DISCRETE VECTOR AND 2-TENSOR ANALYSES AND APPLICATIONS

By

Beibei Liu

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science – Doctor of Philosophy

2015

ABSTRACT

DISCRETE VECTOR AND 2-TENSOR ANALYSES AND APPLICATIONS

By

Beibei Liu

We present novel analysis methods for vector fields and an intrinsic representation of 2-tensor fields on meshes, and show the benefits they bring to discrete calculus, geometry processing, texture synthesis and fluid simulation. For instance, such vector fields and tensor fields in flat 2D space are necessary for example-based texture synthesis. However, many existing methods cannot ensure the continuity automatically or control the singularities accurately. Moreover, extending such analyses to curved surfaces involves several known challenges. First, vectors at different surface points are defined in different tangent planes, so their comparison necessarily involves a concept called *connection* to transport vectors from one tangent plane to another in a parallel way. The few existing approaches for discrete connections offer neither a globally optimal principled definition nor a consistent disretization of differential operators. Second, symmetric 2-tensors, which play a crucial role in geometry processing, are often discretized as components stored in the predefined local frames. There is no convenient way to perform coordinate-independent computations with arbitrary 2-tensor fields on triangulated surface meshes. Finally, the persistent pursue for efficiency in the processing of vector fields in applications such as incompressible fluid simulation often results in undesired artifacts such as numerical viscosity, which prevents a predictive preview for the fine-resolution simulation at coarse spatial and temporal resolutions.

We offer solutions to address these issues using our novel representation and analysis tools.

First, we present a framework for example-based texture synthesis with feature alignment to vector fields with two way rotational symmetry, also known as orientation fields. Our contribution is twofold: a design tool for orientation fields with a natural boundary condition and singularity control, and a parallel texture synthesis adapted specifically for such fields in feature alignment.

Second, we define discrete connection on triangle meshes, which involves closed-form expressions within edges and triangles and finite rotations between pairs of incident vertices, edges, or triangles. The finite set of parameters of this connection can be optimally computed by minimizing a quadratic measure of the deviation from the connection induced by the embedding of the input triangle mesh. Local integrals of other first-order derivatives as well as the L_2-based energies can also be computed.

Third, we offer a coordinate-free representation of arbitrary 2-tensor fields on triangle meshes, where we leverage a decomposition of continuous 2-tensors in the plane to construct a finite-dimensional encoding of tensor fields through scalar values on oriented pieces of a manifold triangulation. We also provide closed-form expressions of common operators for tensor fields, including pairing, inner product, and trace for this discrete representation, and formulate a discrete covariant derivative induced by the 2-tensors instead of the metric of the surface. Other operators, such as discrete Lie bracket, can be constructed based on these operators. This approach extends computational tools for tensor fields and offers a numerical framework for discrete tensor calculus on triangulations.

Finally, a spectral vector field calculus on embeded irregular shape is introduced to build a model-reduced variational Eulerian integrator for incompressible fluid. The resulting simulation combines the efficiency gains of dimension reduction, the qualitative robustness to coarse spatial and temporal resolutions of geometric integrators, and the simplicity of sub-grid accurate boundary conditions on regular grids to deal with arbitrarily-shaped domains. A functional map approach to fluid simulation is also proposed, where scalar-valued and vector-valued eigenfunctions of the Laplacian operator can be easily used as reduced bases. Using a variational integrator in time to preserve liveliness and a simple, yet accurate embedding of the fluid domain onto a Cartesian grid, our model-reduced fluid simulator can achieve realistic animations in significantly less computation time than full-scale non-dissipative methods but without the numerical viscosity from which current reduced methods suffer.

To my dear family

ACKNOWLEDGMENTS

My five years study in the PhD program is like a challenging but exciting journey, which would not have been possible without the guidance and help from many people.

Foremost, I would like to express my sincerest gratitude to my advisor Yiying Tong, who has been supporting and guiding me the whole time during the PhD program. This thesis would not have been possible without his patient guidance and endless encouragement. I can still recall many deadlines we had to work hard together and the great joy we shared in the last five years. Yiying is full of creative ideas and valuable suggestions, which is why I nicknamed him as 'A Walking Wikipedia' . At the same time, he always generously shares his own experience and wishes the best for his students. Yiying Tong has set an example of excellence as a researcher, instructor, teacher and role model, inspiring me to be a better person. Besides my advisor, I would like to thank the rest of my committee members: Leslie Kuhn, Charles B. Owen and Joyce Chai for their encouragement, insightful comments and questions.

It has been a great honor for me to collaborate with many awesome researchers. My special thanks go to Mathieu Desbrun for generously recommending me the summer internship and offering me the postdoctoral scholar position with him. We have successful collaborations on multiple projects where his intuitive and insightful interpretation of the research problems always broadens my horizon. I am also grateful to him for patiently reading and revising my thesis. Meanwhile I would like to thank Fernando de Goes for the tireless and inspiring discussion, Yuanzhen Wang for his patient guidance on the coding and explanation about the differential forms, Gemma Mason for her excellent writing of the fluid project, Julian Hodgson for the appealing rendering results, Jiannan Wang for the nice-annotated code of the texture project, Xiaojun Wang and Feng Xin for their valuable help in installing multiple libraries, Daniel Dault, Jie Li, Rundong Zhao and Balasubramaniam Shanker for their creative ideas in the subdivision projects.

Last but not the least, I would like to thank my parents for their unconditional love and support.

They respect every big decision I made and are always my strongest backing. I am particularly grateful to my husband and soul mate, Lanbo She for giving up his graduate program in China only to join me in MSU. He always tolerates me and encourages me to embrace different challenges bravely. I would also like to thank my dearest friends whom I shared the ups and downs during this wonderful journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In the increasingly digitized world, smooth shapes are often approximated by their discrete counterparts, as computers can only store and process a finite number of digits. Furthermore, continuous processes described by differential equations also need to be adapted to analyze values stored on these discretized domains, described by, *e.g.*, spatial or temporal sample points. However, ad-hoc discretization often fails to keep the global geometric structures, i.e., the *symmetries* and *invariants* that define the geometry. An emergent research field, called Discrete Differential Geometry (DDG), seeks to systematically construct the discrete counterpart of geometric models (in particular, smooth surfaces) from the smooth theory while preserving its fundamental properties. DDG draws upon both differential geometry—focused on properties of smooth manifolds—and the field of computational geometry—concerned with discrete combinatorial geometric objects. The applications of DDG include geometry processing, applied mathematics, computer graphics, and physical simulation. Readers interested in the broad concept of DDG can refer to recent surveys, such as [11, 4] for more details.

In this dissertation, we mainly focus on vector and tensor field analysis and their applications such as texture synthesis and fluid simulation. We first provide mathematical background on these specific topics in Chapter 2, and then go over the various challenges in scientific computing involving vectors and tensors on tessellated domains, the current issues with existing methods that tackle these challenges, and our approach to addressing these issues.

1.1 Vector Field Analysis and Application

Vector field, which assigns a direction and a magnitude per point, is essential in a wide range of applications, including geometry processing, example-based texture synthesis, nonphotorealistic

rendering, and fluid simulation. For instance, in example-based texture synthesis, a vector field is used as a guidance for the local direction and sizing when tiling exemplars in order to synthesize a large image (Fig. 1.2). Fig. 1.1 shows another application where vector fields help find a natural point-to-point mapping between two different shapes.

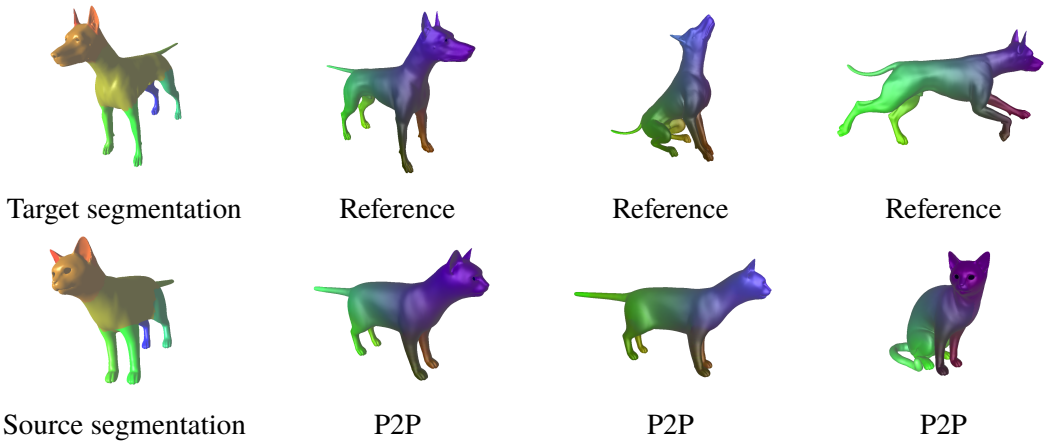


Figure 1.1 **Example application of vector fields: Point-to-point mesh correspondence.** The P2P mapping between dog and cat models with only segmentation information as input.

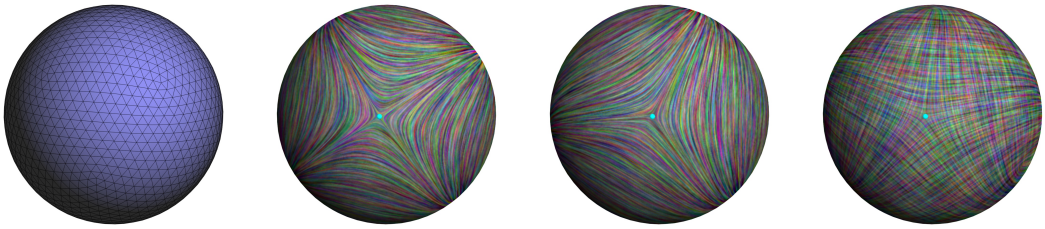
Discrete representation of vector fields Vector fields on triangulated surfaces are often discretized through local coordinates in orthogonal frames defined either on vertices or on faces. A continuous vector field over a mesh is typically evaluated from this finite set of vectors based on piecewise constant interpolation [12] or, to increase smoothness, using nonlinear basis functions derived from the geodesic polar map [13, 1]. In an effort to remove the need for coordinate systems, scalar potentials were proposed as an intrinsic encoding of tangent vector fields: while Tong et al. [14] used two potential values per node interpolated with linear finite elements, Polthier and Preuss [15] offered a discrete notion of Hodge decomposition with proper cohomology by using one value per node and one value per unoriented edge interpolated with conforming and non-conforming linear basis functions respectively. This representation is, however, limiting as it only leads to per-face constant vector fields. Operator-based representations have also been recently proposed [16, 6], but their use is, to date, too restrictive to be widely adopted. Finally, a coordinate free approach to vector field representation was introduced through the use of algebraic topology and

exterior calculus [17] where vector fields are identified to discrete differential 1-forms (i.e., rank-1 tensors of type $(0, 1)$) and interpolation is performed via Whitney basis functions [18]. These edge-based discrete tangent vector fields have since then been shown useful in a variety of applications [19, 20]. The discrete 2-tensor fields proposed in Chapter 5 are fully compatible with this specific form-based representation, and even provide a discrete notion of covariant derivative of vector and covector fields.

Figure 1.2 **Example-based texture synthesis on surface and plane [2].** The square tiles show examplars that are used to synthesize a texture on surfaces guided by precomputed vector fields.

N-way Rotational Symmetry Field An N-way rotational symmetry (RoSy) field is a more general case of direction field, where the direction is considered invariant under rotation $2\pi/N$ (see Fig. 1.3), or as the equivalence class of the N directions along evenly distributed angles. The special case of 2-RoSy field has been known as orientation field, and long been used in fingerprint research, e.g., in [21], since the ridges and valleys on a fingerprint image do not have distinct forward or backward directions along them. The models commonly used for detecting singularities in fingerprints are often with few parameters, but more accurate orientation fields are proven important in enhancing latent fingerprints collected at crime scenes [22]. In graphics, Zhang et al. [23] introduced interactive 2-RoSy design on surfaces, and Palacios et al. [24] extended this idea to N-RoSy fields with $N \geq 3$. Building N-RoSy fields can also be achieved through specifying compatible

singularities and modifying parallel transport: Lai et al. [25] focused on designing Riemannian metrics compatible with the local symmetry of N-RoSy fields, while Crane et al. [26] proposed to directly design a connection that is flat almost everywhere except at singularities, instead of using the Levi-Civita connection induced by the Riemannian metric. Ray et al. [27] introduced the concept of turning numbers and offered another equivalent definition for singularities of N-RoSy fields.



(a) sphere mesh (b) vector field (c) orientation field (d) cross field

Figure 1.3 **n-vector fields on sphere mesh.** A sphere mesh (a) and a vector field with one saddle singularity (b). Its corresponding 2-RoSy field is (c) and its 4-RoSy field is (d). Results computed by the method in Chapter 4.

1.1.1 Orientation field guided texture synthesis

A direct application of orientation fields in graphics is texture synthesis. Texture, carrying information about geometric details or material properties and stored as a 2D image, is indispensable in decorating 3D surfaces. While manually producing a texture is labor-intensive and requires artistic skills, texture synthesis technique is automatic when examplars are given and local directions are specified. It is widely used in applications such as rendering, image editing and video synthesis, and extensively practiced in industry, e.g., in game engines and feature films [28]. However, a precomputed vector field used to guide the tiling of a texture still needs to be designed. For textures with two way rotational symmetry, the guidance fields do not have to be continuous everywhere, where the vectors should be allowed to have nearly opposite directions to have more natural control. To generate large textures satisfying user specification, some key requirements need to be in place for the method to be practical.

Challenges An *intuitive* control is often highly beneficial in the design process of guidance fields. For instance, if nonintuitive Dirichlet or Neumann boundary conditions are used (i.e, the pre-designed vector field is tangential/perpendicular to the boundary) , an artist may have to spend extra time figuring out their influence when preparing feature alignment fields, whereas a natural boundary condition can lead to an expected smooth field without additional user intervention. *Smoothness* in the final texture is another basic requirement for seamless appearance of the objects being decorated. To offer additional flexibility, guidance fields can be N-RoSy fields.

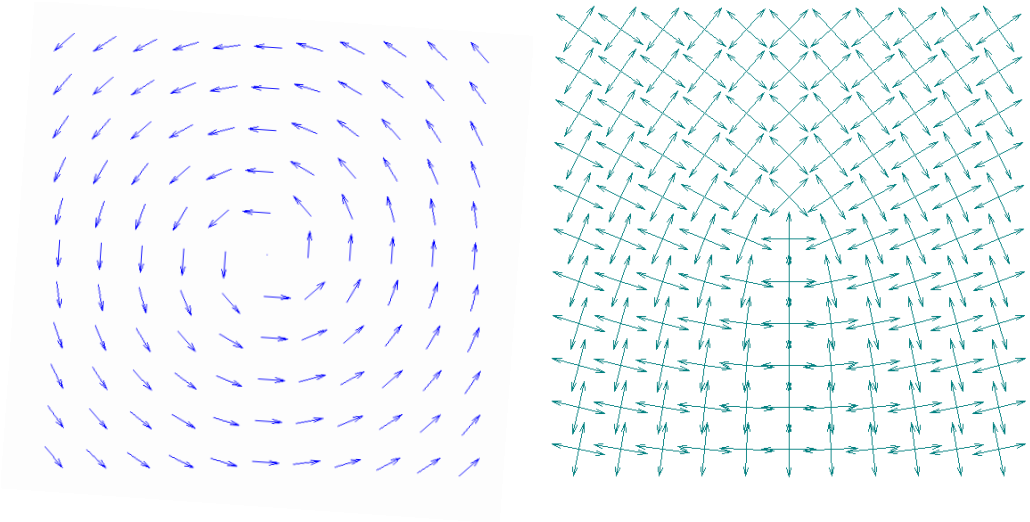


Figure 1.4 **Vector field (left) and its corresponding 4-RoSy field(right).**

For instance, the 4-RoSy field generated from the vector field (in Fig. 1.4) allows more ways to put together checkerboard or cross-like textures. At each point, we have an equivalent set of four vectors, and we can choose any of them as the *representative* vector. Such a *representative* vector field is not continuous, while the associated *crosses* can still be continuous everywhere in the picture. Real-world textures often contain N-way rotational symmetries, that is, the pattern would nearly coincide with itself after rotating by an angle of $\frac{2\pi}{N}$. These symmetries allow patterns to be aligned continuously not only with smooth vector fields, but with direction fields that are not smooth in the traditional sense. Thus, using procedures designed with vector fields as an input for generating seamless textures can be challenging when orientation fields are used in regions with discontinuity in the choice of forward or backward directions. Special attention must be paid

in the treatment of neighborhoods containing such discontinuity, which is inevitable for generic orientation fields. Furthermore, a design algorithm should be efficient enough to make the system interactive and flexible.

