adapted to the surface normal. Suppose that at boundary vertices $i \in \partial\mathcal{K}_0$ we write the field as $E_i = \exp(\alpha_i \widehat{N}_i) E^0_i$, *i.e.*, as a rotation of the reference frame E^0 (from Sec. 2.4) by an angle α_i around the normal. Letting these angles be free parameters in the optimization, and letting $N_{ij} := \frac{1}{2}(N_i + N_j)$, the constraint

$$\omega_{ij} = \omega^0_{ij} + (\alpha_j - \alpha_i)\widehat{N}_{ij}, \qquad ij \in \partial\mathcal{K}_1 \qquad (20)$$

then allows the field to freely rotate around the normal (App. A.4), while ensuring the *total* rotation around closed loops—and in particular, around singular triangles—is fixed: consider summing $\alpha_j - \alpha_i$ around any loop (see also App. A.4.1). For most domains this constraint also ensures trivial monodromy around *all* noncontractible loops (not just those on the boundary); see App. A.2.1.

3.2.2 Curve Adaptation. For hexahedral meshing, the monodromy $\Omega_{ijk} \in \mathfrak{so}(3)$ around any singular curve must have magnitude $k\pi/2$ for some $k \in \mathbb{Z}$, and direction parallel to the curve tangent T. We must also apply linear constraints that ensure frames are *adapted* (*i.e.*, tangent) to the curve. Both conditions are essential: a fractional turn around an arbitrary axis does not define a consistent frame (Fig. 8); a field that merely makes *some* consistent rotation—but not around the curve tangent—is generally not meshable (Fig. 3, *left*).

Monodromy. Recall the notation from Sec. 1.1.1. For triangles $ijk \in \mathcal{S}^A_2$ we set the value of Ω_{ijk} to $2\pi\sigma_{ijk}\widehat{N}_{ijk}$, giving the unit normal N_{ijk} the same orientation as the tube. Since these triangles already contain all tube vertices, we omit the structure equation (Eqn. 17) from interior triangles $ijk \in \mathcal{S}^B_2$, which would be redundant. Finally, for the nonsingular triangles $ijk \in \partial\mathcal{S}$, we set $\Omega_{ijk} = 0$.



Adaptation. To adapt frames to singularity and feature curves, we include linear constraints akin to Eqn. 20 for each edge $ij \in \mathcal{S}_1$ running along a singularity tube. For each vertex $i \in \mathcal{S}_1$, let T_i denote the unit normal N_{ijk} of the associated triangle ijk (see inset) and let E^0_i be an arbitrary reference frame adapted to $\mathcal{S}_1$ at i (*e.g.*, the frame of least twist). For each edge $ij \in \mathcal{S}_1$, let $T_{ij} := \frac{1}{2}(T_i + T_j)$ and let ω^0_{ij} be the Darboux derivative of E^0 (Eqn. 7). We can then specify two different kinds of constraints—either

$$\omega_{ij} = \omega^0_{ij} + \alpha_{ij}\widehat{T}_{ij}, \qquad (21)$$

or

$$\omega_{ij} = \omega^0_{ij} + (\alpha_j - \alpha_i)\widehat{T}_{ij}. \qquad (22)$$

In both cases, the values ω^0_{ij} account for the bending of the curve, ensuring that the frame remains adapted as it moves from i to j. The $\widehat{T}_{ij}$ terms determine the frame's torsion along the curve: using free parameters $\alpha_{ij} \in \mathbb{R}$ per edge permits any torsion whatsoever, whereas taking differences of free parameters $\alpha_i \in \mathbb{R}$ per vertex forces the *total* torsion around closed loops to equal the total torsion of the reference frame E^0 (since the differences sum to zero).

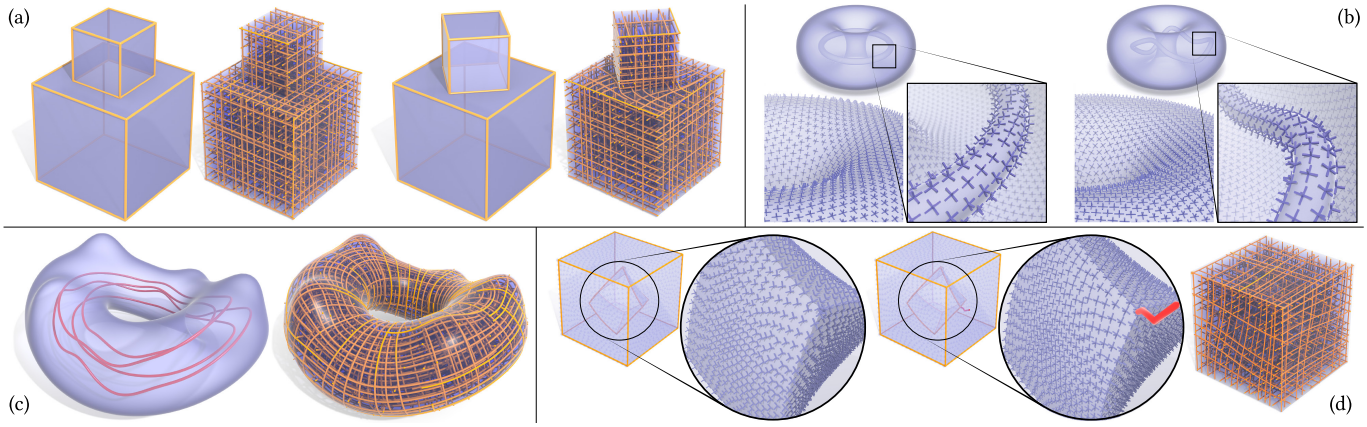
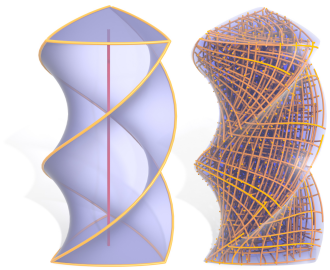


Fig. 12. Even when the constraint set has disconnected components, integrability of ω is typically sufficient to ensure that the frame is correctly adapted to boundary normals and curve tangents—even in the absence of symmetry. Here we show a domain with disconnected feature curves (a), disconnected boundary components (b), and singular loops that make no contact with the boundary (c). A rare exception is shown in (d), where the cross field on two nested cubes can be globally represented by an ordinary rotation field; here we can simply connect components by a feature curve (in red) to ensure proper alignment.



While Eqn. 22 provides explicit control when there are multiple solutions (say, a solid torus *without* boundary adaptation), it is typically easier to use Eqn. 21, since the torsional period need not be chosen *a priori*. Consider for example the twisted prism shown in the inset: to obtain a torsion compatible with the boundary normals one could either use Eqn. 22 and design an initial frame E^0 along the singular (red) curve that rotates by $4\pi/3$ around the vertical axis, or use Eqn. 21 and simply let the free parameters α_{ij} automatically determine the correct torsion (as done for the prism). Fig. 13 shows a similar example for closed loops.

Finally, for sharp feature curves along the boundary we simply set $\omega_{ij} = \omega^0_{ij}$ where ω^0 is the Darboux derivative of the cross field best adapted to the curve tangent and the boundary normals at each vertex (see edge *ab* in Fig. 11). Since crosses must remain adapted to the normals, a free torsion parameter is not needed.

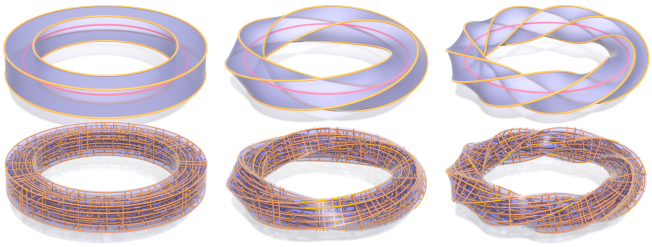


Fig. 13. We can allow the torsion of the frame along singular and feature curves to be free during optimization—and hence do not have to determine torsional periods *a priori*. Here for instance the frame automatically makes the correct number of twists as it travels around the red loop (from left to right: 0, 1, and 2), keeping it compatible with the boundary normals.

3.2.3 Disconnected Components. A special case to consider are domains where the constraint set is disconnected (as in Fig. 12). Since constraints on ω prescribe only the local *change* in the field—and not its absolute orientation—it is not immediately obvious that a field adapted to one boundary component will be adapted to all others. Crane et al. [2010, Section 2.8] describe a similar situation in 2D, where disconnected components of directional constraints are joined by paths with prescribed angle sums. The same strategy cannot be applied in 3D, due to the failure of Eqn. 3.