In Chapter 3 we introduce a novel framework for the entire pipeline of orientation field guided texture synthesis [29]. Our main contributions include

- A tangent vector field design tool with natural boundary conditions based on the minimization of the Dirichlet energy.
- An orientation field design tool based on an associated vector field, with straightforward control over singularities and their orientations.
- A parallel texture synthesis adapted to handle any discontinuity in an orientation field.

1.1.2 Vector field analysis

As differential calculus is one of fundamental mathematics tool for *curved* surfaces, differential analysis of vector fields is often mandatory for geometry processing, fluid simulation, etc. In Sec 2.2 we introduce the mathematical background for comparing nearby tangent vectors in a geometric sense, through the concept of connection. Intuitively, a connection prescribes in a given local frame field how the frame at one point should be modified to produce a *parallel* frame at a nearby point, so as to allow the comparison between vectors (and tensors) in nearby frames.

Various discretization methods of connection have been proposed. For instance, [30] used what conceptually amounts to Christoffel symbols between vertex-based tangent planes to describe the effects of parallel transport, in an effort to introduce linear rotation-invariant coordinates. However, these coefficients end up bearing little resemblance to their continuous equivalents. Kircher and Garland [31] proposed a triangle-to-triangle connection in the context of free-form deformation, but no notion of differentiation was discussed. A formal discrete version of connections between triangles was defined in [26], encoding the alignment angle for parallel transport from one triangle

to an adjacent one, and with which piecewise-constant unit vector and n-direction fields can be derived for any given set of singularities.

The recent work of [1] defines a notion of parallel transport through the blending of geodesic polar maps similar to [13], which determines a connection between vertices as opposed to triangles. This approach results in a continuous notion of vector fields (and n-vector fields) compared to the piecewise constant discretization per face of [26, 32, 33], and thus allows a formal evaluation of the Dirichlet energy. Their choice of connection is based on the even distribution of the Gaussian curvature of the input mesh from vertices to faces, which leads to closed-form expressions of the L_2 integrals they sought. However, the deviation (and thus, the discretization error) of their connection from the Levi-Civita connection, a canonical connection induced by the embedding of the mesh in $\mathbb{R}^3$, is difficult to quantify since no closed-form expression of the covariant derivative itself was provided. Finally, first-order derivative operators such as divergence or curl cannot be evaluated in their framework—neither pointwise, nor as local integrals. The work of [34] provides discrete covariant derivatives induced by discrete symmetric 2-tensors as a global mapping from a pair of discrete 1-forms to another discrete 1-form, but offers no pointwise expressions either.

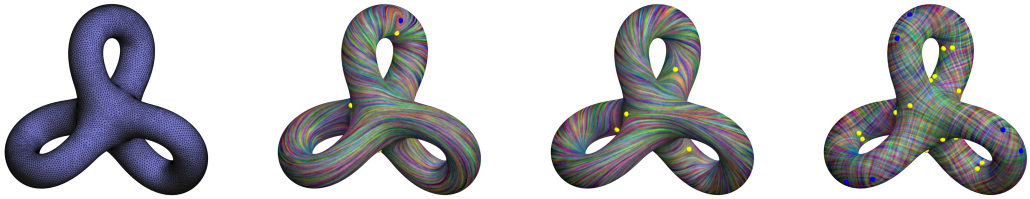
Overall, no current approaches offer a discrete connections that can be argued to be optimally close to the canonical connection introduced by its embedding, neither do they provide the discrete operators to capture local or global derivatives consistently. In Chapter 4, we introduce discrete counterparts of these terms. Our contributions include:

- A discrete connection with closed-form which is as-Levi-Civita-as-possible.
- A closed-form expression for the covariant derivative, offering pointwise or integral evaluations of first-order derivative operators and relevant energies.

Significant numerical improvements over previous methods are obtained for analytical vector fields when this as-Levi-Civita-as-possible discrete connection is used for discrete operators on vector fields. We also demonstrate the relevance and practical use of our discrete connections by contribut-

ing new numerical tools for n-vector field and n-direction field editing that control the position and orientation of both positive and negative singularities.

(a) cow mesh (b) vector field (c) orientation field (d) cross field



(a) tripletorus mesh (b) vector field (c) orientation field (d) cross field

Figure 1.5 **Smoothest n-vector field on different shapes.** A surface mesh (a) and its corresponding smoothest vector field (b), 2-RoSy field (c) and 4-Rosy field (d). Results computed by the method in Chapter 4.

1.1.3 Spectral vector field processing for fluid simulation

Accurate simulation of incompressible fluids is another application of vector fields, and a well-studied topic in computational fluid dynamics. Fluid animation research is driven by a different emphasis: in the context of computer graphics, the focus is on capturing the visual complexity of typical incompressible fluid motions (such as vortices and volutes) with minimum computational cost. Early computer animation Eulerian methods for incompressible fluid simulation were based on explicit finite differences [35] which suffered from the slow convergence of their iterative approach to divergence-free projection. Stam [36] introduced semi-Lagrangian advection and a sparse Poisson solver which brought much improved efficiency and stability. However, these improvements came at the cost of significant dissipation—a common issue that one can partially mitigate via vorticity confinement [37], reinjection of vorticity with particles [38], or curl correction [39]. Significantly less dissipative time integrators were also proposed through semi-Lagrangian advec-

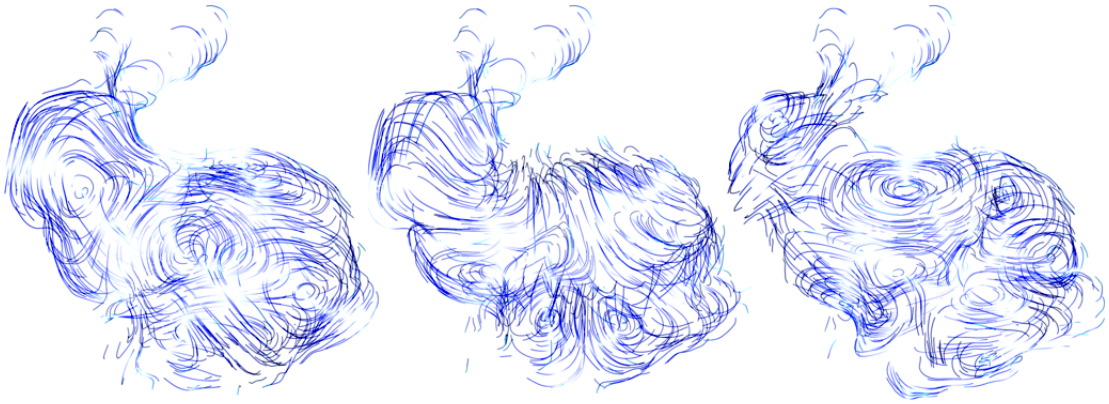


Figure 1.6 **Model-reduced fluids on regular grids.** Our proposed energy-preserving approach integrates a fluid flow variationally using a small number of divergence-free velocity field bases over an arbitrary domain (visualized here are the 5^{th}, 10^{th}, and 15^{th} eigenvectors of the 2-form Laplacian) computed with subgrid accuracy on a regular grid (here, a $42 \times 42 \times 32$ grid) through the method introduced in Chapter 6. This integrator is versatile: it can be used for realtime fluid animation, magnetohydrodynamics, and turbulence models, with either explicit or implicit integration.

tion of vorticity [19], or even energy-preserving methods [40]. However, these improved numerical methods often carry higher computational costs. Consequently, coupled Eulerian-Lagrangian (hybrid) methods (see, for instance, [41, 42]) have flourished recently, as they offer a good compromise between efficiency and dissipation.

Handling boundaries well is also crucial for incompressible fluids, as boundary layers can significantly impact fluid motion. Avoiding the staircase effects that voxelized domains generate was achieved using simplicial meshes or hybrid meshes [43], but irregular connectivity often affects the efficiency of the solvers involved. Inspired by the immersed boundary and interface methods, the use of regular grids with modified numerical operators to handle arbitrary domains was proposed in [44], then made convergent by [7] while maintaining symmetry of the solves needed by the integrators. Another approach using virtual nodes was also proposed recently [45].

Fluid simulation over non-flat domains has received significant attention as well. Most notably, Stam adapted his Stable Fluid method to handle curvilinear coordinates [46], while Azencot et al. [47] recently proposed to use the Lie derivative operator representation in the spectral domain to represent a velocity field on an arbitrary surface, and performed advection of vorticity through a linearized exponential map of the operator representation. Methods that are using only intrinsic

operators can also handle curved domains without alterations [19, 40].

In Chapter 6, we formulate a model-reduced variational fluid integrator that combines the benefits of *non-dissipative* integrators with the use of *dimension reduction* and *Cartesian grids* over arbitrary domains. Based on a description of the fluid motion through functional maps, a variational integrator is derived from Hamilton's principle [48, 49], resulting in a Lie algebra integrator with non-holonomic constraints [50, 10]. We use spectral approximation of the functional map through (cell-based) scalar and (face-based) vector Laplacian eigenvectors in order to offer model reduction without losing the variational properties of the integrator, with controllable energy cascading. This setup allows us to use not only low frequencies to capture the basic behavior of a flow, but also a few selected higher frequencies to add realism at low cost. Furthermore, we extend the embedded-boundary approach of [7] to our framework in order to compute spectral (scalar- and vector-valued) basis functions of arbitrary domains directly on *regular grids* for fast computations with sub-grid accuracy. Finally, our approach uses the typical Eulerian setup of flux-based solvers; consequently, addition of fine details through spectral noise [51], wavelet [52], empirical mode decomposition [53], subgrid turbulence [54, 55], curl correction [39], or through enforcing Lagrangian coherent structure [56] can be done straightforwardly. We demonstrate the efficiency of our resulting integrator through a number of examples in 2D, 3D, and curved 2D domains, as well as its versatility by pointing out how to extend its use to magnetohydrodynamics, subgrid scale models, and other fluid equations. Our approach thus extends the variational approach of [40, 10] to arbitrary reduced bases, adopts the (now Eulerian) vorticity advection of [57], and offers a structure-preserving version of the Laplacian-based integrator of [9].

1.2 Discrete 2-Tensor Fields on Triangulated Surface Mesh

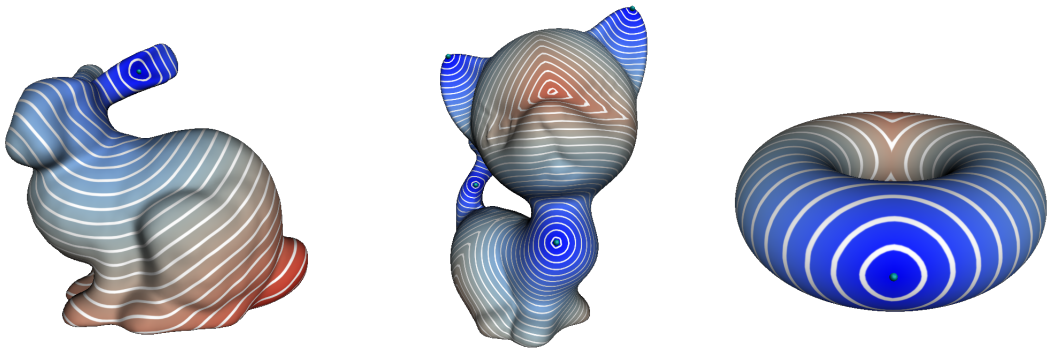
Tensors of rank two are commonplace in geometry processing, e.g., as a way to encode sizing and orientation fields for meshing purposes [58, 59, 60]. While discrete representations of tangent vector fields and antisymmetric tensors (i.e., forms) on triangulations have been widely used,

few approaches provide convenient ways to perform computations with arbitrary 2-tensor fields on triangulated surface mesh, unlike their lower rank couterparts. Much like early work on discrete vector fields, they are often defined by first establishing a local tangent space basis per vertex [59], edge [61], or face [62, 63, 64], then storing the four components of the tensor in each of these frames. A notable coordinate-free alternative exists for purely *antisymmetric* 2-tensors (called 2-forms): they are scalar multiples of $J = \left(\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right)$ in any orthogonal coordinate frame since $RJR^t = J$ for an arbitrary rotation matrix R. They can thus be encoded as discrete differential 2-forms via one scalar per face and Whitney basis functions [65, 4]. (Dual 2-forms, defined per dual cell, can also be used.) In comparison, symmetric tensors have rarely been discussed in the discrete realm [66, 67], yet they are implicitly behind all inner products of forms or vectors, generalized Laplacian operators [68], and the notion of Hodge star [69]. Our discrete encoding of tensors encompasses both symmetric and antisymmetric tensors in a consistent framework.

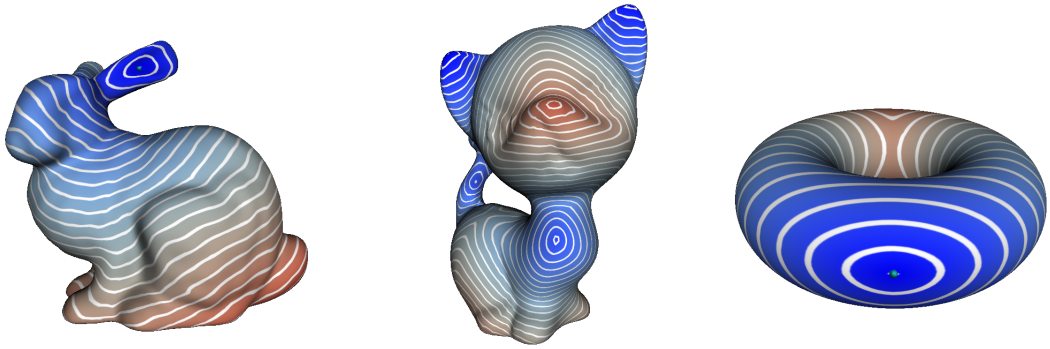
In Chapter 5, we introduce a numerical framework to encode and manipulate 2-tensors on triangle meshes. Our work [34] is based on a novel coordinate-free decomposition of continuous 2-tensor fields in the plane. By leveraging this decomposition, we construct a finite-dimensional representation of 2-tensors on discrete surfaces that is fully compatible with the DEC [4] and FEEC [65] machinery. Our discrete 2-tensors exactly mimic the continuous notion of divergence-free, curl-free, and traceless tensors, and recover many well-known discrete operators commonly used in geometry processing. Finally, our approach offers a discrete counterpart to both covariant derivative and Lie bracket of 1-forms (or vector fields), and provides an extension of the heat method [3] to compute anisotropic geodesics.

1.3 Organization

The rest of dissertation is organized as follows. We first recap important mathematical concepts used throughout the dissertation in Chapter 2. Then, we discuss the 2D orientation field design with natural boundary conditions in Chapter 3. A principled approach to connection discretization



(a) isotropic geodesic (a) isotropic geodesic (a) isotropic geodesic



(b) anisotropic geodesic (b) anisotropic geodesic (b) anisotropic geodesic

Figure 1.7 **Influence of the metric tensor on geodesics.** (a) visualization of an isotropic geodesic where the distance is generated by the heat method [3]. The curvature tensor of a surface is applied to compute anisotropic geodesics (b) with our generalized Laplacian operator proposed in Chapter 5.

on curved surfaces and its application to N-RoSy design is presented in Chapter 4. Discretization of 2-tensor fields and its application in anisotropic geodesic calculation is discussed in Chapter 5 and a spectral vector field processing on irregularly shaped 3D domain is introduced to build a novel model-reduced variational integrator for fluid simulation in Chapter 6. Finally, we summarize the contributions from this thesis and discuss future work in Chapter 7.

CHAPTER 2

MATHEMATICAL BACKGROUND

In this chapter, we present the necessary mathematical background of the continuous theory and an existing discrete structure-preserving counterpart of a collection of tools in continuous vector field analysis called *exterior calculus*. Some basic concepts, including manifolds, tangent vectors, tensors, differential forms are introduced in Sec 2.1. *Connection* and *covariant derivative* operators are discussed in Sec 2.2, whose discrete counterparts are proposed in Chapter 4. Sec 2.3 presents the decomposition of tensor fields into simpler building blocks, which enables our intrinsic representations in Chapter 5. Finally, Sec 2.4 introduces a tool used throughout this dissertation, namely, *Discrete exterior calculus* (DEC), which provides a discrete representation of the *differential forms* with their differential and integral calculus. Note that we restrict the exposition in Sec 2.2 and Sec 2.3 to the surface case only.