However, the situation turns out to be *easier* in 3D than in 2D: any integrable 1-form ω already describes a frame that is correctly adapted to all constraints. The basic reason is illustrated in Fig. 8: suppose a cross field E had Darboux derivative ω, but was *not* correctly adapted to the constraint set at some vertex $i \in \mathcal{K}_0$. Due to the constraints in Sections 3.2.1 and 3.2.2, the monodromy of ω around any loop γ based at i must be a cube symmetry *around the axes of the adapted frame*. In general, then, developing an incorrectly adapted frame E_i around such a loop would yield an inequivalent frame E'_i, *i.e.*, ω would not actually be the Darboux derivative of E—a contradiction. The only exception is when *all* loops based at all boundary points have monodromy equal to the identity, *i.e.*, when the solution can be globally expressed as an ordinary frame field rather than a cross field. (See also discussion in App. A.3.1.)

In short, as long as ω is integrable, special treatment of disconnected components is typically not needed. For example, Fig. 12c shows correct adaptation to both singular curves and boundary normals on an asymmetric torus with four disconnected singular loops of index +1/4. In contrast, Fig. 12d, *left* shows misalignment on an example where the solution can be expressed as an ordinary frame field. Here, connecting the two components by an index-0 feature curve with free torsion (*à la* Eqn. 21) restores proper alignment. In practice we often find that no additional constraints are needed even when the solution *can* be represented by an ordinary frame field—see for instance Fig. 12a and b. Further analysis of this behavior is an interesting question for future work.

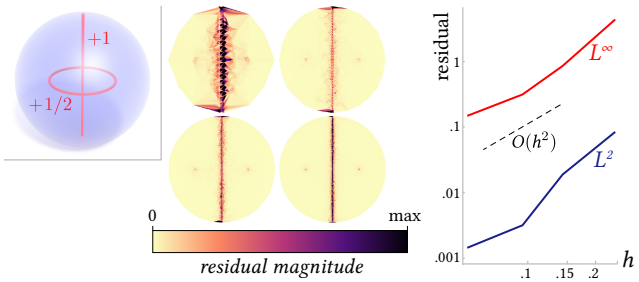


Fig. 14. Our discretization of Cartan's structure equation exhibits second order convergence with respect to mean edge length h, providing good numerical behavior even on coarse models.

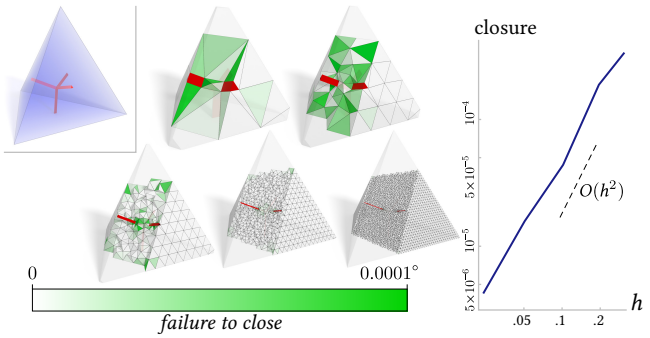


Fig. 15. Due purely to discretization error, rotations $\exp(\omega_{ij})$ exhibit an extremely small failure to close around triangles ijk, which vanishes rapidly under refinement. *Left:* cross section of the example shown in the upper-left. *Right:* convergence with respect to mean edge length h.

3.3 Optimization Problem (3D)

Our overall optimization problem is a nonlinear least squares problem subject to linear constraints:

$$\begin{aligned} \min_{\substack{\omega:\mathcal{K}_1\to\mathfrak{so}(3)\\ \alpha:\mathcal{B}\to\mathbb{R}}} \quad & ||\omega||^2 + aR(\omega) \\ \text{s.t.} \quad & \omega_{ij} = \omega^0_{ij} + \tfrac{1}{2}(\alpha_j - \alpha_i)\widehat{N}_{ij}, \quad \forall ij \in \partial\mathcal{K}_1 \\ & \omega_{ij} = \omega^0_{ij} + \tfrac{1}{2}\alpha_{ij}\hat{T}_{ij}, \quad \forall ij \in \mathcal{S}_1. \end{aligned} \tag{23}$$

Here, $\mathcal{B}$ is the set of vertices and edges where the adaptation constraints have real degrees of freedom α. The relative strength of the two objectives is controlled by the parameter $a > 0$, which affects only the rate of convergence (we use $a = 1000$ in all examples). In practice, we observe that this problem appears to produce globally optimal solutions, since any (empirically) initial guess leads to an identical minimizer—see Sec. 4.2 for further discussion.

Field Integration (3D). To recover the final field E, we propagate ω across the domain via breadth-first parallel transport exactly as in 2D (Sec. 2.4), except that the parallel transport matrices are now just $R_{ij} = \mathsf{I}$. Since ω determines E only up to a global rotation we start with an adapted frame on the domain boundary, though the particular choice of initial vertex $i \in \partial\mathcal{K}_0$ does not matter—see in particular Fig. 16.

4 EVALUATION AND RESULTS

4.1 Domain Generation

The volume mesh $\mathcal{K}$ is generated by specifying (i) the domain boundary, as an ordinary triangle mesh, and (ii) a collection of triangular singularity tubes, terminating at triangles on the domain boundary. The composite triangle mesh is then handed to any standard method for constrained Delaunay triangulation—we use *TetGen* [Si 2015] with default settings, and do not perform any subsequent processing to the mesh. Mesh sizes in our examples ranged from 22k to 222k tets, with an average size around 100k tets. We construct tubes by sweeping a triangle along a given collection of polylines; tubes meeting at an interior node are joined by a single tetrahedron. For complex or noisy singularity networks this simple sweeping procedure can be error prone, though is of course not fundamental to our approach—see Sec. 5 for further discussion.

4.2 Validation

Numerical experiments help validate our formulation. Fig. 14 plots the residual of the discrete structure equation (Eqn. 17) with respect to mesh refinement, indicating second-order convergence; as is standard for singular PDEs, we measure error on a fixed subdomain away from singular curves. In Fig. 15 we quantify the integrability of ω by measuring the magnitude of the monodromy in each face ijk (*à la* Eqn. 9), which is no more than a small fraction of a degree even on the coarsest mesh. Here again we observe the expected second order convergence, strongly suggesting that any lack of integrability is purely due to discretization error, rather than a failure of the solver to produce an optimal solution. Fig. 16 further confirms that our solution is almost perfectly integrable not only locally but also *globally*: here we propagate ω in breadth-first order either from the domain boundary (where the field is known), or from an arbitrary point on the interior; in each case, the global accumulation of error is small enough that the maximum change in any cross is no more than about 1°. We also check that the integrated frame E is closely adapted to the normal of the domain boundary and the tangents of the singularity curves: across all examples in the paper, the average error ranges from 0.014° to 1.72° with a standard deviation of 0.64°, even for the large index singularities in Fig. 9 and highly twisted boundary in Fig. 13. Overall, the discretization appears to be extremely accurate, even on coarse meshes.

Initialization. Fig. 17 shows the solutions obtained when initializing ω with random values, constant values, or the solution to an easier problem where we omit the quadratic term $\omega \wedge \omega$ from Eqn. 19 and can hence just solve a linear system. In each case the minimizer is *identical*, up to floating point error. This behavior is representative of our experience across a wide variety of examples: we always get the same solution, independent of initialization; we do not require a carefully-designed solver or optimization strategy. Though Eqn. 23 is not convex, such experiments strongly suggest that the solutions we obtain are globally optimal, much as eigenvalue problems are nonconvex, yet easily admit global minimizers. Further analysis of this problem is an interesting topic for future work.

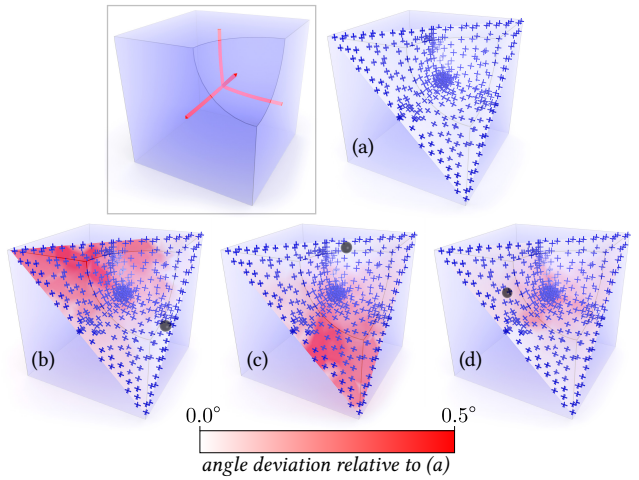


Fig. 16. Even on a fairly coarse mesh (4k vertices, pictured *top left*), local integrability error is small enough that we obtain a virtually identical cross field whether we integrate ω via a breadth-first search from the domain boundary (a), or from an arbitrary interior point (b, c and d).