2.1 Basic Concepts from Differential Geometry

Classical differential geometry focuses on the study of the geometry (like curves and surfaces) through the techniques of linear algebra, differential calculus and integral calculus. In this section, we introduce some basic concepts involved in our work and restrict our discussion to subsets of $\mathbb{R}^3$. Note that the goal here is to provide readers with some prior knowledge in differential geometry and the intuition behind key concepts, rather than rigorous mathematical definitions of these terms. Refer to, *e.g.*, Lee's book[70] if further details are desired.

The basic shapes or domains used in differential geometry are described by manifolds. An *n-manifold* M is a space that locally looks like flat Euclidean space $\mathbb{R}^n$, but may have different global topology. For instance, the surface of Earth can be treated as a 2-manifold, since no matter where you stand on Earth, it looks locally like a flat $2D$ domain. Euclidean spaces, curves, and surfaces

are all special cases of manifolds.

2.1.1 Tensors

Quantities stored on manifolds can be represented through functions with multiple components. A particularly important class of multiple component functions are called tensors. We introduce the vectors and covectors before describing the tensors. We then discuss the differential and integral calculus of tensors.

Tangent vector. For a point $\mathbf{p}$ on a surface M embedded in $\mathbb{R}^3$, a tangent vector $\mathbf{v}$ at $\mathbf{p}$ is just a tangent vector to a curve on the surface M that passes through $\mathbf{p}$. The set of all tangent vectors to M at $\mathbf{p}$ is called the *tangent plane* of manifold M at point $\mathbf{p}$, which is often denoted as T_pM.

Covector. For a finite-dimensional vector space V. A *covector* on V is defined as a real-valued linear function on V as $\omega : V \to \mathbb{R}$. The space of all the covectors on V is also a vector space, which is the *dual space* to V and is often denoted as V^*. T_p^*M is thus the dual space of T_pM, sometimes called the cotangent space, the elements of which are called (tangent) covectors, cotangent vectors, or 1-forms.

Tensor. Just as a covector can be seen as a linear operator that maps a vector to a real number, vector can be seen as a linear operator which maps a covector to a real number, since for finite-dimensional vector spaces, the dual space of the dual space is identical to the original space. A rank-(m, n) *tensor* is a generalization of both vector and covector, in the sense that it can be treated as a multilinear operator mapping m covectors and n vectors to a scalar. Scalar fields, vector fields and covector fields are just rank-0, rank-(1,0), and rank-(0,1) tensor fields, respectively. A *metric tensor* is a symmetric rank-$(0, 2)$ tensor that is positive definite. It takes a pair of tangent vectors as input and produces a scalar value, which provides a distance measurement on the manifold. Given a basis frame of the tangent space, a metric tensor can be represented as a symmetric positive-definite matrix. In a local basis field $(\mathbf{X_u}, \mathbf{X_v})$ of tangent spaces on a surface, a vector field can be

expressed as $\mathbf{w} = u\mathbf{X_u} + v\mathbf{X_v}$. The length of a vector can be calculated as:

$$g(\mathbf{w}, \mathbf{w}) = (u, v) \begin{pmatrix} E & F \\ F & G \end{pmatrix} (u, v)^T \tag{2.1}$$

where $E = \mathbf{X_u} \cdot \mathbf{X_u}$, $F = \mathbf{X_u} \cdot \mathbf{X_v}$ and $G = \mathbf{X_v} \cdot \mathbf{X_v}$ provides the canonical embedding metric g of the surface. The length of a path can be evaluated as the integral of the tangent vectors along the path. For a flat plane with Cartesian coordinate system, the metric tensor is just the identity matrix.

Differential forms. A particularly useful tensor is the antisymmetric rank-$(0, p)$ tensor. A tensor is antisymmetric if swapping two vectors in the input list of the tensor, the output changes sign. Such tensors are called p differential forms, or just p-forms. They are tensor fields whose integral on p dimensional submanifolds are independent of the coordinate systems. For instance, differential forms can be integrated over curves(1-form), surfaces(2-form) and volumes(3-form).

Thus, covectors are 1-forms and antisymmetric rank-$(0, 2)$ tensors are 2-forms. In $\mathbb{R}^3$, p-forms can be expressed in bases spanned by forms such as dx, $dxdy$ or $dxdydz$:

- 0-form: smooth function f
- 1-form: integrand in line integral $fdx + gdy + hdz$
- 2-form: integrand in surface integral $fdxdy + gdzdx + hdydz$
- 3-form: integrand in volume integral $fdxdydz$

On an n-manifold, all k-forms with $k > n$ are zero, due to the *antisymmetry*.

2.1.2 Exterior calculus.

A set of operators can be defined on differential forms to form a differential calculus for integrable quantities on manifolds. We list these operators first before establishing the correspondence of them with the classical vector field analysis on surfaces. In the following, we denote the set of p-forms as Ω^p.

- *exterior derivative* d: $\Omega^k \to \Omega^{k+1}$;
- *contraction operator* i_X: $\Omega^k \to \Omega^{k-1}$, where X is a vector;
- *Hodge star* $\star$: $\Omega^k \to \Omega^{n-k}$;
- *wedge product* $\wedge$: $\Omega^p \times \omega^q \to \omega^{p+q}$.

On surfaces (2-manifolds), 0-forms and 2-forms have a single component, so they can be represented by scalar fields; 1-forms have two components, so they can be represented by vector fields (although they are in fact covector fields). In such representations, d applied to 0-forms produces gradients, d applied to 1-forms produces curl. i_X with a 2-form is just a scaling of X, and i_X with a 1-form is the dot product between the two vectors. The Hodge star $\star$ applied to 0-form as multiplication by area density, to 1-form as a rotation by $\pi/2$, and to 2-form as division by area density. The wedge product between 0-form and a p-form is just a scaling of the other form, the wedge product between two 1-forms is the cross product, which produces a scalar on surfaces.

2.2 Connections on Smooth Manifolds

In order to take derivatives of vector fields on manifolds, one must account for the fact that the vector field components are defined on different basis frames over different tangent spaces. Thus, one must first define a way to compare vectors living on nearby tangent spaces in a geometric manner. A *connection* plays such a role, as it maps a tangent vector (seen as an infinitesimal movement toward a nearby point) to an infinitesimal linear transformation aligning the local tangent plane to the tangent plane at that nearby point.

2.2.1 Definition

As we will review shortly, a connection is equivalent to a notion of parallel transport, i.e., how to slide tangent vectors along curves on M. A *metric connection* has the additional property that the

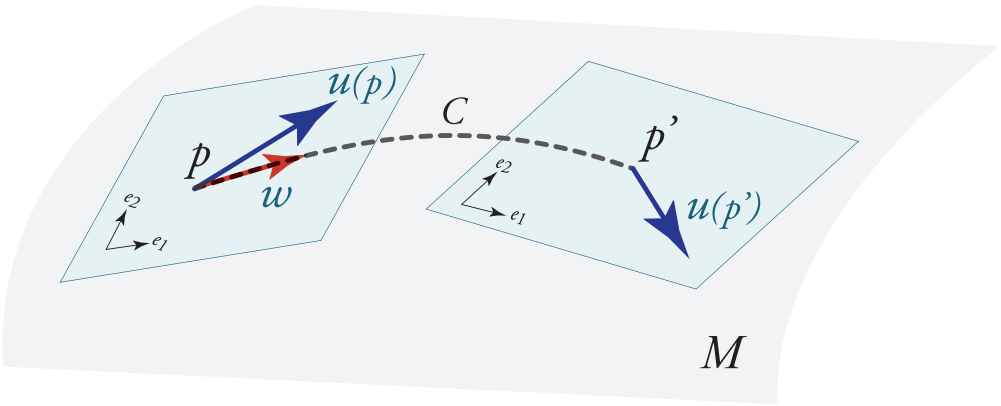


Figure 2.1 **Smooth connection.** On a smooth manifold, a connection indicates how a tangent vector at point $\mathbf{p}$ is parallel transported along a path C to a nearby point $\mathbf{p}' = \mathbf{p} + \epsilon\mathbf{w}$, accounting for the change of frame between the two tangent spaces. From a connection the notion of (covariant) derivative of vector fields is derived, as nearby vectors can now be compared.

metric on M is preserved by parallel transport. Given a frame field $(\mathbf{e}_1(\mathbf{p}), \mathbf{e}_2(\mathbf{p}))$, a connection is determined by the *coefficients* ω^i_{jk} defined as

$$\nabla_{\mathbf{e}_k}\mathbf{e}_j = \sum_i \omega^i_{jk}\mathbf{e}_i$$

Each coefficient ω^i_{jk} thus describes the i-th component of the change of the basis vector $\mathbf{e}_j$ along $\mathbf{e}_k$. (It can also be interpreted as the k-th component of a matrix-valued 1-form $\Omega = (\omega^i_j)$ as we will see next.)

The *Levi-Civita connection* of M is a special metric connection: it is the only one that satisfies $\omega^i_{jk} - \omega^i_{kj} = 0$ for frames formed by tangent vectors of isocurves of a parameterization. Note here we restrict the discussion of connection to the metric connection. For definitions of other connections defined on vector or frame bundles, we refer the readers to [71].

2.2.2 Connection 1-form

A connection can be regarded as a way of determining whether a vector $\mathbf{u}$ in the tangent space of $\mathbf{p}$ is parallel to a vector in the infinitesimally nearby tangent space of $\mathbf{p}'$ (see Fig. 2.1). In the local

frame $(\mathbf{e}_1(\mathbf{p}), \mathbf{e}_2(\mathbf{p}))$ at $\mathbf{p}$,

$$\mathbf{u}(\mathbf{p}) = u^1\mathbf{e}_1(\mathbf{p}) + u^2\mathbf{e}_2(\mathbf{p}) = (\mathbf{e}_1(\mathbf{p}), \mathbf{e}_2(\mathbf{p})) \begin{pmatrix} u^1 \\ u^2 \end{pmatrix}$$

The connection provides an expression for the vector at $\mathbf{p}'$ that is parallel to $\mathbf{u}(\mathbf{p})$ via an infinitesimal perturbation of the original coordinates:

$$\mathbf{u}'(\mathbf{p}') = (\mathbf{e}_1(\mathbf{p}'), \mathbf{e}_2(\mathbf{p}'))(I - \Omega(\mathbf{w})\epsilon) \begin{pmatrix} u^1(\mathbf{p}) \\ u^2(\mathbf{p}) \end{pmatrix},$$

which can then be compared against

$$\mathbf{u}(\mathbf{p}') = (\mathbf{e}_1(\mathbf{p}'), \mathbf{e}_2(\mathbf{p}')) \begin{pmatrix} u^1(\mathbf{p}') \\ u^2(\mathbf{p}') \end{pmatrix}.$$

Formally, the connection is thus defined as a matrix-valued 1-form Ω, which encodes that the frame $(\mathbf{e}_1(\mathbf{p}), \mathbf{e}_2(\mathbf{p}))$ is aligned to the frame

$$(\mathbf{e}_1(\mathbf{p}'), \mathbf{e}_2(\mathbf{p}'))(I - \Omega(\mathbf{w})\epsilon)$$

when moving along $\mathbf{w}$.

For simplicity, we assume that the local frame field is orthonormal, i.e. $(\mathbf{e}_1, \mathbf{e}_2) = (\mathbf{e}, \mathbf{e}^\perp)$, where $\mathbf{e}$ is a unit vector, and $\mathbf{e}^\perp$ is its 90°-rotation in a given metric g of the surface. Since we assumed that the connection is metric preserving (*i.e.*, that the basis vectors are parallel to two mutually orthogonal unit vectors in nearby frames), the connection Ω simplifies to an *antisymmetric* matrix representing an infinitesimal rotation of the form:

$$\Omega = \begin{pmatrix} \omega_1^1 & \omega_2^1 \\ \omega_1^2 & \omega_2^2 \end{pmatrix} = \begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix} = \omega J,$$

where J is the 90°-rotation matrix

$$J = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.$$

Therefore, given a frame field, the connection is defined by a rate-of-rotation-valued 1-form ω describing how fast the local frame should rotate to align to nearby frames when moving along a certain direction.

Curvature of connection The exterior derivative of the local representation of the connection ω defines, up to a sign, the curvature of the connection, and is denoted as $K = -d\omega$. Any closed path ∂R around a region R of the manifold therefore accumulates a rotation of frame, and it is equal to the integral of the connection curvature over R. For the canonical Levi-Civita connection, this curvature corresponds to the conventional notion of Gaussian curvature.

2.2.3 Covariant derivative on smooth manifolds

Geometric intuition Covariant derivative of a vector field $\mathbf{u}$ at a point $\mathbf{p}$ on a surface M is that $\nabla\mathbf{u}$ encodes the rate of change of $\mathbf{u}$ around $\mathbf{p}$ (check Fig. 2.1). Projecting the derivative of a vector field $\mathbf{u}$ along a vector $\mathbf{w}$ leads to a vector $\nabla_{\mathbf{w}}\mathbf{u}$, which indicates the difference between vectors $\mathbf{u}(\mathbf{p})$ at $\mathbf{p}$ and $\mathbf{u}(\mathbf{p}')$ at a nearby point $\mathbf{p}' \equiv \mathbf{p} + \epsilon\mathbf{w}(\mathbf{p})$, where $\epsilon \in \mathbb{R}$ is approaching 0. However, these vectors live in different tangent spaces, so the component-wise differences depend on the choice of local basis frames, and taking their differences in a manner that is purely geometric (i.e.,coordinate frame-independent) requires the additional information of connection.

2.2.4 Metric-preserving covariant derivative

Given a frame field and a connection 1-form, the covariant derivative $\nabla\mathbf{u}$ of vector field $\mathbf{u} = u^1\mathbf{e}_1 + u^2\mathbf{e}_2$ can be expressed in coordinates as

$$\nabla\mathbf{u} = (\mathbf{e}_1, \mathbf{e}_2) \left[\begin{pmatrix} du^1 \\ du^2 \end{pmatrix} + \omega\, J \begin{pmatrix} u^1 \\ u^2 \end{pmatrix} \right], \tag{2.2}$$

where d is the exterior derivative [72] applied to the components of the vector field, and the connection is used to locally account for the change of frames. While it is now a stand-alone operator

on vector fields akin to the gradient operator for functions, it can also be paired with another vector field $\mathbf{w}$ to measure the directional derivative of $\mathbf{u}$ along $\mathbf{w}$, i.e.,

$$\nabla_{\mathbf{w}}\mathbf{u} = (\mathbf{e}_1, \mathbf{e}_2)\left[\begin{pmatrix} du^1(\mathbf{w}) \\ du^2(\mathbf{w}) \end{pmatrix} + \omega(\mathbf{w})\ J \begin{pmatrix} u^1 \\ u^2 \end{pmatrix}\right].$$

Note that even if ω is dependent on the choice of the local frame field, $\nabla\mathbf{u}$ is a proper tensor field defined on the tangent bundle.

Geometric decomposition In an orthonormal frame field, the covariant derivative of the vector field $\mathbf{u}$ can be expressed in a matrix form. For clarity, we omit the basis $(\mathbf{e}_1, \mathbf{e}_2)$ appearing in the expression of $\nabla\mathbf{u}$ in Eq. 2.2 in what follows. Representing the reflection matrix by

$$F = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

and the 2×2 identity matrix by I, this matrix representation can be rearranged into four geometrically relevant terms:

$$\nabla\mathbf{u} = \frac{1}{2}[I\nabla\cdot\mathbf{u} + J\nabla\times\mathbf{u} + F\nabla\cdot(F\mathbf{u}) + JF\nabla\times(F\mathbf{u})], \tag{2.3}$$

where $J\nabla\times\mathbf{u}$ (measuring the curl of $\mathbf{u}$) is the only antisymmetric term. Moreover, we can rewrite this decomposition as a function of two other relevant derivatives:

$$\nabla\mathbf{u} = \partial\mathbf{u} + F\bar{\partial}\mathbf{u},$$

where the holomorphic derivative $\partial \equiv \frac{1}{2}(I\nabla\cdot + J\nabla\times)$ contains divergence and curl of the vector field, neither of which depend on the choice of local frame; whereas the Cauchy-Riemann operator (or complex conjugate derivative) $\bar{\partial} \equiv \frac{1}{2}(I\nabla\cdot F + J\nabla\times F)$ depends on the choice of frame. Due to the use of reflection, $\bar{\partial}\mathbf{u}$ behaves as a 2-vector (2-RoSy) field (see Appendix A).