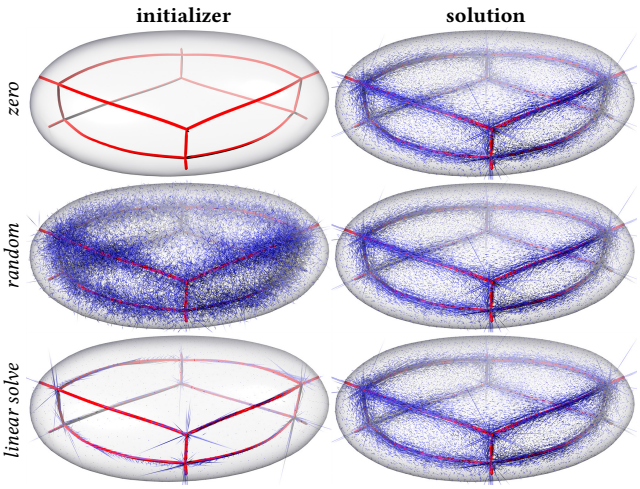


Fig. 17. Independent of initial guess *(left)*, our optimization problem yields an identical solution *(right)* up to floating point error. Here we plot ω as a vector per edge.

Local Smoothing. We also compared the raw output of our algorithm with the field obtained by performing additional local smoothing, using a simple iterative scheme akin to Gao et al. [2017]. At each iteration the frame E_i is replaced with the Karcher mean of its neighbors, *i.e.*, the minimizer of the energy $\sum_{ij} w_{ij} d(E_i, E_j)^2$, where $d(\cdot,\cdot)$ is the distance on SO(3), and $E_j \in \mathrm{SO}(3)$ is the representative of the cross at j closest to E_i. Even on coarse meshes, this procedure yields virtually no change to our solution (Fig. 18). In other words, our smoothness energy captures what one might naturally desire at the discrete level: it minimizes the difference in rotation between adjacent crosses. (Note that we do *not* use this smoothing procedure for any other examples.)

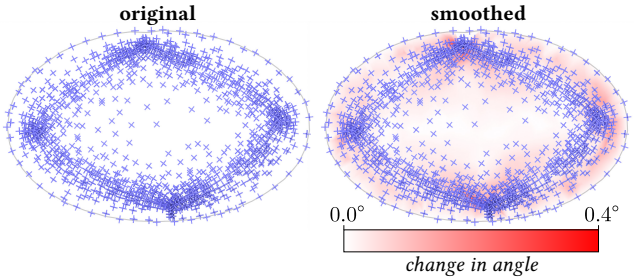


Fig. 18. Applying additional local smoothing makes an imperceptible change to our solution, indicating that it also does a good job of minimizing rotational differences at the discrete level. Here we visualize a cross section before and after smoothing, as well as the change in angle due to smoothing.

4.3 Performance

The main cost in our algorithm is solving the optimization problem for ω on the volume (Eqn. 23); here we used a standard Levenberg–Marquardt solver without line search [Moré 1978], though many efficient alternatives are available [DeVito et al. 2017]. Each iteration entails solving a roughly $|3E| \times |3E|$ positive definite linear system; we made no effort whatsoever to optimize our code, and simply use the `backslash` command in *MATLAB* (which performs sparse Cholesky factorization); Eqn. 15 was solved using `quadprog` in *MATLAB*, but can easily be reformulated as a sparse linear system (Sec. 2.3). With this implementation, setting up and solving our two optimization problems on a mesh with 130,000 edges takes a couple minutes on a 4GHz Intel Core i7 with 16GB of RAM. The number of iterations does not seem to depend strongly on mesh resolution: all of our examples take about 5–10 iterations to converge. Other steps did not contribute significantly to computational cost.

4.4 Comparisons

The only other method which generates a meshable field compatible with a given set of singular curves is the one of Liu et al. [2018]. Since both methods produced fields with the same global topology, we compared field smoothness, quantified using either (i) the quaternion Dirichlet energy optimized by Liu et al. [2018, Equation 23], or (ii) the ℓ_2 norm of angle differences between frames. More precisely, we sum over interior faces to get

$$\begin{aligned} \phi_{\mathbb{H}} &:= (\textstyle\sum_{ijk} w_{ijk} |q_{ijka} - q_{ijkb}|^2)^{1/2}, \quad \text{and} \\ \phi_{\theta} &:= (\textstyle\sum_{ijk} w_{ijk} \theta_{ijk}^2)^{1/2}. \end{aligned}$$

Here q_{ijka}, q_{ijkb} are quaternions expressing the frames in the two tets containing ijk, θ_{ijk} is the smallest angle between the same two frames, and the weight $w_{ijk} \in \mathbb{R}$ is triangle area divided by the dual edge length (*i.e.*, the diagonal Hodge star on dual 1-forms). To provide a fair comparison, we sample our fields onto the meshes used by Liu *et al.* On average we find that our fields exhibit about 20% and 32% lower energy with respect to $\phi_{\mathbb{H}}$ and ϕ_{θ}, *resp.* In other words, they are smoother even with respect to Liu *et al.*'s own measure of smoothness—which is not too surprising, given their use of an eigenvalue relaxation. In the context of meshing, the rotational smoothness ϕ_{θ} is likely a more natural measure of field quality, since frame magnitude plays no role.

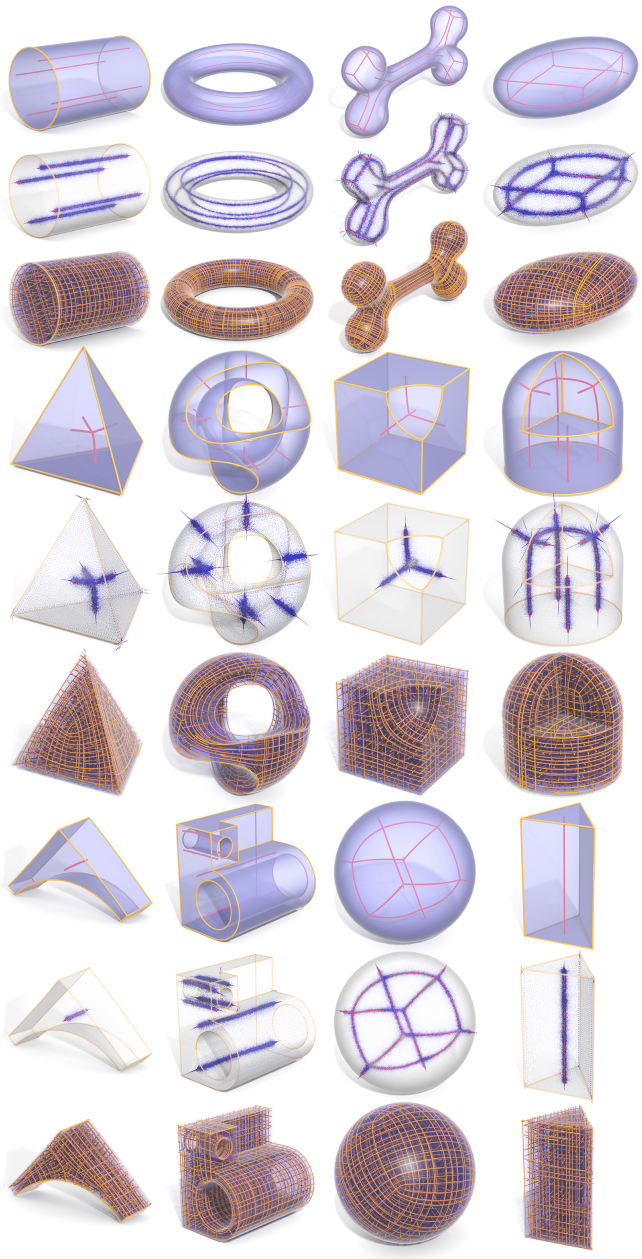


Fig. 19. Fields computed via our method; for each model we show the input network *(top)*, Darboux derivative ω *(middle)* and 3D cross field *(bottom)*.