Parallel transport Parallel transport of a vector field $\mathbf{u}$ along a path $C(s)$ (where s is a parameterization of the path) between two nearby points of a manifold $\mathcal{M}$ in a given metric connection

is defined by $\nabla_{C'(s)}\mathbf{u} = 0$, where $C'(s)$ denotes the unit tangent to the path $C(s)$. This implies by integration that a vector, represented in the basis $(\mathbf{e}, \mathbf{e}^{\perp})$, evolves infinitesimally along the path via the connection as

$$\mathbf{u}(s) = \exp\left(-J\int_0^s \omega(C'(\alpha))d\alpha\right)\mathbf{u}(0). \tag{2.4}$$

Consequently, a vector parallel-transported along this path undergoes a series of infinitesimal rotations, adding up to a finite rotation $-\rho = -\int_C \omega$, since $\exp(\theta J) = \cos\theta I + \sin\theta J$. This matrix exponential thus induces a linear mapping (namely, a rotation because we restrict our discussion to metric connections) between the tangent spaces of the two points.

2.2.5 Relevant energies

Based on the decomposition of the covariant derivative operator in Eq. 2.3, we can also express the Dirichlet energy E_{D} of vector field as the sum of two meaningful energies:

$$E_{\mathrm{D}}(\mathbf{u}) = \frac{1}{2}\int_S |\nabla\mathbf{u}|^2 dA = \frac{1}{2}\left(E_{\mathrm{A}}(\mathbf{u}) + E_{\mathrm{H}}(\mathbf{u})\right).$$

The antiholomorphic energy E_{A} measures how much the vector field deviates from being harmonic, and the holomorphic energy E_{H} measures how much the field deviates from satisfying the Cauchy-Riemann equations:

$$\begin{cases} E_{\mathrm{A}}(\mathbf{u}) = \frac{1}{2}\int_M [(\nabla\cdot\mathbf{u})^2 + (\nabla\times\mathbf{u})^2] dA, \\ E_{\mathrm{H}}(\mathbf{u}) = \int_M (\bar{\partial}\mathbf{u})^2 dA. \end{cases}$$

While complex numbers are often used to express these energies, we stick to basic vector calculus in our work for simplicity.

Furthermore, the difference between E_{H} and E_{A} leads to a boundary term and a term related to the connection curvature $K = -d\omega$,

$$E_{\mathrm{A}}(\mathbf{u}) - E_{\mathrm{H}}(\mathbf{u}) = \int_{\partial M} \mathbf{u}\times(\nabla\mathbf{u}) + \int_M K|\mathbf{u}|^2. \tag{2.5}$$

In Chapter 4, all the operators and energies presented will be given a discrete formulation for their evaluation on triangle meshes.

2.3 Decomposition of Tensor Field on Smooth Manifolds

We begin with a brief review of existing decompositions of arbitrary rank-2 tensors on smooth surfaces $\mathcal{M}$ with boundaries $\partial\mathcal{M}$. Note that we will restrict our exposition to tensors of type $(0,2)$ (i.e., acting on tangent vectors), but equivalent expressions for tensors of type $(1,1)$ or $(2,0)$ can be derived using proper raising or lowering of indices with the flat and sharp operators defined by the Riemannian metric $\mathfrak{g}$.

2.3.1 Antisymmetric vs. symmetric tensors

Just as a matrix A can be decomposed into a symmetric $\frac{1}{2}(A+A^t)$ and an antisymmetric $\frac{1}{2}(A-A^t)$ part, a rank-2 tensor field $\tau \in \mathcal{T}$ can be decomposed into an antisymmetric (or skew-symmetric) tensor $\mu \in \mathcal{A}$ and a symmetric tensor $\sigma \in \mathcal{S}$ with $\tau = \mu + \sigma$. Therefore,

$$\mathcal{T} = \mathcal{A} \oplus \mathcal{S}. \tag{2.6}$$

This decomposition is trivially an orthogonal direct sum for the Frobenius inner product $\langle ., . \rangle_F$ due to the fact that the product of an antisymmetric matrix and a symmetric matrix is traceless, and thus their inner product vanishes. Note that antisymmetric tensors are also called "forms", and have been extensively used as the basis of exterior calculus [72]. Common geometric notions such as metric, stress, and strain are, instead, symmetric tensors.

2.3.2 Decomposition of antisymmetric 2-tensors

Antisymmetric rank-2 tensors $\mu \in \mathcal{A}$, dubbed 2-forms, are particularly simple on surfaces: they are of the form $\mu = s\,\mu_{\mathfrak{g}}$ where s is an arbitrary scalar function. They can be further decomposed, via Hodge decomposition [72], as the orthogonal direct sum $\mu = d\omega \oplus h$, where ω is an arbitrary 1-form and h is a harmonic 2-form—which is simply a constant p times $\mu_{\mathfrak{g}}$ if $\mathcal{M}$ has a single connected component. Consequently, by applying the Hodge decomposition on ω, we see that 2-forms can be written in full generality as:

$$\mu = s\,\mu_{\mathfrak{g}} = (\Delta q + p)\,\mu_{\mathfrak{g}}, \tag{2.7}$$

where Δq is the Laplacian of a scalar function q, and p is a scalar constant (non-zero constant functions are, indeed, not in the image of the Laplacian operator).

2.3.3 Decomposition of symmetric 2-tensors

Symmetric rank-2 tensors can also be decomposed further. Berger and Ebin [73] were the first to propose a notion of decomposition of symmetric tensors on *arbitrary* manifolds that extends the well-known Hodge decomposition of vector fields and forms. Noticing the role of the kernel and image of divergence and curl in the Hodge decomposition, they proposed to orthogonally decompose a symmetric tensor via the image of an operator P (with injective principal symbol) and the kernel of its adjoint operator P^* (uniquely defined via $\langle P\cdot,\cdot\rangle_F = \langle \cdot, P^*\cdot\rangle$ up to boundary conditions):

$$\mathcal{S} = \operatorname{Im} P \oplus \operatorname{Ker} P^*. \tag{2.8}$$

This is the generalization of the well-known fact that, for any given matrix, its range and the kernel of its transpose form an orthogonal decomposition of the entire space. We review relevant examples of this versatile construction next.

Divergence-based expression. One of the most common differential operators on manifolds is the covariant derivative [74], which extends the notion of directional derivative for arbitrary manifolds. The covariant derivative $\nabla\omega$ of a 1-form ω returns a rank-2 tensor whose symmetric part is the Killing operator of ω, i.e., $\frac{1}{2}\left(\nabla\omega+\nabla\omega^t\right) \coloneqq \mathcal{K}(\omega)$.[i] The Killing operator is, itself, remarkably relevant in differential geometry: its kernel corresponds to vector fields (known as Killing vector fields) that define isometric flows on the surface [20]. For $P \equiv \mathcal{K}$ in Eq. (2.8), the adjoint operator P^* turns out to be the negated divergence operator $\mathrm{d}iv$ on tensors [73], implicitly defined on a closed surface as:

$$\langle \sigma, \mathcal{K}(\omega)\rangle_F = -\langle \mathrm{d}iv\ \sigma, \omega\rangle_1 \quad \forall \sigma \in \mathcal{S}. \tag{2.9}$$

[i] Note that our definition of the Killing operator differs by a factor $\frac{1}{2}$ from most authors, in an effort to simplify further expressions.

Note that, for flat domains with the Euclidean metric I, the Killing operator can be expressed in local coordinates as a symmetric matrix with entry (i, j) of the form $\frac{1}{2}(\partial_j + \partial_i)$, while the divergence operator reduces to the divergence of each column of the matrix form of a tensor. From the Berger-Ebin decomposition, we conclude that any symmetric tensor field is composed of a divergence-free part plus an element of the image of the Killing operator:

$$\mathcal{S} = \operatorname{Im} \mathcal{K} \oplus \operatorname{Ker} \mathrm{d}iv\,. \tag{2.10}$$

Curl-based expression. We can also define a similar decomposition using this time the notion of curl of a tensor. In fact, the relationship between $\mathrm{d}iv$ and $curl$ for 2-tensors in $\mathcal{M}$ is simple: just like the curl of a vector field is minus the divergence of its rotated version, for a 2-tensor σ we have $curl\ \sigma := \mathrm{d}iv\ (\star\sigma\star^{-t})$, where $\star\sigma\star^{-t}$ is the $\star$-conjugate of σ. We thus get a Berger-Ebin decomposition

$$\mathcal{S} = \operatorname{Im} \overline{\mathcal{K}} \oplus \operatorname{Ker}\ curl\ , \tag{2.11}$$

where the operator $\overline{\mathcal{K}}$ indicates the $\star$-conjugate of the Killing operator, i.e., $\overline{\mathcal{K}}(\omega) := \star\mathcal{K}(\omega)\star^{-t}$.

Trace-based expression. Another canonical operator on tensors is the trace tr. Its Berger-Ebin based decomposition is rather trivial since the adjoint of the trace is simply $\mathrm{tr}^*(z) = z\mathfrak{g}$ for any scalar function z, thus leading to:

$$\mathcal{S} = \operatorname{Im} \mathrm{tr}^* \oplus \operatorname{Ker} \mathrm{tr}\,. \tag{2.12}$$

2.3.4 Remarks

We conclude this section with a few observations.

Boundary conditions. In order to uniquely define the adjoint relation in Eq. (2.9), we must prescribe boundary conditions for 2-tensor fields. Similar to the case of 1-forms, this can be achieved by either prescribing boundary tensors (Dirichlet boundary condition) or specifying their normal derivatives (Neumann boundary condition).

Physical Interpretation. Tensor decompositions are often used to characterize deformations in mechanical systems. The antisymmetric part of asymmetric tensors (Eq. (2.6)), for instance, reveals infinitesimal rotations in a fluid motion. Divergence-free tensors and the Killing operator (Eq. (2.10)) indicate force equilibrium and deviation from isometries in solid mechanics. Similarly, the trace-based decomposition (Eq. (2.12)) identifies local dilations and shearing, commonly controlled in elasticity via Lamé parameters.

Covariant Derivative. As mentioned in §2.3.3, the covariant derivative maps a 1-form ω to a 2-tensor field $\nabla\omega$ which is the sum of a symmetric and an antisymmetric part:

$$\nabla\omega = \mathcal{K}(\omega) - \tfrac{1}{2}d\omega, \tag{2.13}$$

where the antisymmetric part is half the curl of the vector field $\omega^\sharp$ associated to ω. Therefore, the covariant derivative $\nabla\omega$ identifies the scalar function q in Eq. (2.7) with the (negated) function g induced by the coclosed part $\star dg$ of ω.

Generalized Laplacian. The standard Laplace-Beltrami operator Δ on a function z is defined as $\mathrm{d}iv\ (\nabla z)$. This operation generalizes for a symmetric 2-tensor field σ in $\mathcal{M}$ as

$$\Delta^\sigma z \coloneqq \mathrm{d}iv\ (\sigma\nabla z) = \star\left(\mathrm{d}iv\ \sigma \wedge \star dz\right) + \mathrm{tr}\Big(\sigma\, \mathfrak{g}^{-1}\, \mathrm{Hess}(z)\Big)\,, \tag{2.14}$$

where $\mathrm{Hess}(z)$ is the Hessian of z. This operator is particularly relevant in the computation of quasi-harmonic fields [68] and in elasticity [75]. Graphics applications have also used this generalized Laplacian to compute anisotropic parameterization [63, 76] and filtering [77], and more recently to design simplicial masonry structures [78, 79]. Note that, when σ is a divergence-free tensor field, the generalized Laplace operator becomes *linear accurate*, i.e., $\Delta^\sigma z = 0$ for any linear function z in the plane.

2.4 Discrete Exterior Calculus

A computational tool used throughout this dissertation is the discrete implementation of the aforementioned exterior calculus, which preserves crucial differential identities in vector field analysis. As the p differential forms are integrands on p dimensional shapes in the domain, they admit natural discretization, through their integral values on p dimensional cells in tessellated domains.

In particular, for surface mesh, a *primal* $0/1/2$-form is stored as one value integrated per vertex/edge/face, while a *dual* $0/1/2$-form is stored as one value integrated per face/edge/vertex. The usual differential operators in vector field analysis can be implemented through two basic operators d and $\star$ in DEC. The first operator *differential*, or exterior derivative, d_k maps k-forms to $k{+}1$-forms by evaluating the sum of k-form integrals on the boundary of $k{+}1$-cells, and the second operator *Hodge star* $*_k$ maps primal k-forms to dual $2{-}k$-forms in 2D by rescaling them using the size ratio between k-cells and dual $2{-}k$-cells. We may interpret d_0 as gradient ∇, d_1 as curl $\nabla\times$, $*_0$ as multiplication by the area form, $*_1$ as rotation by 90° on the tangent plane, and $*_2$ as division by the area form. Other differential operators can be assembled from d and $*$, e.g. *co-differential* $\delta_k = *_{k-1}^{-1} d_{k-1}^T *_k$, which are adjoint to differentials. In particular, divergence $\nabla\cdot$ can be implemented by δ_1. On a triangle mesh, the d's are implemented transposes of the signed incidence matrices, and the $*$'s are implemented as diagonal matrices with the ratios between the sizes of dual mesh cells and the corresponding primal mesh cells on the diagonal. For more details, refer to [4].

CHAPTER 3

ORIENTATION FIELD GUIDED TEXTURE SYNTHESIS

3.1 Introduction

Texture synthesis is a popular method to acquire textures. Textures often contain salient feature directions, whose alignment is often controlled by a user-specified guidance direction field. Such a direction field is also mandatory for most methods on surfaces even when there are no features.

For textures with two way rotational symmetry (2-RoSy), the guidance fields do not have be continuous everywhere. Instead, nearby vectors should be allowed to have nearly opposite directions to have natural singularities. Such fields are called 2-RoSy fields, or orientation fields. For instance, fingerprints are oblivious to whether the direction is forward or backward along the ridges. The principal curvature direction fields on surfaces is another extremely important example of 2-RoSy. In this chapter, we specifically target at developing a method for handling such fields.

In the following, we first briefly discuss the most relevant related work on orientation field design and on texture synthesis. We then present our vector field design algorithm for the natural/free boundary condition in Sec. 3.3. The singularity control in orientation fields is presented in Sec. 3.4. Next, we elaborate on the nontrivial modifications to make parallel appearance texture synthesis applicable to orientation fields in Sec. 3.5. Examples demonstrating the capability of our system are shown in Sec. 3.6. We conclude this chapter in Sec. 3.7.

3.2 Related Work

Vector field design In a flexible and intuitive texturing system, users should be able to control the orientation and sizing of textures on surface. Such controls are often achieved by designing a vector field prescribing one of the axes of the local coordinate frames. Some vector field design tools used

interpolation from scattered user-specified directions [80, 81], and others also allow singularity control [13, 17]. Zhang et al. [13] proposed to use geodesic polar maps and parallel transport to create radial basis functions given users' requirements. Fisher et al. [17] employed the tools from discrete exterior calculus, by representing the field as discrete 1-forms and solving linear equations with user-defined constraints. Our vector field design is based on [17], with one main difference on the treatment of the free boundary condition. The change is necessary as the efficient tools from discrete exterior calculus cannot express the vector field Dirichlet energy of vector fields as a direct combination of the basic operators in exterior calculus for surfaces with boundaries. More precisely, in this case, the Dirichlet energy of a tangent vector field is different from the sum of the squared sum of the L_2-norms of its divergence field and its curl field, as detailed in Sec 3.3.

Texture synthesis The literature on generating large texture patches automatically from given exemplars is vast, as such example-based texture synthesis techniques, among the state-of–the-art texture acquisition methods, are easy to use and capable of producing results without unnatural artifacts or periodicity. Most of the example-based methods are based on the Markov Random Field theory, assuming that the combined probability distribution of pixels has stationarity and locality. The actual implementation can be pixel-based, patch-based, or more generally, optimization-based. Patch-based algorithms [82, 83, 84] extract consistent patches from the exemplar and glue them together to create large textures. They can be highly efficient with neighborhoods faithful to those in the exemplar. However, they do not provide large variation and can be inefficient for runtime synthesis due to the sequential nature of the process. Some pixel-based algorithms, on the other hand, are able to generate high quality results interactively through multi-scale Gaussian image stacks and parallel texture synthesis [85, 2]. Extensions to perform texture synthesis on surfaces by forming seamless texture across atlas charts can be found in, e.g., [86, 2]. Refer to the surveys [87, 88] for more information on texture synthesis.