In terms of performance, the bottleneck in our algorithm is a nonlinear least squares problem; for Liu *et al.*it is a principal eigenvalue problem. In 2D both problems are efficiently solved via a small sequence of sparse linear systems using a fixed (symbolic or numeric) factorization, but in 3D sparse direct solvers generally exhibit poor scaling and hence neither method can benefit from the amortized gains of prefactorization. Iterative solvers for least squares [DeVito et al. 2017] or eigenvalue problems provide an attractive alternative,

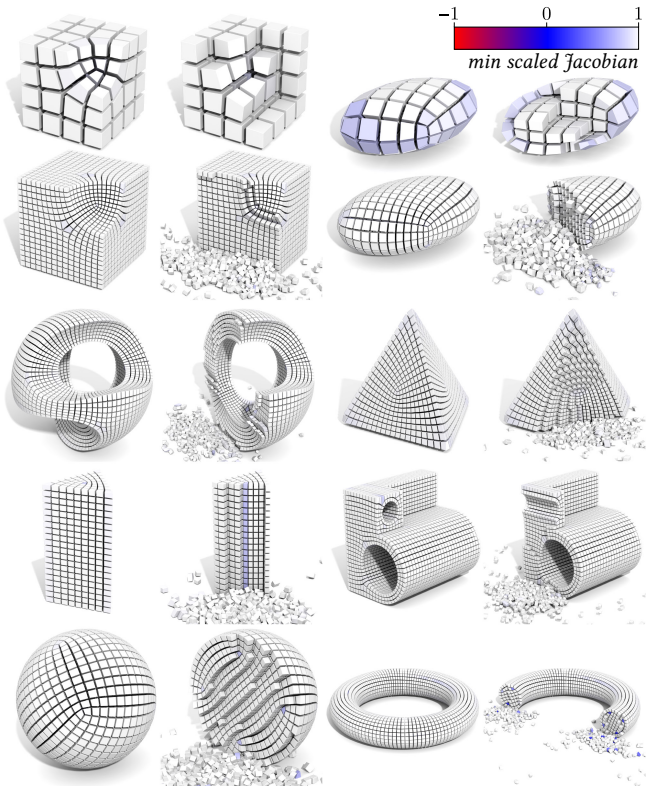


Fig. 20. Hexahedral meshes generated from our fields; for each mesh we show a "fallaway" view to visualize interior element quality. Even coarse meshes (top row) respect the given singularity structure, and generally exhibit good element quality.

though a real-world performance comparison is far from clear given the broad range of options. Other aspects of computation (such as our 2D problem, or the merging & zippering in Liu *et al.*) seem not to contribute significantly to practical runtime. Storage cost is also similar: we store three real values per edge (encoding an element of $\mathfrak{so}(3)$); Liu *et al.* store four real values per tet (encoding a quaternion); in practice the ratio of edges to tets in a Delaunay mesh is roughly 6:5, making the overall ratio of DOFs very close to 1:1.

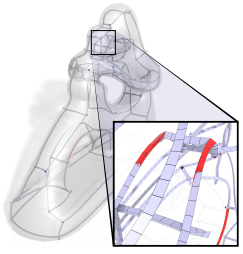
4.5 Examples

Fields. Examples of fields computed via our method are shown in Figures 1, 6, 9, 12, 13 and 19. In each case the input to the algorithm was a description of the domain boundary (blue), a valid network of singularity curves (red), and curves along sharp features (yellow); input data comes from Liu et al. [2018]. The Darboux derivative ω is plotted by drawing vectors that show the axis and angle of rotation (dark blue). The length of these vectors indicates the rotational smoothness of the field, verifying that non-smoothness occurs only near singularities and sharp corners, and falls off rapidly everywhere else. To visualize the cross field E obtained from ω, we trace integral curves through an interpolated field given by the barycentric weighted Karcher mean on $SO(3)$.

Meshing. Though our aim in this paper is not to build a full end-to-end meshing pipeline, we performed several preliminary experiments. In particular, we performed field-aligned parameterization via *CubeCover* [Nieser et al. 2011] and extracted hexahedral meshes using *HexEx* [Lyon et al. 2016]. To get frames on tetrahedra (needed by CubeCover) we computed the Karcher mean of frames at vertices; we also inserted the barycenter of each singular face $ijk \in \mathcal{S}_2^A$ (updating our mesh via *TetGen*) and omitted these vertices when taking averages. No additional processing was used; likewise, we made no modifications to the meshing algorithms, apart from using *CoMISo* in CubeCover [Bommes et al. 2012]. Matchings for CubeCover were obtained by finding the closest rotation, but in principle we should be able to make this step even more robust near singularities by using angle information from ω. Several examples are shown in Fig. 20, where we plot the *minimum scaled Jacobian* for each cell; here 1 is ideal and negative values indicate inversion (see [Vyas and Shimada 2009, Section 8.1] for a definition). To visualize element quality on the domain interior, we also provide a "fallaway" view where we run a rigid body simulation on elements removed by a cutaway plane. We applied no post-processing, and generally obtained high-quality elements with no inversions; in all cases the input singularity structure was preserved exactly.

5 LIMITATIONS AND FUTURE WORK

The main limitation of our method is that the user is required to specify a valid singularity network—an enticing question is how moving frames may help with automatic generation of such networks. Here our PDE-constrained optimization problem may fit nicely with recent techniques for computing optimal singularities via *measure relaxation* [Soliman et al. 2018]. There is currently no clear reason why our nonlinear least squares problem should always yield a globally optimal (or even integrable) solution, as it appears to do in practice (Fig. 17); a deeper understanding of this phenomenon may prove valuable. Pure rotation fields with disconnected boundary components may be misaligned (Sec. 3.2.3), but this issue is largely addressed via extra feature curves.



A practical nuisance is building nicely-shaped tube geometry—on complicated examples (as shown in the inset), our naïve extrusion code often generates self-intersections (red) which cause *TetGen* to fail. This limitation is of course not fundamental to our formulation, and might be easily addressed using more flexible node geometry (*e.g.*, octahedra rather than just tetrahedra) which would also allow higher-degree nodes. Alternatively, it may be useful to consider a numerical treatment that does not depend on a special mesh structure, such as finite element or boundary element methods [Arnold et al. 2006; Solomon et al. 2017]. Finally, the machinery of moving frames is not tied in any way to the rotation group $SO(3)$, or to symmetries of the cube (see App. A). Hence, much of our algorithm can be directly applied to other Lie groups G and/or other symmetry groups, which may facilitate more general field-guided *anisotropic* meshing problems (*e.g.*, for boundaries with sharp dihedral angles), as recently explored in 2D [Diamanti et al. 2014; Jiang et al. 2015].

ACKNOWLEDGMENTS

Thanks to Heng Liu and David Bommes for help with parameterization and mesh extraction, and to the anonymous reviewers for useful feedback. This work was supported by a Packard Fellowship, NSF Award 1717320, and gifts from Autodesk, Adobe, and Facebook.

REFERENCES

R. Abraham, J. E. Marsden, and R. Ratiu. 1988. *Manifolds, Tensor Analysis, and Applications: 2nd Edition.* Springer-Verlag, Berlin, Heidelberg.
C. Armstrong, H. Fogg, C. Tierney, and T. Robinson. 2015. Common Themes in Multi-block Structured Quad/Hex Mesh Generation. *Proced. Eng.* 124 (2015).
D. Arnold, R. Falk, and R. Winther. 2006. Finite element exterior calculus, homological techniques, and applications. *Acta Numer.* 15 (2006), 1–155.
M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, and E. Grinspun. 2008. Discrete Elastic Rods. *ACM Trans. Graph.* 27, 3 (Aug. 2008), 63:1–63:12.
David Bommes, Henrik Zimmer, and Leif Kobbelt. 2009. Mixed-integer Quadrangulation. *ACM Trans. Graph.* 28, 3, Article 77 (July 2009), 10 pages.
D. Bommes, H. Zimmer, and L. Kobbelt. 2012. Practical Mixed-Integer Optimization for Geometry Processing. In *Curves and Surfaces.* 193–206.
M. Brin, K. Johannson, and P. Scott. 1985. Totally peripheral 3-manifolds. *Pacific J. Math.* 118, 1 (1985), 37–51.
K. Crane, F. de Goes, M. Desbrun, and P. Schröder. 2013. Digital Geometry Processing with Discrete Exterior Calculus. In *ACM SIGGRAPH 2013 courses (SIGGRAPH '13).*
K. Crane, M. Desbrun, and P. Schröder. 2010. Trivial Connections on Discrete Surfaces. *Comp. Graph. Forum (SGP)* 29, 5 (2010), 1525–1533.
F. de Goes, M. Desbrun, and Y. Tong. 2016. Vector Field Processing on Triangle Meshes. In *ACM SIGGRAPH 2016 Courses (SIGGRAPH '16).* 27:1–27:49.
M. Desbrun, E. Kanso, and Y. Tong. 2006. Discrete Differential Forms for Computational Modeling. In *ACM SIGGRAPH 2006 Courses (SIGGRAPH '06).* 16.
Z. DeVito, M. Mara, M. Zollöfer, G. Bernstein, C. Theobalt, P. Hanrahan, M. Fisher, and M. Nießner. 2017. Opt: A Domain Specific Language for Non-linear Least Squares Optimization in Graphics and Imaging. *ACM Trans. Graph.* (2017).
T. Dey, F. Fan, and Y. Wang. 2013. An Efficient Computation of Handle and Tunnel Loops via Reeb Graphs. *ACM Trans. Graph.* 32, 4 (2013).
O. Diamanti, A. Vaxman, D. Panozzo, and O. Sorkine. 2014. Designing N-PolyVector Fields with Complex Polynomials. *Proc. Symp. Geom. Proc.* 33, 5 (Aug. 2014).
M.P. do Carmo. 1994. *Differential Forms and Applications.* Springer-Verlag.
J. Frauendiener. 2006. Discrete differential forms in general relativity. *Classical and Quantum Gravity* 23, 16 (2006).
X. Gao, W. Jakob, M. Tarini, and D. Panozzo. 2017. Robust Hex-dominant Mesh Generation Using Polyhedral Agglomeration. *ACM Trans. Graph.* 36, 4 (2017).
A. Hertzmann and D. Zorin. 2000. Illustrating Smooth Surfaces. In *Proc. SIGGRAPH.*
A. Hirani. 2003. *Discrete Exterior Calculus.* Ph.D. Dissertation. Caltech.
J. Huang, Y. Tong, H. Wei, and H. Bao. 2011. Boundary Aligned Smooth 3D Cross-frame Field. *ACM Trans. Graph.* 30, 6, Article 143 (Dec. 2011).
T. Jiang, X. Fang, J. Huang, H. Bao, Y. Tong, and M. Desbrun. 2015. Frame Field Generation Through Metric Customization. *ACM Trans. Graph.* 34, 4 (July 2015).
Junho Kim, M. Jin, Q. Zhou, F. Luo, and X. Gu. 2008. Computing Fundamental Group of General 3-Manifold. In *Advances in Visual Computing.*
Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2013. Globally optimal direction fields. *ACM Trans. Graph.* 32, 4 (2013).
J. Lee. 2003. *Introduction to Smooth Manifolds.* Springer.
Y. Li, Y. Liu, W. Xu, W. Wang, and B. Guo. 2012. All-hex Meshing Using Singularity-restricted Field. *ACM Trans. Graph.* 31, 6 (Nov. 2012), 177:1–177:11.
Y. Lipman, D. Cohen-Or, R. Gal, and D. Levin. 2007. Volume and Shape Preservation via Moving Frame Manipulation. *ACM Trans. Graph.* 26, 1 (Jan. 2007).
Y. Lipman, O. Sorkine, D. Levin, and D. Cohen-Or. 2005. Linear Rotation-invariant Coordinates for Meshes. *ACM Trans. Graph.* 24, 3 (July 2005).
H. Liu, P. Zhang, E. Chien, J. Solomon, and D. Bommes. 2018. Singularity-constrained Octahedral Fields for Hexahedral Meshing. *ACM Trans. Graph.* 37, 4 (2018).
M. Lyon, D. Bommes, and L. Kobbelt. 2016. HexEx: Robust Hexahedral Mesh Extraction. *ACM Trans. Graph.* 35, 4 (July 2016), 11.
E. Mansfield, G. Marí Beffa, and J.P. Wang. 2013. Discrete moving frames and discrete integrable systems. *Found. Comput. Math.* 13, 4 (2013).
J. Moré. 1978. The Levenberg-Marquardt Algorithm: Implementation and Theory. In *Numerical Analysis*, G.A. Watson (Ed.). Lecture Notes in Mathematics, Vol. 630.
M. Nieser, U. Reitebuch, and K. Polthier. 2011. CubeCover: Parameterization of 3D Volumes. *Computer Graphics Forum* 30, 5 (2011).
P. Olver. 2000. Moving Frames in Geometry, Algebra, Computer Vision, and Numerical Analysis. In *Foundations of Computational Mathematics.*
J. Palacios, L. Roy, P. Kumar, C.Y. Hsu, W. Chen, C. Ma, L.Y. Wei, and E. Zhang. 2017. Tensor Field Design in Volumes. *ACM Trans. Graph.* 36, 6 (Nov. 2017).
J. Palacios and E. Zhang. 2007. Rotational Symmetry Field Design on Surfaces. *ACM Trans. Graph.* 26, 3 (July 2007).

H. Pan, Y. Liu, A. Sheffer, N. Vining, C.J. Li, and W. Wang. 2015. Flow Aligned Surfacing of Curve Networks. *ACM Trans. Graph.* 34, 4 (July 2015).
N. Ray, D. Sokolov, and B. Lévy. 2016. Practical 3D Frame Field Generation. *ACM Trans. Graph.* 35, 6 (Nov. 2016).
R.W. Sharpe. 2000. *Differential Geometry: Cartan's Generalization of Klein's Erlangen Program.* Springer New York.
H. Si. 2015. TetGen, a Delaunay-Based Quality Tetrahedral Mesh Generator. *ACM Trans. Math. Softw.* 41, 2 (Feb. 2015).
Y. Soliman, D. Slepčev, and K. Crane. 2018. Optimal Cone Singularities for Conformal Flattening. *ACM Trans. Graph.* 37, 4 (2018).
J. Solomon, A. Vaxman, and D. Bommes. 2017. Boundary Element Octahedral Fields in Volumes. *ACM Trans. Graph.* 36, 4, Article 114b (May 2017).
Vaxman, Campen, Diamanti, Panozzo, Bommes, Hildebrandt, and Ben-Chen. 2016. Directional Field Synthesis, Design and Processing. *Comp. Graph. Forum* (2016).
V. Vyas and K. Shimada. 2009. Tensor-Guided Hex-Dominant Mesh Generation with Targeted All-Hex Regions. In *Proc. Int. Mesh. Roundtable*, Brett W. Clark (Ed.).
F.W. Warner. 2013. *Foundations of Differentiable Manifolds and Lie Groups.*
W. Yu, K. Zhang, and X. Li. 2015. Recent algorithms on automatic hexahedral mesh generation. In *Int. Conf. Comp. Sci. Ed.* 697–702.

A SMOOTH FORMULATION

Our formulation is based on Cartan's *method of moving frames*—the basic idea is to express the derivatives of a frame field *with respect to the field itself*, akin to using body-centered angular velocities. Just as the fundamental theorem of calculus asserts that an ordinary function is determined by its derivative (up to a constant shift), an analogous theorem tells us that a frame field can be recovered from its *Darboux derivative*, up to a global rotation (Thm. A.2). In this section we provide essential background on moving frames, and show how they can be extended to symmetric 3D cross fields.

Traditionally, moving frames are introduced using orthonormal coordinate frames on $\mathbb{R}^n$ [do Carmo 1994]; a more modern approach is to consider a *principal bundle*, where orthonormal frames are replaced by elements of some *Lie group* G [Sharpe 2000]. This perspective helps make sense of 3D cross fields, since the space of crosses can be described as the quotient of the rotation group $G = \mathrm{SO}(3)$ by the cube symmetries Γ. Although this space is no longer a group, it is still a manifold on which the Darboux derivative *locally* satisfies the usual structure equation. Globally, the only difference is that monodromies are no longer trivial, but instead look like symmetries of the cube. An interesting consequence is that, in most cases, an integrable Darboux derivative now *uniquely* determines a cross field, *i.e.*, there is no longer a choice of global rotation (App. A.3.1).

We begin with a brief review of Lie groups, followed by a discussion of moving frames and their connection to 3D cross fields. Throughout we make use of *differential forms*—see Crane et al. [2013] for a pedagogical introduction, and do Carmo [1994] or Abraham et al. [1988] for a more detailed reference.

A.1 Lie Groups

Lie groups and *Lie algebras* provide a unified picture of spatial transformations and their derivatives (*resp.*). The basic idea is that, since transformations can vary continuously, they can be viewed as points on a *smooth manifold*; since they can be composed in a natural way, they also have the structure of a *group*. For concreteness we consider the special case of rotations around the origin in $\mathbb{R}^n$, represented by $n \times n$ orthogonal matrices with positive determinant (Sec. 0.3). This example captures the most essential features of the general case and will be needed to describe cross fields; the cartoon in Fig. 21 helps provide intuition for the discussion below.