Relation to our work For our task of orientation field guided texture synthesis, we use a modified version of the tangent field design method [17]. The vector field design through a weighted least squares method leads to a straightforward Poisson-equation-like linear system. When the singularities are moved, there is no need to rebuild spanning trees as in [26], or rerun discrete Ricci flows [25]. For index-1 singularities (poles), only the right hand side of the linear system is modified; for index-−1 singularities (saddles), an efficient increment to the Cholesky factorization of the left hand side can be performed. Furthermore, while texture synthesis depends only on the unit direction of the field in most cases, the true vector field design in engineering applications could benefit from a method that allows direct control over divergence and curl as in [17]. Our texture synthesis stage is based on [85] and [2], which manipulate pixel coordinates to overcome the issue of lack of efficiency in order-independent neighborhood matching. However, for orientation field guided synthesis, the upsampling and correction steps in the top-down multiscale approach must be substantially modified.

3.3 Vector Field Design with Natural Boundary Conditions

Before presenting our modification to the natural boundary condition, we briefly recap the method described in [17]. We demonstrate the necessity of our modification through examples.

The tangent vector field design problem is formulated as a weighted least squares problem in [17]. The design constraints are specified through user-controlled curl and divergence of the vector field, as well as direct constraints on the vectors, all at scattered locations. The curl is only non-zero at user-specified vortices, and the divergence is only non-zero at user-specified sources or sinks. The direct constraints can be any prescription of the vector at selected locations, but are often specified in batches through user sketch strokes.

When the relevant fields are expressed as discrete differential forms, the above weighted least squares problem has a straightforward implementation through discrete exterior calculus (DEC). The resulting linear system is essentially a Poisson equation combined with terms from soft con-

straints.

3.3.1 Setup

The computation is carried out on a 2-manifold with boundary, represented by a triangle mesh M , with vertex set V, edge set E, and triangle set T. A vector field $\mathbf{u}$ can be stored as a 1-form, i.e. one scalar value per oriented edge $e_i \in E$, denoting the line integral along the edge $c_i = \int_{e_i} \mathbf{u}$. A scalar field s can be stored as a 0-form, one value per vertex $v_i \in V$, $s_i = s(v_i)$, or as a 2-form, one value per triangle $t_j \in T$, $s_j = \int_{t_j} s_j$. In particular, the divergence of a vector field can be represented by a 0-form, while its curl can be represented by a 2-form.

Assume that U represents the vector field, S the divergence field, C the curl field, U_Z the constraints on selected edges, where Z is the matrix projecting an array representing a 1-form onto an array assembled by one value per user-selected edge. The desired vector field can be computed by the weighted least squares solution of the following equations,

$$\delta U = S, \;\; dU = C, \;\; ZU = U_Z,$$

leading to

$$(*(d\delta + \delta d) + Z^T W Z)U = *dS + *\delta C + Z^T W U_Z,$$

where d the *differential*, δ the *co-differential* and $*$ the *Hodge star* operator in DEC. Here we omit the subscripts when they can be determined from the context. Also W specifies the weighting of the direct constraints. Aside from the term induced by Z, the resulting symmetric linear system is simply the vector field Poisson equation, where the Laplace-Beltrami operator $d\delta + \delta d$ is equivalent, up to a sign, to what can be obtained from the vector calculus identity

$$\nabla^2 = \nabla\nabla\cdot - \nabla\times\nabla\times \, .$$

3.3.2 Natural boundary conditions

In [17], the free boundary condition, for the case when the vector field is not restricted to be at a certain angle with the boundary, is implemented by adding a term to properly include the integral of di-

vergence for the partial Voronoi cells at the boundary. However, this still leaves a high-dimensional kernel for the resulting linear operator. Numerically the operator is likely to be positive definite due to the discretization, but it leads to spurious singularities, unless sufficient direct constraints are included or when the much denser bi-Laplacian is included.

Our remedy to the above problem is based on the observation that the free boundary condition should be obtained through minimizing the Dirichlet energy $\int_M |\nabla \mathbf{u}|^2$. Choosing a local orthonormal frame $\{\mathbf{e}_1, \mathbf{e}_2\}$ at each point, we denote the partial derivatives of the components of $\mathbf{u}$ by $u_{\alpha,\beta} = \frac{\partial u_\alpha}{\partial x_\beta}$. Assuming trivial connection, we ignore the curvature-related term (see the Weitzenböck formula in, e.g., [74]), and focus on the influence of the boundary

$$\begin{aligned}\int_M |\nabla \mathbf{u}|^2 &= \int_M u_{1,1}^2 + u_{1,2}^2 + u_{2,1}^2 + u_{2,2}^2 \\ &= \int_M (u_{1,1} + u_{2,2})^2 + (u_{2,1} - u_{1,2})^2 - 2(u_{1,1}u_{2,2} - u_{1,2}u_{2,1}) \\ &= \int_M (\nabla \cdot \mathbf{u})^2 + (\nabla \times \mathbf{u})^2 - \int_{\partial M} u_1 du_2 - u_2 du_1 \end{aligned} \tag{3.1}$$

The boundary term in the last row is a result of Stokes' theorem. In standard calculus, we may rewrite the term as

$$-\int_{\partial M} (\mathbf{u} \times d\mathbf{u}) \cdot \mathbf{n},$$

where $\mathbf{n}$ is the surface normal.

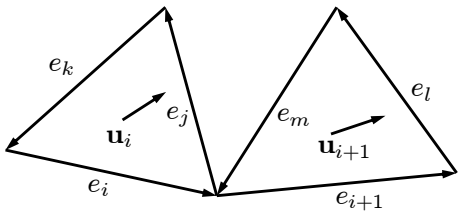


Figure 3.1 **Two consecutive edges along the boundary.**

In the discrete setting, we express the Dirichlet energy as $U^T L U$, where U is the discrete 1-form representation of $\mathbf{u}$, and L is the Laplacian-like matrix to be constructed. We initialize L to the sum

of the divergence and curl terms, and then add the boundary term. To discretize the boundary term, we first turn the boundary integral into a summation over boundary edges

$$\sum_{e_i \in \partial M} (\mathbf{u}_i \times (\mathbf{u}_{i+1} - \mathbf{u}_i)) \cdot \mathbf{n}_i = \sum_{e_i \in \partial M} (\mathbf{u}_i \times \mathbf{u}_{i+1}) \cdot \mathbf{n}_i,$$

where we assume that e_{i+1} is the edge following e_i along the boundary, and $\mathbf{n}_i$ is the surface normal at their shared vertex.

Assuming the discrete curl for boundary triangles to be close to zero as in [17], we have a constant vector within each triangle, which allows us to simply choose any point (in particular, the barycenter) of the triangle for the evaluation of $\mathbf{u}_i$. A discrete 1-form is one value per edge, so $U = (c_1, c_2, \ldots, c_n)$, where n is number of edges. For the pair of boundary triangles shown in Figure 3.1, we have

$$\mathbf{u}_i = c_i \boldsymbol{\phi}_i + c_j \boldsymbol{\phi}_j + c_k \boldsymbol{\phi}_k,$$

$$\mathbf{u}_{i+1} = c_{i+1} \boldsymbol{\phi}_{i+1} + c_l \boldsymbol{\phi}_l + c_m \boldsymbol{\phi}_m,$$

where $\boldsymbol{\phi}_i = \frac{1}{3}(\nabla \phi_{v_2} - \nabla \phi_{v_1})$ is the basis function for edge i pointing from v_1 to v_2 evaluated at the barycenter of the corresponding triangle, and ϕ_v is the linear basis function for vertex v. The update to L involves 18 terms in 9 pairs, e.g.

$$L_{jl} += (\boldsymbol{\phi}_j \times \boldsymbol{\phi}_l) \cdot \mathbf{n}_i, \; L_{lj} += (\boldsymbol{\phi}_j \times \boldsymbol{\phi}_l) \cdot \mathbf{n}_i.$$

On a curved patch, we may obtain the surface normal at the vertex from any reasonable weighted averaging.

As shown in Figure 3.2, any harmonic vector field can be added to a field without changing the target function in [17], leading to spurious singularities, while in our case, the Dirichlet energy minimization produces the expected results.

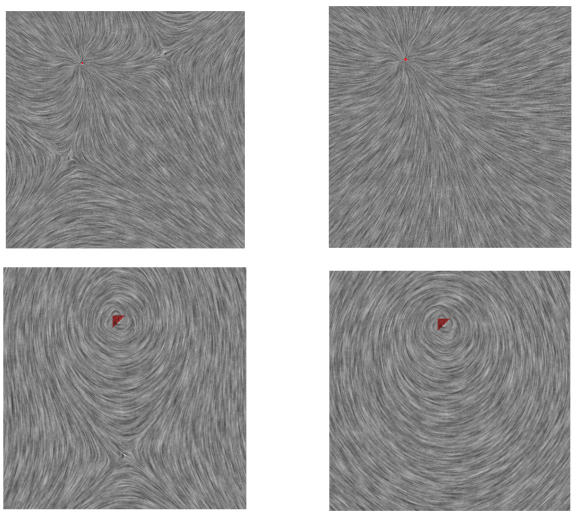


Figure 3.2 **Comparison of results.** User input: a single source (top); a single vortex (bottom). Fisher et al.'s design method produced multiple spurious singularities (left); our method produced the minimizer of the Dirichlet energy (right).

3.4 Orientation Field Design

Representation of the direction field We follow the practice in [23], and represent the direction field by the angle θ denoting the deviation from the x-axis of a local coordinate frame. We can construct the frame field by specifying the x-axis through the solution of the above natural boundary condition vector field design obtained by fixing a single vector, in the case of trivial topology. Otherwise, it can be computed by first specifying some singularities consistent with the Poincaré-Hopf index theorem, and use trivial connection [26] or discrete Ricci flow [25] to construct the frame field.

The orientation field can then be represented by a smooth vector field with the angle 2θ, since $2(\theta+\pi)$ leads to the same angle. Using a complex number to represent the vector in the local frame, we can use the square and square root operations for complex numbers to convert the orientation field and vector field from each other.

The major features in an orientation field are determined by the singularities. Two basic singularities, wedge and trisector (Figure 3.3) with index-$\frac{1}{2}$ and index-$-\frac{1}{2}$, respectively, can be used to produce singularities of arbitrary indices in the orientation field.

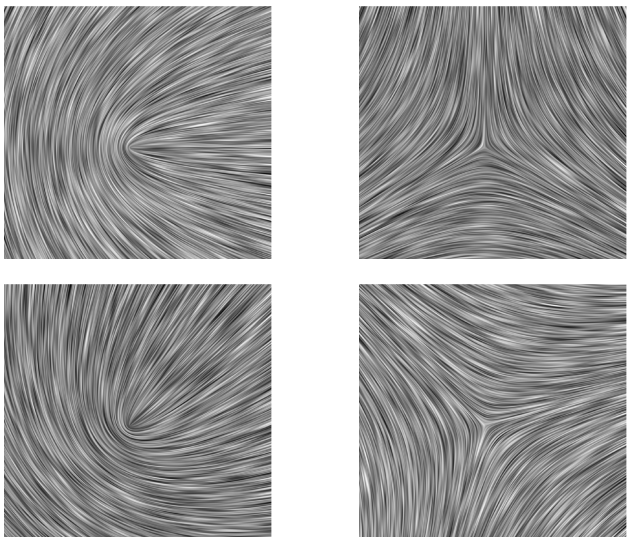


Figure 3.3 **Basic singularities for orientation fields: wedge (left) and trisector (right).** Our system also provides control over the orientations. Top: original; Bottom: 45° rotated.

We present a simple method of specifying not only the singularity types and locations, but also their orientations. Wedges correspond to sources/sinks and vortices, for example, a source correspond to wedge with a horizontal flow line connected to the singularity. As described in [13], we can see that the local field in a small neighborhood around a source vertex v have the form of $e^{i\theta}$, where θ is the angle between the displacement from v to the point in the neighborhood and the local x-axis direction. Thus, the corresponding orientation field is of the form $e^{i\theta/2}$, the expected wedge. For a vortex, the vector field is of the form $e^{i(\theta+\pi/2)}$. Combining the two with weights $\cos 2\alpha$ and $\sin 2\alpha$, we have the orientation field of the form $e^{i(\theta/2+\alpha)}$, i.e. a rotated wedge. The trisectors correspond to saddle points, which can be constructed by controlling the one-ring of a vertex v to have a vector of $e^{-i(\theta+2\alpha)}$ in each incident triangle. See the 45° rotated wedge and trisector in the bottom row of Figure 3.3.

With a proper combination of singularities of positive and negative indices, there would rarely be any uncontrolled singularities with our proper boundary condition. This behavior can be understood by following the same argument as in [26, 25]: additional vector field constraints can be seen as just smooth deviation from an initial orientation field satisfying the singularity constraints.

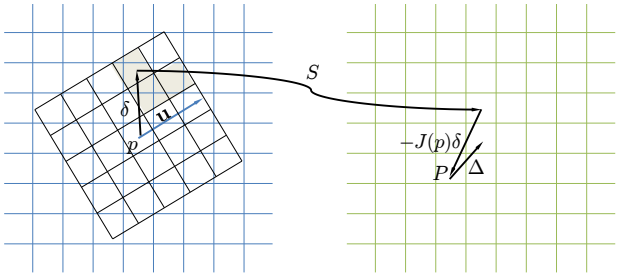


Figure 3.4 **One of the three predicted texture locations for the upper-right corner in the four-corner neighborhood.**

3.5 Texture Synthesis for 2-RoSy Field

Traditional controllable texture synthesis often uses a designed smooth vector field as user input. This is not the same as using an orientation field, since the field of representative vectors chosen from one of the two direction is inevitably discontinuous in the presence of at least one wedge or trisector. Thus as we adapt the strategy from [85, 2], and perform the coarse-to-fine texture synthesis, we must introduce an "upside-down" mapping style to enforce the continuous appearance. We restrict our discussion to the planar case, as the curved patch case is treated by combining the Jacobian of the parameterization as in [2]

3.5.1 Anisotropic texture synthesis

We first briefly summarize the appearance space synthesis in [2] before discussing the modifications. In the preparation stage, a Principal Component Analysis (PCA) is performed on the set of all 5×5-neighborhoods in the exemplar Gaussian stack at each level, to create a 8D appearance vector space, turning the exemplar into a 2D array of appearance vectors $\tilde{E}$. The parallel synthesis then repeats three main steps, namely *upsampling*, *jittering*, and *correction*, until the finest level texture is generated.

The appearance vector at each output pixel is represented through a mapping to a point in the exemplar $\tilde{E}$. Denoting the texture exemplar coordinates for a pixel p in the output pyramid at level L by $S_L[p]$, the upsampling pass for the level-L+1 pixels corresponding to level-L pixel p is rather

straightforward, when there is a guidance field:

$$S_{L+1}[2p + \tilde{\triangle}] = S_L(p) + J(p)\frac{1}{2}\triangle,$$

where

$$\triangle \in \{\begin{pmatrix} -1 \\ -1 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \end{pmatrix}\}, \ \tilde{\triangle} = \frac{1}{2}(\triangle + \begin{pmatrix} 1 \\ 1 \end{pmatrix}),$$

and $J(p)$ is the Jacobian matrix for local S to follow the guidance field, which is essentially a rotation matrix aligning x-axis to the given guidance direction combined with a possible scaling.

In the correction step, a four-corner neighborhood $N_S(p; \triangle)$ is sufficient due to the use of appearance vectors. For better convergence in the parallel synthesis, [2] suggested to average the appearance vector for each corner $\triangle$ from the corner values predicted from three offset locations $\delta(\triangle, M) = \hat{\varphi}(\triangle) + \hat{\varphi}(M\triangle), M \in \mathcal{M}$, where

$$\mathcal{M} = \{\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}\},$$

and $\hat{\varphi}(\triangle) = |\varphi(\triangle)/\|\varphi(\triangle)\| + 1/2|$ is the normalized version of warped offset $\varphi(\triangle) = J^{-1}(p)\triangle$. As the predicted texture location is $P(p, \delta) = S[p + \delta] - J(p)\delta$, shown in Figure 3.4, the final formula is

$$N_S(p; \triangle) = \frac{1}{3}\sum_{M \in \mathcal{M}} \tilde{E}[P(p, \delta(\triangle, M)) + \triangle].$$

The neighborhood is then compared with the precomputed neighborhoods in the exemplar. For fast comparison on GPU, the neighborhood can be further compressed through another PCA.