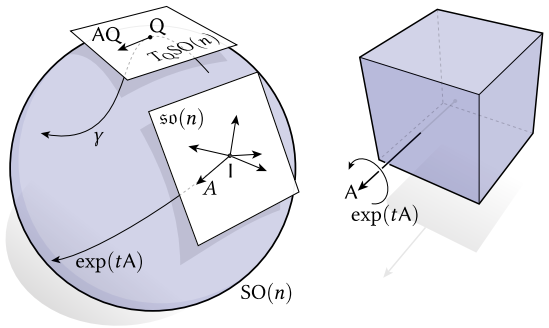
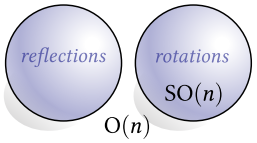


Fig. 21. Rotations of $\mathbb{R}^n$ can be viewed as a smooth manifold $\mathrm{SO}(n)$, where a curve γ describes a continuous family of rotations, and its tangents hence encode angular velocities. For example, the *exponential map* $\exp(t\mathrm{A})$ describes rotation at a constant velocity A for time t *(right)*, starting at the identity I. The *Lie algebra* $\mathfrak{so}(n)$ is the set of velocities A at the identity; any velocity at a point $\mathrm{Q} \in \mathrm{SO}(n)$ can be expressed as AQ for some $A \in \mathfrak{so}(n)$.

Group Structure. Rotations exhibit several natural properties: the composition of two rotations $\mathrm{Q}_1, \mathrm{Q}_2$ is another rotation $\mathrm{Q}_2 \circ \mathrm{Q}_1$; there is an *identity* rotation I that does nothing; every rotation Q can be undone by some *inverse* Q^{-1}; and different groupings of rotations have the same effect, *i.e.*, $(\mathrm{Q}_1 \circ \mathrm{Q}_2) \circ \mathrm{Q}_3 = \mathrm{Q}_1 \circ (\mathrm{Q}_2 \circ \mathrm{Q}_3)$. In general, any collection of objects with this behavior is called a *group*. Since rotations are represented by orthogonal matrices, the collection of all rotations is called the *special orthogonal group* $\mathrm{SO}(n)$, where *special* refers to the fact that rotations also preserve orientation ($\det(\mathrm{Q}) > 0$).



Manifold Structure. Much as a smooth surface can be expressed as the zero level set of a smooth function $f : \mathbb{R}^n \to \mathbb{R}$, we can view the group $\mathrm{O}(n)$ of orthogonal matrices as the zero set of the function $f(\mathrm{Q}) = \mathrm{Q}^\mathsf{T}\mathrm{Q} - \mathrm{I}$ taking matrices to symmetric matrices. This set has two components: one with positive determinant, corresponding to the rotation group $\mathrm{SO}(n)$, and another with negative determinant, corresponding to reflections (which do not form a group). This perspective allows us to think of rotations as a continuous space where nearby points represent similar rotations. Formally, since the zero matrix is a regular value of f, $\mathrm{SO}(n)$ is a smooth manifold of dimension $n(n-1)/2$—see [Warner 2013, Example 1.40].

Lie Algebra. The identity rotation I can be thought of as a special point on $\mathrm{SO}(n)$. Infinitesimal rotations of $\mathbb{R}^n$ are then described by vectors A in the tangent space $T_\mathrm{I}\mathrm{SO}(n)$, also known as the *Lie algebra* $\mathfrak{so}(n)$. Each Lie algebra element is represented by a *skew-symmetric* matrix $\mathrm{A}^\mathsf{T} = -\mathrm{A}$. To see why, consider a time-varying rotation $\mathrm{Q}(t)$ starting at $\mathrm{Q}(0) = \mathrm{I}$. Differentiating the relationship $\mathrm{Q}^\mathsf{T}(t)\mathrm{Q}(t) = \mathrm{I}$ at $t = 0$ yields $\frac{d}{dt}\mathrm{Q}^\mathsf{T}(0) = -\frac{d}{dt}\mathrm{Q}(0)$, *i.e.*, any infinitesimal rotation of the identity has the form $\mathrm{A}^\mathsf{T} = -\mathrm{A}$, as discussed in Sec. 0.3. Infinitesimal changes to any other rotation Q can then be expressed as AQ for some $\mathrm{A} \in \mathfrak{so}(n)$. The *Lie bracket* $[\mathrm{A}_1, \mathrm{A}_2] := \mathrm{A}_1\mathrm{A}_2 - \mathrm{A}_2\mathrm{A}_1$ on $\mathfrak{so}(n)$ captures the failure of small rotations to commute.

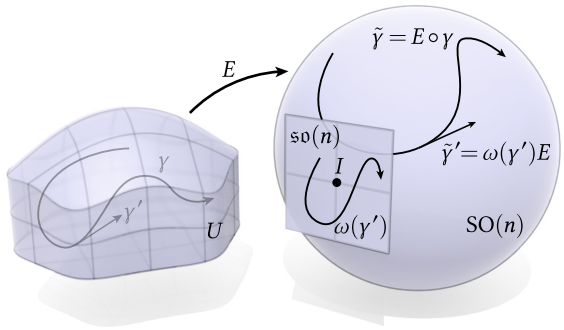


Fig. 22. A frame field on a region $U \subset \mathbb{R}^n$ can be viewed as a map E to the space $\mathrm{SO}(n)$ of rotations. The Darboux derivative $\omega(\gamma')$ expresses the change in the field as one walks along a curve γ, relative to E itself.

Exponential Map. Given a unit tangent vector A at the identity, the *exponential map* $\exp(t\mathsf{A})$ gives the point obtained by walking along the Lie group for a time t in the direction A along a straightest path or *geodesic*. In $\mathrm{SO}(n)$, $\exp(t\mathsf{A})$ is the rotation obtained by starting at the identity and integrating the angular velocity A for time t. In 2D for instance, where a skew-symmetric matrix A is determined by a single number $\theta \in \mathbb{R}$, $\exp(\mathsf{A})$ is just the corresponding rotation matrix given in Eqn. 1. Hence, when using angles to represent 2D frames we are working in the Lie *algebra*; when working with 2×2 rotation matrices we are working in the Lie *group*.

A.2 Moving Frames

How can we express the change in a spatially-varying frame field? Consider a solid region $U \subset \mathbb{R}^3$ bounded by a smooth surface ∂U. A *moving frame* on U is a smoothly-varying orthonormal frame $E : U \to \mathrm{SO}(3) \subset \mathbb{R}^{3\times 3}$ taking each point $p \in U$ to a rotation $E(p)$ (Fig. 22). The first-order change in E at a point $p \in U$ is described by the *differential* dE, which maps any vector $X \in \mathbb{R}^3$ to the directional derivative along X:

$$dE_p(X) := \lim_{\varepsilon \to 0} \frac{E(p + \varepsilon X) - E(p)}{\varepsilon}.$$

The key idea behind moving frames is to express this change *with respect to the frame itself*, via the *Darboux derivative*

$$\omega_p(X) := (dE_p(X))E_p^{-1}. \tag{24}$$

The transformation E_p^{-1} takes us from the global coordinate frame to a local, moving frame that depends on the point p. In terms of the Lie group SO(3), it takes a vector tangent to the point $E_p \in \mathrm{SO}(3)$, and maps it to a tangent at the identity I, *i.e.*, to an element of the Lie algebra $\mathfrak{so}(3)$. The Darboux derivative is therefore an $\mathfrak{so}(3)$*-valued 1-form*, *i.e.*, a linear map $\omega : TU \to \mathfrak{so}(3)$ from tangent vectors to Lie algebra elements.

A.2.1 Integrability. Given a 1-form ω, can we construct a corresponding frame E? For any initial value $E_0 \in \mathrm{SO}(3)$, we can at least integrate ω along a simple path $\gamma : [0, L] \to U$ to get a *development* $\tilde{\gamma} : [0, L] \to \mathrm{SO}(3)$ whose Darboux derivative agrees with ω:

$$\omega \circ d\gamma = (d\tilde{\gamma})\tilde{\gamma}^{-1}.$$

In general, however, a *closed* curve $\gamma(0) = \gamma(L)$ may not have a development, since the frames at the two endpoints may not agree. This failure to close is called the *monodromy* of ω around γ:

$$\Phi_\omega(\gamma) := \tilde{\gamma}(L)\tilde{\gamma}(0)^{-1}.$$

For ω to consistently describe a frame over all of U, it must therefore have trivial monodromy $\Phi_\omega(\gamma) = \mathsf{I}$ around all closed loops γ. Equivalently, ω must satisfy a *structure equation* that accounts for monodromy around small, contractible loops; it must also exhibit trivial monodromy around a collection of large, noncontractible loops that generate the fundamental group $\pi_1(U)$.