For efficiency, the search of best-matching exemplar pixel is limited to the k-coherent set

$$\mathcal{C}(p) = \{C(p, \triangle, i) | i = 1 \ldots k, \|\triangle\| < 2\},$$

where the candidates are predicted from nearby points $p + \triangle$ with the precomputed k-coherent offset C'_i,

$$C(p, \triangle, i) = S[p + \triangle] + C'_i(S[p + \triangle]) - J(p)\triangle.$$

We follow their practice of choosing $k = 2$.

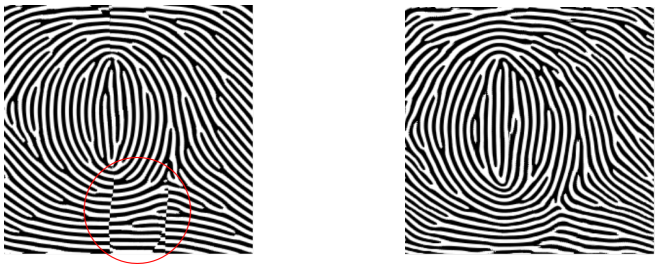


Figure 3.5 **Comparison of the results from the parallel anisometric texture synthesis method without (left) and with (right) our modifications.** The representative vector field has discontinuity within the red circle.

3.5.2 Handling orientation

When the anisotropic parallel synthesis is applied to orientation fields, there are visible artifacts when the representative vectors are changing to the opposite directions (Figure 3.5). Increasing the amount of jittering or rearranging the texture exemplar in a more symmetric way would not solve the problem. Cutting the output into charts according the field and use indirection map would not be less costly and less effective than our solution.

Our modification is based on the observation that the discontinuity in the texture is mainly due to the incorrectly predicted texture coordinates $P(p, \delta)$ as shown in Figure 3.6, which affects both the neighborhood construction and the candidate sets. This issue can be fixed by modifying the prediction to

$$\hat{P}(p, \delta) = S[p + \delta] - J(p + \delta)\delta.$$

The four-corner neighborhood is also modified to

$$\hat{N}_S(p; \triangle) = \frac{1}{3} \sum_{M \in \mathcal{M}} \tilde{E}[\hat{P}(p, \delta(\triangle, M)) + (-1)^{c(p+\delta,p)} \triangle],$$

where the consistency $c(p+\delta, p)$ is defined as $\mathbf{u}(p+\delta) \cdot \mathbf{u}(p) < 0$, a binary indicator of the presence of a 180° rotation.

If we are using color pixels, this would have sufficed. However the appearance vectors represent 5×5-neighborhoods. So they would be containing the wrong appearance if we use the appearance vector at the flipped predicted location directly. To handle the issue without losing much efficiency,

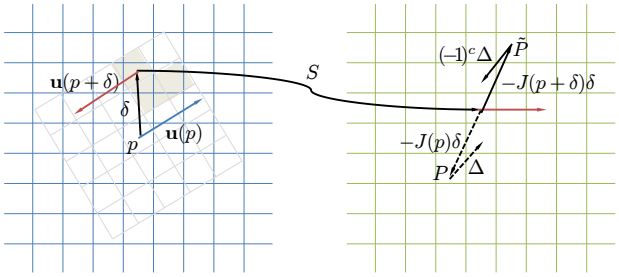


Figure 3.6 **Reason for the disconituity of the synthesized image.** Following the dashed direction would have produced a wrong prediction, while $\tilde{P}$ properly takes into account the mutual orientation.

we build the 8D appearance space from the set containing both the 5×5-neighborhoods and their rotated images. Then we store two appearance images for the exemplar, one for the original $\tilde{E}_0$, the other for the rotated $\tilde{E}_1$. We store one boolean variable $I(p)$ for each output point, indicating whether the rotated appearance is used.

Putting these together, we have the final formula for the neighborhood construction

$$\tilde{N}_s(p;\triangle) = \frac{1}{3}\sum_{M\in\mathcal{M}} \tilde{E}_{\alpha(p,\delta)}[\tilde{P}(p,\delta(\triangle,M)) + (-1)^{\alpha(p,\delta)}\triangle], \tag{3.2}$$

where

$$\alpha(p,\delta) = I(p+\delta) + c(p+\delta,p)$$

combines the effects of the consistency and the current indicator, and

$$\tilde{P}(p,\delta) = S[p+\delta] - (-1)^{\alpha(p,\delta)}J(p)\delta$$

is the modified prediction.

A modification is also in place for the k-coherent candidates

$$C(p,\triangle,i) = S[p+\triangle] + C_i'(S[p+\triangle]) - (-1)^{\alpha(p,\triangle)}J(p)\triangle$$

When comparing the neighborhood information constructed with the candidates, $\tilde{E}_{\alpha(p,\triangle)}(C(p,\triangle,i))$ should be used to account for the possible relative rotations.

When the best match is found at the candidate predicted by offset $\triangle$, the indicator I is updated as well as S,

$$I(p) = \alpha(p,\triangle).$$

Finally, the upsampling step is also adapted to

$$S_{L+1}[2p + \tilde{\triangle}] = S_L(p) + (-1)^{I(p)} J(p) \frac{1}{2} \triangle,$$

and

$$I_{L+1}[2p + \tilde{\triangle}] = I_L(p) + c_{L,L+1}(p, 2p + \tilde{\triangle}),$$

where $c_{L,L+1}(p, q)$ is defined to be $\mathbf{u}_L(p) \cdot \mathbf{u}_{L+1}(q) < 0$, a binary valued function for checking the consistency of the orientation field between different scales, as the coarse and fine levels of the output image may choose different representatives when downsampling from the original orientation fields.

3.6 Results

The tests of our algorithm on examples were performed on a regular laptop with Intel Core2Dual with 4GB memory. In all of our tests, the method took no more than a fraction of a second, allowing for interactive manipulation of the singularity and direction constraints, even though our implementation is not optimized. In theory, we can reach the same efficiency of [17] and [2] in the respective stages, as only negligible overhead is incurred by the modifications to the 1-form-based tangent design method and parallel control texture synthesis.

In Figure 3.7, we show that our system can easily create fingerprint-like images imitating the five main categories of singularity layouts in human fingerprints. We show the method applied to more exemplars for planar regions with orientation fields in Figure 3.8. We use the same procedure in [2] for generating the texture in texture domain while taking into account the Jacobian of the parameterization, and some results are shown in Figure 3.9.

Limitations There is no strict guarantee that additional saddle points would not emerge in our vector field design, if, e.g., we place two sources close to each other. However the same could happen for methods such as [26, 25]: if one specifies some vector direction constraints as in our

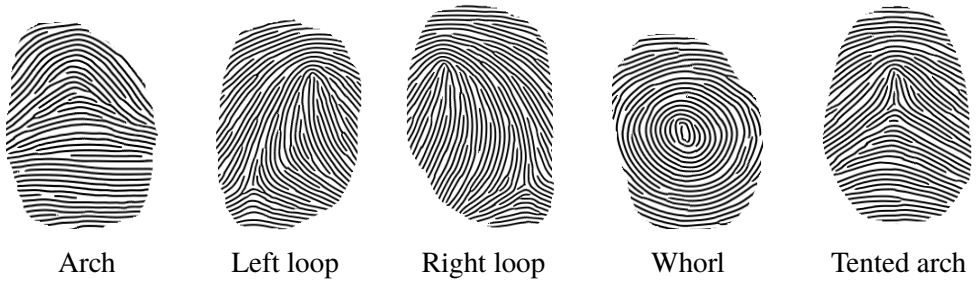


Figure 3.7 **Examples for the five major categories of fingerprints generated by our texture synthesis.**

saddle point placement, extra singularities would have to be generated in addition to those used in constructing the almost everywhere flat metrics or connections. On the other hand, in practice, with a proper mixture of positive and negative singularities, which does not produce excessively large indices in local regions, it would take some strong vector direction constraints to produce additional singularities with any method with proper free boundary condition. Another issue is that our texture synthesis does not provide direct control over the bifurcation and ending of the features contained the exemplar (See, e.g. Figures 3.7 and 3.8), but this is common to many anisometric texture synthesis methods.

3.7 Conclusion

We presented a simple framework based on tangent vector field design. We eliminated the spurious singularities produced by the free boundary condition through including the missing boundary term. Given a local frame field with trivial connection, we covert the vector field into an orientation field by taking the square root of the complex representation of the vector in the local frame, which halves the angle to the x-axis. We also provide control over the orientations of the wedge and trisector singularities. The designed orientation field can then be used in a parallel texture synthesis, adapted for orientation fields. Such texture synthesis allows on-the-fly synthesis and is GPU-friendly, due to its order-independence.

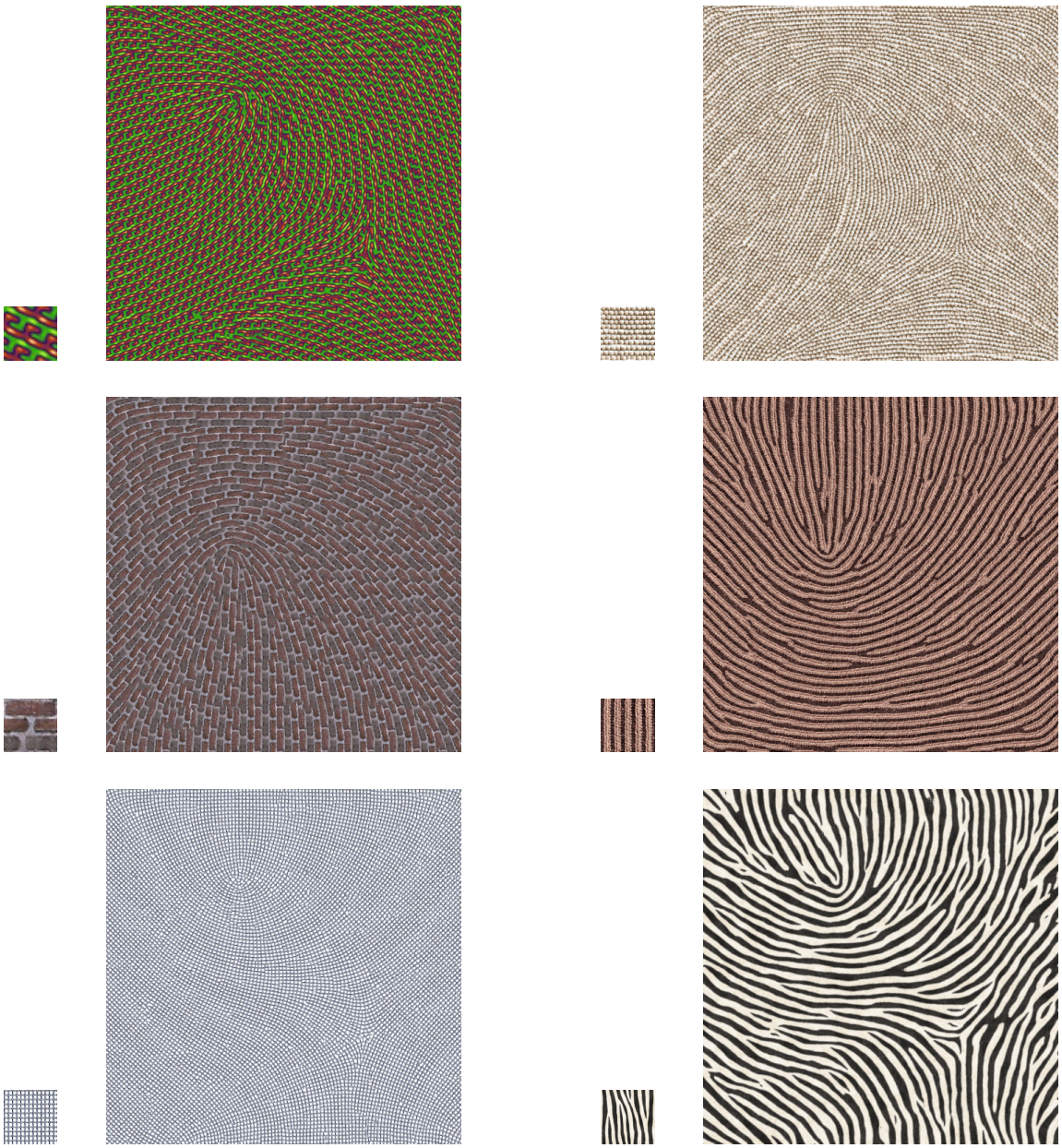


Figure 3.8 **Results with various textures on planar regions.**

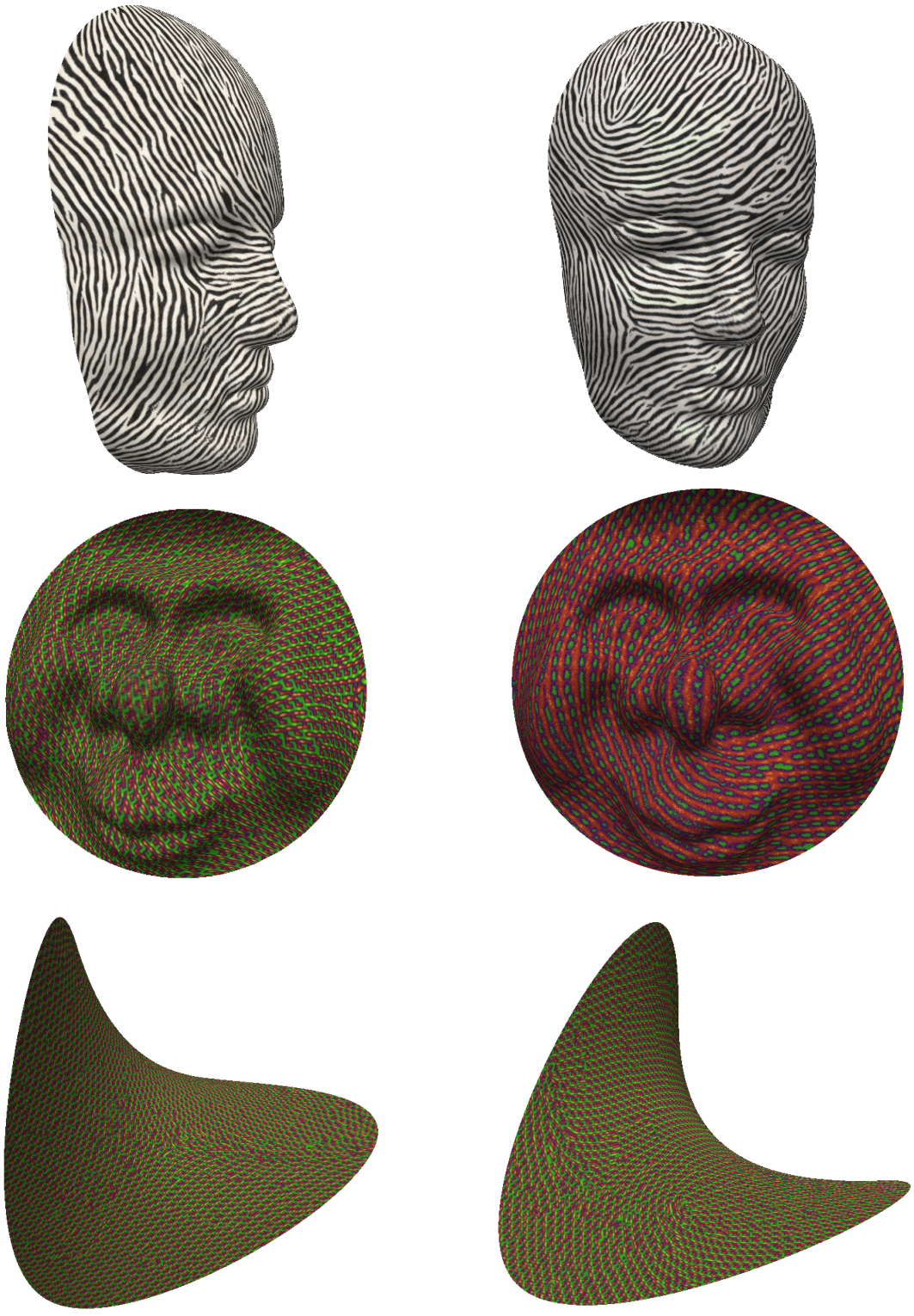


Figure 3.9 **Results for orientation fields on curved patches.**

CHAPTER 4

DISCRETE CONNECTION AND COVARIANT DERIVATIVE FOR VECTOR FIELD ANALYSIS AND DESIGN

4.1 Introduction

Covariant differentiation defines a notion of derivative along tangent vector fields of a curved manifold. Established by Ricci and Levi-Civita, the covariant derivative also relates to the concept of connection (and thus, parallel transport) of vector fields widely used in physics, particularly in gauge theory and relativity. It is also the basic tool to formally measure how a vector field changes over a curved surface, as needed in a wide variety of geometry processing applications ranging from texture synthesis to shape analysis.