Structure Equation. Viewing E as a map into $\mathrm{SO}(3) \subset \mathbb{R}^{3\times 3}$, and ω as a matrix-valued 1-form, we can write Eqn. 24 as

$$dE = \omega E. \tag{25}$$

Taking the exterior derivative yields

$$0 = d(dE) = (d\omega)E - \omega \wedge dE = (d\omega)E - \omega \wedge \omega E, \tag{26}$$

where in the final step we apply Eqn. 25. Since E is invertible at each point, Eqn. 26 is equivalent to *Cartan's 2nd structure equation*

$$d\omega = \omega \wedge \omega. \tag{27}$$

For $\mathfrak{so}(3)$-valued 1-forms α, β, the wedge product is given by

$$(\alpha \wedge \beta)(X, Y) = \tfrac{1}{2}\left([\alpha(X), \beta(Y)] - [\alpha(Y), \beta(X)]\right),$$

where $[\cdot, \cdot]$ is the Lie bracket (App. A.1). The structure equation provides a local integrability condition [Sharpe 2000, Thm. 6.1]:

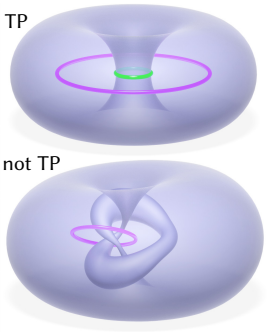
Theorem A.1. *On any simply-connected region $B \subseteq U$, an $\mathfrak{so}(3)$-valued 1-form ω satisfying Eqn. 27 is the Darboux derivative of some moving frame $E : B \to \mathrm{SO}(3)$.*

Nonsimply-Connected Domains. For domains with nontrivial topology (*e.g.*, a solid torus), we must also ensure that ω encodes a well-defined frame around noncontractible loops. If ω already satisfies Eqn. 27, then two homotopic loops γ_1, γ_2 starting and ending at the same basepoint $b \in U$ will have the same monodromy [Sharpe 2000, Thm. 7.7]; picking a different basepoint merely conjugates the monodromy by a fixed element of SO(3) [Sharpe 2000, Thm. 7.11]. It is therefore enough to ensure that ω has trivial monodromy around a representative loop γ from each class in the fundamental group $\pi_1(U, b)$ based at any point $b \in U$ [Sharpe 2000, Thm. 7.14]:

Theorem A.2 (Fundamental theorem of nonabelian calculus). *Let ω be an $\mathfrak{so}(3)$-valued 1-form on a path connected domain $U \subset \mathbb{R}^3$. Then ω is the Darboux derivative of a moving frame $E : U \to \mathrm{SO}(3)$ if and only if*

(i) it satisfies the structure equation $d\omega = \omega \wedge \omega$, and
(ii) it has trivial monodromy around some representative loop γ in each class of $\pi_1(U)$.

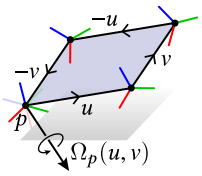
Moreover, for any two loops $\gamma_1, \gamma_2 \in \pi_1(U, b)$, the monodromy of the concatenated loop $\gamma_1 + \gamma_2$ is just the product $\Phi(\gamma_1)\Phi(\gamma_2)$ [Sharpe 2000, Prp. 7.10]. Hence, it is sufficient to have trivial monodromy around a collection of generators for $\pi_1(U)$.



In fact, suppose that every closed loop in U is freely homotopic to some loop on the boundary ∂U, *i.e.*, that U is *totally peripheral (TP)* [Brin et al. 1985]. Then it is enough to have trivial monodromy around all loops on the boundary, which we ensure by asking the monodromy of ω to agree with the Darboux derivative ω^0 of some fixed boundary frame E^0 (Sec. 3.2.1). This strategy works on most domains—for instance, the solid torus has a single generator homotopic to a loop on the boundary (inset, top). In the rare case where U is not totally peripheral, such as the complement of a trefoil knot (inset, bottom) one could explicitly compute the generators [Kim et al. 2008] and include them as feature curves (*à la* Sec. 3.2.2), though this strategy was not needed in our examples.

A.2.2 Curvature and Singularities. The failure of a 1-form ω to be integrable is captured by the *curvature 2-form*

$$\Omega := d\omega - \omega \wedge \omega.$$



Geometrically, $\Omega(u,v)$ describes the limit monodromy around an infinitesimal parallelogram with edges u, v (Ambrose-Singer). If $\Omega = 0$, then ω is at least locally integrable.

For a 2-manifold M, a key observation from Crane et al. [2010] is that ω is still integrable almost everywhere even if Ω is nonzero on a collection of isolated singular points $p_1, \ldots, p_n \in M$. More specifically, suppose

$$\Omega = \sum_{i=1}^{n} 2\pi\sigma_i \delta_{p_i},$$

where δ_p is a Dirac delta at p, and $\sigma_i \in \mathbb{Z}$ is the index of the singularity at p_i. Then ω encodes a well-defined frame field on $M \setminus \{p_1, \ldots, p_n\}$, where it still describes whole rotations around closed loops (Fig. 4).

Likewise, in 3D, we can encode a network of singular curves $\gamma_1(s), \ldots, \gamma_n(s)$ with prescribed indices $\sigma_1, \ldots, \sigma_n$ (*resp.*) by letting Ω be a distribution supported on these curves. In particular, let

$$\Omega(s) = \sum_{i=1}^{n} \sigma_i \mathcal{H}^1_{\gamma_i} \widehat{T}_i(s),$$

where $T_i(s)$ is the tangent to γ_i at s, and $\mathcal{H}^1_\gamma$ is the Hausdorff measure associated with γ (*i.e.*, $\mathcal{H}^1_\gamma(B) := \int_{\gamma \cap B} ds$ for any subset $B \subset U$). A 1-form ω satisfying the augmented structure equation

$$d\omega = \omega \wedge \omega + \Omega \tag{28}$$

then exhibits the prescribed number of rotations for small loops around γ, and some whole number of rotations around all loops in $U \setminus (\gamma_1 \cup \cdots \cup \gamma_n)$. If we also want the field to be adapted to a singular curve γ (*e.g.*, to ensure the field is locally meshable, as illustrated in Fig. 3), we can require that $\omega = \omega^0 + \alpha\widehat{T}$, where ω^0 is the Darboux derivative of some fixed cross field on γ and the 1-form $\alpha : T\gamma \to \mathbb{R}$ parameterizes the torsion along γ (see Sec. 3.2.2 for further discussion).

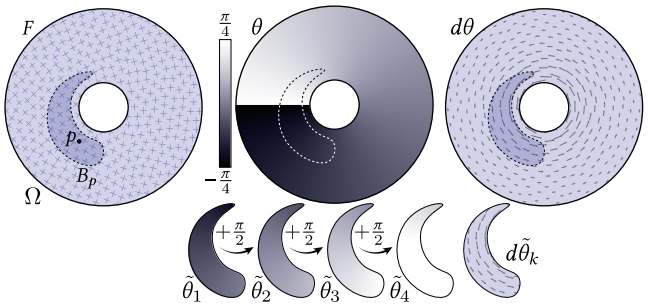


Fig. 23. Even if a cross field F cannot be represented by any globally continuous function θ, there are always several *local trivializations* $\tilde{\theta}$ in a simply connected neighborhood B_p around each point p, which differ only by quarter rotations. Since the derivatives $d\tilde{\theta}_k$ are all the same, they can be used to define a globally continuous derivative $d\theta$.

A.3 Symmetric Moving Frames

To represent 3D cross fields, we depart from the ordinary theory of moving frames and replace SO(3) with its quotient by cube symmetries. Since SO(3) has no normal subgroups, this quotient cannot be a (Lie) group; nonetheless, we can still take a quotient in the topological sense to obtain a smooth manifold where each point specifies a unique cross. Even in the absence of group structure, the manifold structure remains sufficient to express integrability conditions on ω.

More precisely, let $\Gamma \subset \mathrm{SO}(3)$ denote the rotational symmetries of the standard cube (sometimes called the *rotational octahedral group*), and let $\mathcal{C} := \mathrm{SO}(3)/\Gamma$ denote the quotient of the manifold SO(3) by the right action of Γ, *i.e.*, two rotations $E_1, E_2 \in \mathrm{SO}(3)$ are considered equivalent if $E_2 = E_1 g$ for some $g \in \Gamma$. It is then a standard result that $\mathcal{C}$ is a smooth manifold, with a smooth covering map $P : \mathrm{SO}(3) \to \mathcal{C}$ [Lee 2003, Proposition 9.26].