Unfortunately, an appropriate discrete counterpart of such a differential operator acting on simplicial manifolds remains elusive. In this chapter, we offer a full discrete treatment of connection through Whitney basis functions that leads to analytical expressions of the covariant derivative for finite-dimensional, vertex-based tangent vector fields on simplicial manifolds. We demonstrate the relevance and novelty of our contributions to vector and n-direction field editing, as it offers control in position *and* orientation of both positive and negative singularities.

4.1.1 Related work

While many graphics applications (spanning texture synthesis and fluid animation) require vector fields on triangle meshes, we only review the key computational ingredients that have been formulated to deal with the analysis and design of vector and n-direction fields over triangulated surfaces.

4.1.1.1 Vector fields

Computational tools for vector fields on discrete surfaces are required whether the user is given a tangent vector field to analyze or (s)he needs to design a vector field from a sparse set of desired constraints. For instance, discrete notions of divergence and curl (vorticity) were formulated [15, 14]; topological analysis also attracted interest, in which positions of vector field singularities are identified, merged, split, or moved [89, 13]; quadratic energies measuring vector field smoothness were also introduced since their minimizers (possibly with added user constraints) limit the appearance of singularities [17].

4.1.1.2 From vector fields to n-direction fields

The more general case of n-direction fields (called unit n rotational symmetry (RoSy) fields in [24]) such as direction fields (n=1) or cross fields (n=4) were numerically handled through energy minimization as well, but the energies that were initially proposed for this case were highly non-linear or involved integer variables [90, 24, 27, 91, 92, 93]. A quadratic energy was recently introduced in [1] through a discretized version of the Dirichlet energy, extending the quadratic energy of [17] which only accounted for the squared sum of the divergence and of the curl of vector fields over the surface. The extra curvature and boundary terms involved in this new approach were also shown to offer additional user control.

4.1.2 Outline and notations

The continuous definitions and relevant properties of connections, covariant derivatives, and associated energies are already discussed in Chapter 2. We propose a discrete definition of connection on a simplicial complex in Sec. 4.2, before discussing in Sec. 4.3 how to compute a globally optimal discrete connection in the sense that it is closest to the Levi-Civita connection of the surface. We then provide in Sec. 4.4 closed-form expressions for basis functions of vector fields and covariant derivatives based on our discrete connections, before explaining in Sec. 4.5 how these numerical

tools can be leveraged to improve (n-)vector and n-direction field editing on triangle meshes. We conclude with visual results of vector field editing and numerical comparisons of our various operators in Sec. 4.6.

Throughout our exposition, we denote by $\mathcal{T}$ a triangulated 2-manifold of arbitrary topology, with a set of vertices $V=\{v_i\}_i$, edges $E=\{e_{ij}\}_{i,j}$ and triangles $T=\{t_{ijk}\}_{i,j,k}$. Each vertex v_i is assigned a position $\mathbf{p}_i$ in $\mathbb{R}^3$. Each edge further carries an arbitrary but fixed orientation, while vertices and triangles always have counterclockwise orientation by convention. Index order indicates direction, in the sense that edge e_{ij} is directed from vertex v_i to v_j. We also exploit the containment relation of a simplicial complex by defining σ to be a *face* of η, and η a *coface* of σ, iff $\sigma \subset \eta$ with $\sigma, \eta \in \mathcal{T}$. We denote the angle in a triangle t_{ijk} between jk and ji by $\theta_{ijk} > 0$. The Gaussian curvature of $\mathcal{T}$ at a vertex v_i is thus expressed as $\kappa_i = 2\pi - \sum_{t_{ijk}} \theta_{kij}$. Finally, we denote by φ_i, φ_{ij}, and φ_{ijk} the Whitney bases of 0-forms on vertices v_i, 1-forms on edges e_{ij}, and 2-forms on triangles t_{ijk} respectively; φ_i is the piecewise linear function with $\varphi_i(v_j) = \delta_{ij}$ (where δ is the Kronecker symbol), while the other form bases are defined as: $\varphi_{ij} = \varphi_i d\varphi_j - \varphi_j d\varphi_i$ and $\varphi_{ijk} = 2\, d\varphi_i \wedge d\varphi_j$ [18].

4.2 Connections on Simplicial Manifolds

We now move to the discrete setting and describe the construction of a discrete connection on simplicial manifolds.

4.2.1 Rationale

Of the seeming inadequacy of triangle meshes The continuous notion of connection does not quite apply as is on a non-smooth manifold. In particular, a triangulated 2-manifold $\mathcal{T}$ has a piecewise linear embedding in 3D Euclidean space with piecewise constant normals, concentrating Gaussian curvature solely at vertices and making local tangent spaces only unequivocally well-defined in the interior of triangles. As a consequence, a vector field's covariant derivative induced by the

Levi-Civita connection of a triangle mesh may not be *pointwise* finite. However, since a pair of triangles can be isometrically flattened, there is a clear way to parallel transport a vector within a pair of adjacent triangles using the Levi-Civita connection induced by the Euclidean metric. A purely discrete notion of arbitrary connections was derived from this idea in [26], using discrete dual connection 1-forms that store adjustment rotations along dual edges, representing an *integral* form of connection. Unfortunately, dual discrete 1-forms have no simple interpolation basis functions other than piecewise constant per triangle to define a proper connection 1-form everywhere on the mesh—thus preventing the evaluation of the L_2-based energy integrals described in Sec. 2.2.5.

Of the importance of the Levi-Civita connection Yet, the conventional notions of divergence or curl of vector fields on smooth surfaces *derive* from the Levi-Civita connection induced by the embedding in $\mathbb{R}^3$ (see Eq. 2.3). We are therefore caught in a dilemma: either we give up on using piecewise linear surfaces and go for higher order surface descriptions for which smoothness is no longer an issue, or we modify, as little as possible, the notion of Levi-Civita connection on the triangle mesh (by "spreading" the Gaussian curvature around vertices) so that one can create smooth vector fields that have continuous covariant derivatives. We opt for the second option in this chapter to retain the simplicity of triangle meshes, while offering a finite-dimensional space of smooth vector fields for which the pointwise derivatives defined in Sec. 2.2.4 and the energies defined in Section 2.2.5 are well defined—and in our case, known in closed form. Note that the piecewise linear nature of simplicial complexes implies that the representation of a connection across simplices (equipped with their own local basis frame) must be discontinuous in order for parallel transport to produce continuous tangent vector fields: we will therefore formulate a generic notion of simplicial connection with finite rotations between adjacent simplices, but a continuous closed-form expression within simplices.

Previous attempts Vertex-based interpolation of vector fields has recently been shown useful to either define local discrete first-order derivatives [13], or evaluate global L_2 norm of deriva-

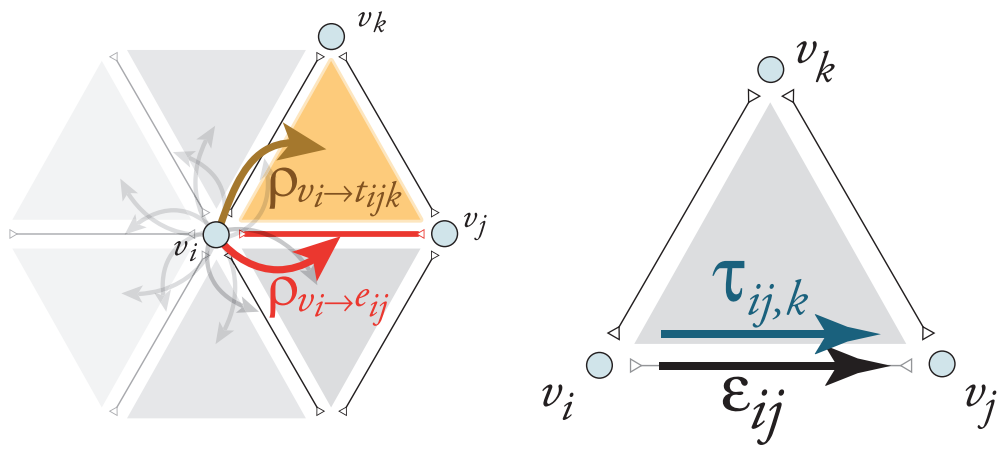


Figure 4.1 **Simplicial connection.** (left) Each vertex v_i is given an impulse rotation angle $\rho_{v_i \to e_{ij}}$ to edge e_{ij} and $\rho_{v_i \to t_{ijk}}$ to triangle t_{ijk}. (right) A continuous connection within simplices is encoded through edge rotation ϵ_{ij} and half-edge rotation $\tau_{ij,k}$ interpolated via Whitney basis functions.

tives [1]—but so far never both. Additionally, vector field interpolation is usually made with specifically-designed basis functions, but the connection implicitly defined by these basis functions is not known analytically. Therefore, no analysis of the resulting connection (in particular, compared to the canonical Levi-Civita connection of the metric induced by the Euclidean embedding of the mesh) has been proposed for this vertex-based vector field setup. Yet, the choice of connection significantly impacts the accuracy of differential operators and energies used ubiquitously in geometry processing since it affects the evaluation of the components of the covariant derivative.

Approach We contribute a definition of discrete connection 1-forms in which we fully exploit the simplicial structure of meshes. By defining a local frame for each simplex of an input mesh, we encode the connection through finite rotation angles between incident simplices and via piecewise linear Whitney 1-forms within simplices. Our construction and its closed-form expressions will allow us to evaluate both local integrations and L_2 norms of derivatives, providing a discrete notion of covariant derivative for vertex-based vector fields over triangle meshes.

4.2.2 Discrete connection 1-form

While the continuous interpretation of parallel transport described in Eq. 2.4 applies directly in regions of the simplicial complex $\mathcal{T}$ that are locally flat (such as along an edge or within a triangle), a special treatment is needed for paths crossing an edge or going through a vertex where tangent space discontinuities happen. We now formulate a generic notion of discrete connection 1-form on manifold triangle meshes through continuous Whitney 1-forms within simplices and impulse (integrated) rotation angles between incident simplices.

Simplicial frames We first arbitrarily choose a frame for each vertex, oriented edge, and triangle of $\mathcal{T}$. Selecting a unit tangent direction $\mathbf{e}_\sigma$ per oriented simplex $\sigma \in \mathcal{T}$ suffices, as a frame can be assembled by picking this reference direction and its $\pi/2$-rotated direction $\mathbf{e}_\sigma^\perp$. This direction can be seen as indicating the x-direction of a local parameterization around the simplex. For an oriented edge, a straightforward choice is along itself; a simple choice of frames for each vertex v_i is the direction of one of the oriented edges emanating from it; and for each triangle it can be one of its (counterclockwise oriented) edges. The choice of frames can be arbitrary as it will not influence the results in any way; however, fixing simplicial frames is a necessary step to both represent and numerically manipulate connections (just like in the continuous case): an assignment of frames $(\mathbf{e}_\sigma, \mathbf{e}_\sigma^\perp)$ on all simplices σ formally defines a section of the frame bundle for each one-ring, over which any tangent vector is expressed by its coordinates (2 scalar values) in this frame field—and over which the connection 1-form ω lives. Note that these frames are purely intrinsic to the surface as in [13, 1], and do *not* define tangent planes in the embedding space.

Whitney-based connections within simplices We can still use the continuous notion of parallel transport within each simplex: smooth paths are well defined in the interior of both edges and triangles. A particularly convenient finite-dimensional representation of a connection 1-form within a simplex is to use discrete 1-forms stored as oriented edge values interpolated via Whitney (piecewise linear) basis functions [18]. We thus represent a continuous connection per (oriented) edge e_{ij}

by providing the (signed) rotation angles ϵ_{ij} experienced while traveling along e_{ij}. Similarly, a continuous connection over (oriented) triangles is defined by providing the (signed) rotation angles $\tau_{ij,k}$ accumulated while traveling inside each triangle t_{ijk} along its half-edge e_{ij} (see Fig. 4.1(right)). Note that $\tau_{ji,k} = -\tau_{ij,k}$ since it denotes the rotation induced along the same half-edge but going from v_j to v_i instead of v_i to v_j. However, $\tau_{ij,l}$ denotes the rotation angles induced along its opposite half-edge, i.e., expressed in triangle t_{ilj}, and, of course, $\tau_{ji,l} = -\tau_{ij,l}$.

Consequently, given $3|F|$ values $\{\tau_{ij,l}\}$ and $|E|$ values $\{\epsilon_{ij}\}$, we can reconstruct the triangle-restricted continuous connection 1-form τ within each triangle t_{ijk} expressed as:

$$\tau = \tau_{ij,k}\varphi_{ij} + \tau_{jk,i}\varphi_{jk} + \tau_{ki,j}\varphi_{ki},$$

where φ_{ij} is the Whitney 1-form basis for e_{ij} (see Sec. 4.1.2); and the edge-restricted continuous connection 1-form ϵ expressed as:

$$\epsilon = \epsilon_{ij} d\varphi_j,$$

where φ_i is the piecewise-linear Whitney basis function for vertex v_i, and $d\varphi_j$ is the restriction of φ_{ij} to the edge e_{ij}. Now parallel transport along any path $C(t)$ within a triangle engenders a discrete rotation angle $\int_{C(t)} \tau$, while parallel transport along any path $C(t)$ within an edge engenders a discrete rotation angle $\int_{C(t)} \epsilon$.

Impulse connections between incident simplices An infinitesimal parallel transport of a vector *from* a simplex *to* one of its cofaces results in a sudden change of frame—creating a finite rotation of the vector coordinates even if the connection is supposed to be flat. For instance, infinitesimally moving from a vertex v_i to one of its emanating edges e_{ij} requires a finite change of coordinates from frame $\mathbf{e}_{v_i}$ to frame $\mathbf{e}_{e_{ij}}$. The same is true for a motion from v_i to a coface triangle t_{ijk}, and from a point on an edge e_{ij} to the same point considered as part of a coface triangle t_{ijk} (see Fig. 4.1(left)). We thus propose to encode our discrete connection between each vertex and its cofaces using rotation angles: they represent the integral of *impulse* (Dirac) connection 1-forms, since they integrate to a finite rotation angle over a zero-length path. For this reason, we refer to them as "impulse rotations".

We denote by $\rho_{v_i \to e_{ij}}$ the vertex-to-edge impulse rotation angle such that $\exp(-J\rho_{v_i \to e_{ij}})$ applied to the components of a vector at v_i expressed in frame $\mathbf{e}_{v_i}$ gives the expression of the parallel-transported vector, still at v_i but considered as part of e_{ij}, hence written in frame $\mathbf{e}_{e_{ij}}$. Similarly, the impulse parallel transport from a vertex v_i to the corner of an incident triangle t_{ijk} is denoted by the rotation angle $\rho_{v_i \to t_{ijk}}$, and the impulse rotation from a point $\mathbf{p}$ on e_{ij} to the same point considered as part of t_{ijk} is denoted as $\rho_{e_{ij} \to t_{ijk}}(\mathbf{p})$. One has $\rho_{v_i \to \sigma} = -\rho_{\sigma \to v_i}$ by definition. We also impose that

$$\rho_{v_i \to e_{ji}} = (\rho_{v_i \to e_{ij}} - \pi) + 2\pi n_{ij},$$

where n_{ij} is either 0 or 1, denoting a possible full rotation necessary to align the two opposite edge-based frames. Note that a single non-zero n_{ij} per vertex v_i is needed: essentially, this 2π offset is needed to form a locally consistent frame field in v_i's one-ring; note also that these n_{ij} coefficients will have no effect on the connection curvatures: they are purely for angle "bookkeeping" purposes. We will describe a systematic way to set the impulse rotations and 2π offsets for a given connection in Sec 4.3.

Note that the connection induced by the Euclidean embedding of a mesh is encoded by an edge-to-triangle rotation $\rho_{e_{ij} \to t_{ijk}}(\mathbf{p})$ equal to a constant angle $\bar{\rho}_{e_{ij} \to t_{ijk}}(\mathbf{p}) = \angle(\mathbf{e}_{e_{ij}}, \mathbf{e}_{t_{ijk}})\ \forall \mathbf{p} \in e_{ij}$, where $\angle$ is measured in the Euclidean metric: in this locally flat metric, a vector only undergoes a change of frame between the edge and its incident triangle in the local hinge map of the edge. Our discrete notion of connection, involving more impulse rotations, is thus a generalization of the continuous notion. While the impulse rotations depend on the choice of simplex frames, they are geometric (intrinsic) entities that define parallel transport from vertices to nearby simplices. A change of simplex frames would therefore require different impulse rotations to encode the same notion of parallel transport—just like a change of a frame field (a section of the frame bundle) in the continuous case would generate a different 1-form connection.