A (Γ-)*symmetric moving frame* on U is then a map $F : U \to \mathcal{C}$. Although F is not a section of a principal bundle (since $\mathcal{C}$ is not a Lie group), we can still define a Darboux derivative globally. A good analogy is a 2D cross field on a region $\Omega \subset \mathbb{R}^2$ expressed as a function $\theta : \Omega \to \mathbb{R}$ giving the angle of one of the four cross directions (Fig. 23). Though we cannot always find a θ that is globally differentiable, we can find a *local trivialization* $\tilde{\theta} : B_p \to \mathbb{R}$ that is differentiable in a neighborhood B_p around any given point $p \in \Omega$. Moreover, the derivative of this function does not depend on *which* function $\tilde{\theta}$ we pick, since they all differ by constant shifts $c \in \frac{\pi}{2}\mathbb{Z}$.

Likewise, in a simply-connected neighborhood $B_p \subset U$ around each nonsingular point $p \in U$, a 3D cross field F can be represented by some ordinary frame field, *i.e.*, a map $\tilde{F} : B_p \to \mathrm{SO}(3)$ such that $P \circ \tilde{F} = F$. All such maps have the same Darboux derivative: if ω satisfies $d\tilde{F} = \omega\tilde{F}$, then it also satisfies $d(\tilde{F}g) = \omega(\tilde{F}g)$ for any $g \in \Gamma$, since $d(\tilde{F}g) = (d\tilde{F})g$. Hence, we obtain a *global* definition for the Darboux derivative $\omega : U \to \mathfrak{so}(3)$ of a symmetric moving frame F: at any point p, ω_p is just the Darboux derivative of any local trivialization. As long as ω satisfies the usual structure equation, it then encodes a well-defined 3D cross field over any nonsingular and simply-connected region of U. If it satisfies the augmented structure equation (Eqn. 28) for an Ω with fractional indices $\sigma_i \in \frac{\pi}{2}\mathbb{Z}$, then it describes a 3D cross field with singular curves.

A.3.1 Nonsimply-Connected Domains. More generally, let U be any path connected domain, and let ω be an $\mathfrak{so}(3)$-valued 1-form on U satisfying the local structure equation. Suppose that the monodromy of each generating loop $\gamma \in \pi_1(U, b)$ is conjugate to some cube symmetry $g \in \Gamma$ with respect to the same frame $E_b \in \mathrm{SO}(3)$, *i.e.*, $\Phi_\omega(\gamma) = E_b g E_b^{-1}$. Let $\tilde{\gamma}$ be the development of ω along γ, starting at $\tilde{\gamma}(0) = E_b$. Then the (right) quotient of $\tilde{\gamma}$ by Γ is a closed loop in $\mathcal{C}$, *i.e.*, there is a well-defined 3D cross field along γ. By arguments virtually identical to those in Sharpe [2000, Chapter 3.7], the same will be true for any loop based at any point, *i.e.*, ω is then the Darboux derivative of some 3D cross field on U. Moreover, this cross field is almost always unique: if we try to develop ω around a loop γ starting with any frame $\widetilde{E}_b$ that is *not* equivalent to E_b, then in general the final frame will not be equivalent to the initial frame, *i.e.*, we do not obtain a consistent cross field (consider Fig. 8). The only exception is when the monodromy around *every* loop is trivial in the usual sense, *i.e.*, if g is always equal to the identity—in this case, as with ordinary frame fields, ω determines the field only up to a choice of global rotation.

A.4 Relationship to Discrete Algorithm

The discrete algorithm in Secs. 2 and 3 is a straightforward discretization of the smooth formulation described above. In particular:

Darboux Derivative. The discrete Darboux derivative (Sec. 1.5) can be given the following interpretation. Consider a frame rotating at a constant angular velocity ω_{ij}/ℓ_{ij} along each edge ij, where ℓ_{ij} is the edge length. The values ω_{ij} then coincide with the integral of the smooth Darboux derivative ω along each edge (which can be arbitrarily large). Moreover, since $R_{ij} = R_{ji}^{-1}$, we have

$$\omega_{ij} = \log(E_j(R_{ij}E_i)^{-1}) = -\log(E_i(R_{ji}E_j)^{-1}) = -\omega_{ji},$$

i.e., it reverses sign with a change in orientation. Hence, ω_{ij} is a *discrete differential 1-form* in the sense of discrete exterior calculus [Hirani 2003; Desbrun et al. 2006].

Dirichlet Energy. The smoothness of a map $E : U \to \mathrm{SO}(3)$ can be measured via the *Dirichlet energy*

$$\mathcal{E}_D = \int_U |dE|^2 \, dV. \tag{29}$$

Since E is orthogonal we have $|dE|^2 = |(dE)E^{-1}| = |\omega|^2$, and can hence write the Dirichlet energy as

$$\mathcal{E}_D = \int_U |\omega|^2 \, dV.$$

The same energy can also be applied to cross fields, since ω depends only on a local trivialization (App. A.3). The discretization in Eqn. 11 and 16 is then obtained via the diagonal norm on discrete differential 1-forms [Desbrun et al. 2006, Section 5.4].

Integrability Conditions and Constraints. As outlined in Sec. 3.1, our discrete structure equation is a direct translation of the smooth structure equation using the discrete exterior derivative and primal-primal wedge product from discrete exterior calculus [Hirani 2003, Sections 3.6 & 7.1]; it also coincides with a 2nd-order expansion via the Baker-Campbell-Hausdorff formula. Eqn. 20 is derived by viewing both α and N as piecewise linear functions interpolating values α_i, N_i (*resp.*) at vertices. Since $d\alpha$ is piecewise constant, integrating Eqn. 30 along edge ij yields

$$\int_{ij} \omega^0 + (d\alpha)\widehat{N}\, ds = \omega_{ij}^0 + \int_{ij} \tfrac{\alpha_j - \alpha_i}{\ell_{ij}} \widehat{N}\, ds = \omega_{ij}^0 + \tfrac{1}{2}(\alpha_j - \alpha_i)(\widehat{N}_i + \widehat{N}_j).$$

Nearly identical calculations yield Equations 21 and 22.

A.4.1 Rotational Invariance of Boundary Singularities. Consider a pair of 3D frame fields $E, \widetilde{E}$ on the boundary ∂U, and assume that $\widetilde{E}$ is a pointwise rotation of E around the normal N by some smoothly-varying angle $\alpha : \partial U \to \mathbb{R}$, *i.e.*, $\widetilde{E} = \exp(\alpha\widehat{N})E$. Equivalently, if E is encoded in global orthonormal coordinates (e_1, e_2, e_3) such that $Ee_3 = N$, we can write $\widetilde{E} = E\exp(\alpha\hat{e}_3)$, *i.e.*, rotate first around e_3, then apply the frame. Noting that $d\exp(\alpha A) = (d\alpha)A\exp(\alpha A)$ for any fixed matrix A, we get

$$\begin{aligned} d\widetilde{E} &= dE\exp(\alpha\hat{e}_3) + Ed\exp(\alpha\hat{e}_3) \\ &= \omega E\exp(\alpha\hat{e}_3) + (d\alpha)\widehat{N}E\exp(\alpha\hat{e}_3) \\ &= \omega\widetilde{E} + (d\alpha)\widehat{N}\widetilde{E}, \end{aligned}$$

which means the Darboux derivatives of E and $\widetilde{E}$ are related by

$$\widetilde{\omega} = \omega + (d\alpha)\widehat{N}. \tag{30}$$

To see that these fields have the same singularities, we simply need to compute their curvature 2-forms $\Omega, \tilde{\Omega}$. From Eqn. 28 we get

$$\begin{aligned} \widetilde{\Omega} &= d\widetilde{\omega} - \widetilde{\omega} \wedge \widetilde{\omega} \\ &= \Omega + d\alpha \wedge d\widehat{N} - \omega \wedge (d\alpha)\widehat{N} - (d\alpha)\widehat{N} \wedge \omega - (d\alpha)\widehat{N} \wedge (d\alpha)\widehat{N} \\ &= \Omega + d\alpha \wedge \left(d\widehat{N} + \omega\widehat{N} - \widehat{N}\omega\right). \end{aligned}$$

Since E is adapted to the boundary, the normal satisfies $dN = \omega N$. In turn, the skew-symmetric matrix $\widehat{N}$ is solution of the equation $d\widehat{N} = \widehat{N}\omega - \omega\widehat{N}$. Hence, the curvature 2-forms of E and $\widetilde{E}$ are equal.

Received January 2019