Curvature of discrete connection A discrete connection with arbitrary assignment of impulse rotations and Whitney-based edge rotations has non-zero curvature almost everywhere. In particu-

lar, a path around the perimeter of a triangle t_{ijk} creates a rotation angle

$$-K_{ijk} = \tau_{ij,k} + \tau_{jk,i} + \tau_{ki,j}.$$

The connection also has curvature around half-edges: a closed path from a point $\mathbf{p}$, along edge e_{ij}, to another point $\mathbf{q}$ on the edge, crossing to face t_{ijk}, following the border edge along $\mathbf{q} \to \mathbf{p}$, then back to $\mathbf{p}$ (Fig. 4.2(left)) accumulates a rotation angle

$$-K_{ij,k}(\mathbf{p}, \mathbf{q}) = -\rho_{e_{ij} \to t_{ijk}}(\mathbf{p}) + \epsilon_{ij}[\varphi_j(\mathbf{q}) - \varphi_j(\mathbf{p})] + \rho_{e_{ij} \to t_{ijk}}(\mathbf{q}) + \tau_{ij,k}[\varphi_j(\mathbf{p}) - \varphi_j(\mathbf{q})]$$

where the index after the comma indicates the half-edge around which the closed path is considered. We will denote by $K_{ij,k}$ the total curvature engendered by a path around the whole half-edge (i.e., starting at v_i, going through v_j, and coming back), that is:

$$-K_{ij,k} = \rho_{v_i \to e_{ij}} + \epsilon_{ij} - \rho_{v_j \to e_{ij}} + \rho_{v_j \to t_{ijk}} + \tau_{ji,k} - \rho_{v_i \to t_{ijk}}$$

The curvature of a discrete connection, unlike the curvature of the triangle mesh which was concentrated purely at vertices, is now "spread" around, with triangle curvatures K_{ijk} and half-edge curvatures $K_{ij,k}$. This will actually render the notion of covariant derivative finite as we discuss in Sec. 4.4.1.

4.2.3 Reduced parameters for discrete connections

While the full description of a discrete connection 1-form requires a large amount of impulse rotations and edge rotations, we propose to focus on discrete connections with *finite curvature* only. That is, we reduce the permissible set of connections by enforcing that their curvatures over loops with zero areas are identically zero, i.e., $K_{ij,k} = 0$. This particular choice is made to enforce pointwise finite values of the covariant derivative, and thus, a finite L_2 norm. We can now parameterize the set of all such discrete connection 1-forms through vertex-based values only, indicating the $3|F| + 3|E|$ rotation angles from each vertex to incident simplices (see Fig. 4.2(right)):

- $3|F|$ vertex-to-triangle impulse rotations $\rho_{v_i \to t_{ijk}}$,

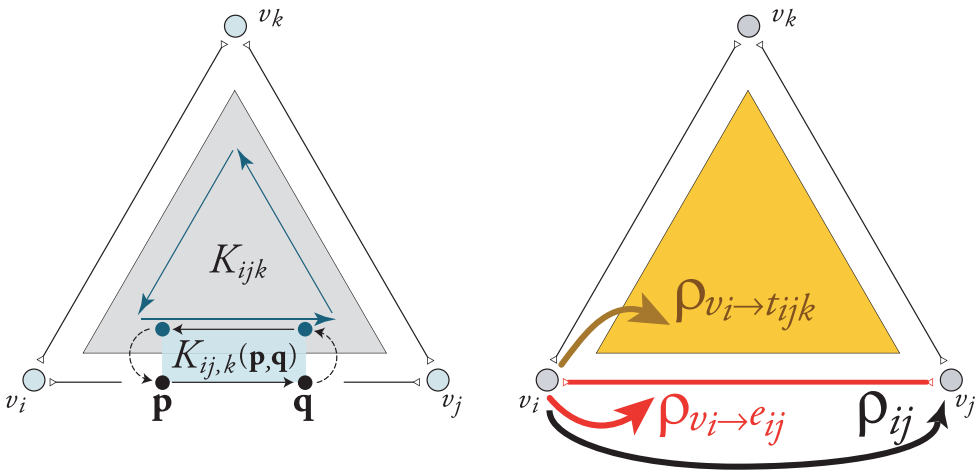


Figure 4.2 **Curvature and Parameters.** Left: Curvature is accumulated along a closed path around the interior of a triangle (K_{ijk}) or a closed path around a section of a half-edge ($K_{ij,k}(\mathbf{p}, \mathbf{q})$). Right: A discrete connection ρ with finite curvature ($K_{ij,k} = 0$) is encoded through only vertex-triangle, vertex-to-edge, and vertex-to-vertex rotation angles.

- $2|E|$ vertex-to-edge impulse rotations $\rho_{v_i \to e_{ij}}$,
- and $|E|$ vertex-to-vertex rotations ρ_{ij}, representing the connection integral from v_i, along e_{ij}, then to v_j.

All other impulse rotations and edge rotations are directly deduced from these reduced parameters by enforcing that $K_{ij,k} = 0$. In particular, we get

$$\begin{aligned} \epsilon_{ij} &= -\rho_{v_i \to e_{ij}} + \rho_{ij} + \rho_{v_j \to e_{ij}}, \\ \tau_{ij,k} &= -\rho_{v_i \to t_{ijk}} + \rho_{ij} + \rho_{v_j \to t_{ijk}}. \end{aligned} \tag{4.1}$$

Notice that a discrete connection in this reduced set has a linearly varying impulse rotation angle $\rho_{e_{ij} \to t_{ijk}}$ at a point $\mathbf{p} \in e_{ij}$ between the edge e_{ij} and an incident triangle t_{ijk} as:

$$\begin{aligned} \rho_{e_{ij} \to t_{ijk}}(\mathbf{p}) &= -\rho_{v_i \to e_{ij}} + \epsilon_{ij}\varphi_j(\mathbf{p}) + \rho_{v_i \to t_{ijk}} + \tau_{ij,k}\varphi_j(\mathbf{p}) \\ &= \varphi_i(\mathbf{p})\,(\rho_{v_i \to e_{ij}} - \rho_{v_i \to t_{ijk}}) + \varphi_j(\mathbf{p})\,(\rho_{v_j \to e_{ij}} - \rho_{v_j \to t_{ijk}}). \end{aligned} \tag{4.2}$$

Also, the connection curvature in a triangle's interior is solely determined by ρ_{ij}:

$$-K_{ijk} = \rho_{ij} + \rho_{jk} + \rho_{ki}.$$

We will now denote by ρ the discrete connection defined by these reduced parameters, i.e., $\rho = (\{\rho_{ij}\}, \{\rho_{v_i \to e_{ij}}\}, \{\rho_{v_i \to t_{ijk}}\})$. As we are about to see, any such discrete connection can be used to define a finite-dimensional space of vector fields on the surface and its associated differential operators. However, most geometry processing tools assume the Levi-Civita connection induced by the Euclidean embedding. We thus describe next how to define a discrete connection as close as possible to the original Levi-Civita connection of the mesh, while keeping covariant derivatives finite.

4.3 Computing Levi-Civita Connections

The reduced parameters of our formulation of connections over triangulated manifolds need to be determined to create an instance of discrete connection. We first provide local choices of reduced parameters that were implicit in previous work, before introducing a global optimization procedure that mimics the work of [26] but within our (primal) connection setup, in the sense that it makes the discrete connection as close as possible to the canonical Levi-Civita connection $\bar{\rho}_{e_{ij} \to t_{ijk}}$ of the input surface.

4.3.1 Connection derived from geodesic polar maps

One choice for evaluating ρ_{ij} based on local measurements from the input mesh makes use of the geodesic polar map as in [13] and [1]. The geodesic polar map proportionally rescales tip angles around each vertex such that they sum to 2π, inducing a flattening of the immediate surrounding of each vertex v_i through a scaling factor

$$s_i = 2\pi / \sum_{t_{ijk}} \theta_{kij} \equiv 2\pi / (2\pi - \kappa_i), \tag{4.3}$$

where κ_i is the Gaussian curvature for v_i. Assuming for simplicity that the frame basis vector $\mathbf{e}_{v_i}$ at vertex v_i was chosen to be aligned to one of the adjacent edge frames $\mathbf{e}_{e_{im}}$, it is natural to select $\rho_{v_i \to e_{ij}} = s_i \angle(\mathbf{e}_{e_{ij}}, \mathbf{e}_{e_{im}})$, where the positive angle $\angle$ is measured by summing triangle tip angles

counterclockwise around v_i between possibly non-consecutive edges e_{ij} and e_{im} around vertex v_i. Next, we select $\rho_{v_i \to e_{ji}} = (\rho_{v_i \to e_{ij}} - \pi) + 2\pi n_{ij}$, where $n_{im} = 1$ and $n_{ij} = 0 \ \forall j \neq m$. With this choice, the Gaussian curvature of v_i is distributed to its one-ring, since the integral of Gaussian curvature around the tip of triangle t_{kij} is now:

$$\begin{aligned}
&- \left[\rho_{v_i \to e_{ij}} + (\bar{\rho}_{e_{ij} \to t_{ijk}} - \bar{\rho}_{e_{ki} \to t_{ijk}}) - \rho_{v_i \to e_{ki}}\right] \\
&= -\rho_{v_i \to e_{ij}} + (\pi - \theta_{kij}) + (\rho_{v_i \to e_{ik}} - \pi) + 2\pi n_{ik} \\
&= (s_i - 1)\ \theta_{kij}.
\end{aligned}$$

The vertex-to-vertex coefficient ρ_{ij} of the discrete connection is then set to be:

$$\rho_{ij} = \rho_{v_i \to e_{ij}} - \rho_{v_j \to e_{ij}},$$

and the triangle curvature K_{ijk} of the connection becomes:

$$K_{ijk} = (s_i - 1)\theta_{kij} + (s_j - 1)\theta_{ijk} + (s_k - 1)\theta_{jki}.$$

This is precisely the choice that the authors of [1] made, except that their restriction on the range of Gaussian curvature is unnecessary here. Given that our continuous edge- and triangle-wise connections are entirely determined by the coefficients of ρ, our approach thus provide a *closed-form expression* of the continuous connection they implicitly used.

This choice of vertex-to-vertex rotation angles does not, however, fully determine a discrete connection—although it is enough to evaluate the Dirichlet energy of a vector field as we will see in Sec. 4.4. Indeed, impulse rotations from vertex to triangles are crucial for the local evaluation of the first-order derivatives divergence, curl and $\bar{\partial}$. An intuitive choice for these vertex-to-triangle rotations is to use the vertex-to-edge impulse rotations induced by the vertex-to-vertex coefficients and the well-defined angles (measured in the actual Euclidean metric) from the edge frame to the triangle frame, i.e.,

$$\rho_{v_i \to t_{ijk}} = \rho_{v_i \to e_{ij}} + \bar{\rho}_{e_{ij} \to t_{ijk}},$$

where the Levi-Civita connection $\bar{\rho}_{e_{ij} \to t_{ijk}} = \angle(\mathbf{e}_{e_{ij}}, \mathbf{e}_{t_{ijk}})$ of the input mesh is used. However, this choice is biased since it only considers the impulse rotations of e_{ij} and not of its neighboring

edges. To be consistent with the geodesic polar map, the rotation from the vertex frame basis $\mathbf{e}_{v_i}$ to any direction between e_{ij} and e_{ik} should be directly computed based on the scaling factor s_i, and should result in a rotation angle in between $\rho_{v_i \to e_{ij}}$ and $\rho_{v_i \to e_{ik}}$. One of the many different ways to enforce this property is thus to pick an arbitrary interior point $\mathbf{c}_{ijk}$ (such as the incenter or the barycenter) of each triangle t_{ijk}, to define $\rho_{v_i \to \mathbf{c}_{ijk}} = (\rho_{v_i \to e_{ij}} + \rho_{v_i \to e_{ik}} + 2\pi n_{ik})/2$, and to define the vertex-to-triangle impulse rotations as

$$\rho_{v_i \to t_{ijk}} = \rho_{v_i \to \mathbf{c}_{ijk}} + \angle(\mathbf{c}_{ijk} - \mathbf{p}_i, \mathbf{e}_{t_{ijk}}), \tag{4.4}$$

where, again, the angles $\angle$ are measured in the actual Euclidean metric of the input mesh.

4.3.2 Locally optimal connection 1-form

The choice of geodesic polar map may, however, result in large connection values τ (as deduced from ρ through Eq. 4.1), indicating a significant mismatch between the local original Levi-Civita connection (**0** inside a triangle) and its discrete counterpart. A simple improvement can be achieved by choosing the vertex-to-triangle rotations that minimize the L_2 norm of this deviation within each triangle while keeping the vertex-to-vertex coefficients ρ_{ij} unchanged. As the L_2 norm of τ per triangle is a quadratic function of its edge values $\tau_{ij,k}$, $\tau_{jk,i}$, and $\tau_{ki,j}$ using the mass matrix of Whitney 1-form basis functions, the local optimal values are found in closed form to be simply $\tau_{ij,k} = -K_{ijk}/3$, which leads to

$$\int_{t_{ijk}} \tau \wedge \star\tau = \frac{1}{36}(\cot(\theta_{ijk}) + \cot(\theta_{jki}) + \cot(\theta_{kij}))K_{ijk}^2.$$

There are, however, multiple choices of vertex-to-triangle impulse rotations that achieve this locally minimal connection. For instance, we could pick one arbitrary impulse $\rho_{v_i \to t_{ijk}}$ per triangle t_{ijk}, then find $\rho_{v_j \to t_{ijk}}$ and $\rho_{v_k \to t_{ijk}}$ so that, for $q \in \{j, k\}$,

$$\rho_{v_q \to t_{ijk}} = \rho_{v_i \to t_{ijk}} + \rho_{qi} + \tau_{iq}. \tag{4.5}$$

One can, instead, compute the three triplets of vertex-to-triangle impulse rotations induced by fixing each one of the corner impulse rotations individually using Eq. 4.5, and average their values to avoid

bias. This averaged choice leads to better accuracy in singularity direction control (see Sec. 4.5), and has proven to be, in all our tests, the *local* definition of connection that generates the least amount of numerical errors (see Table 4.1).

4.3.3 As-Levi-Civita-as-possible connection 1-form

Deriving a discrete connection through a geodesic polar map [1] leads to reasonable connection 1-forms ρ_{ij} on primal edges, and local optimizations of impulse rotations further minimize the resulting triangle-based connection 1-form. We can, however, directly compute a globally optimal discrete connection by selecting the one closest (in a proper sense) to the actual Levi-Civita connection $\bar{\rho}$ derived from the piecewise flat mesh.

In order to define a meaningful notion of optimal connection, we propose the following two measurements of deviation:

$$D_T(\rho) = \sum_{t_{ijk}} \int_{t_{ijk}} \tau \wedge \star\tau,$$
$$D_E(\rho) = \sum_{e,t \mid e \subset t} w_{e,t} \int_e (\rho_{e\to t}(\mathbf{p}) - \bar{\rho}_{e\to t})^2 dl,$$

where $\rho_{e\to t}(\mathbf{p})$ is the linear-varying impulse rotation given in Eq. 4.2, and $w_{e_{ij},t_{ijk}} = \tan\theta_{jki}$ is the inverse of the cotan weight for the Hodge star of 1-forms within the triangle (for tip angles greater than or equal to $\pi/2$, we can use a fixed large value for $w_{e,t}$ instead without substantial impact on the resulting coefficients, as the effect of the cotan weights on the global result is minor as noticed in [26] for the dual version). D_T measures the deviation from the flat connection within triangles, while D_E measures the difference between the true Levi-Civita connection measured by the angles $\angle$ on the input mesh and the impulse rotation induced by the reduced parameters of ρ. Minimizing the quadratic total deviation $D_T + D_E$ (or any linear combination thereof) is thus simple: the optimization procedure amounts to solving a linear system in ρ after we fix its kernel of

size $|V|$ by setting to zero one of the vertex-to-face impulse rotations $\rho_{v_i \to t_{ijk}}$ per vertex v_i (these $|V|$ gauge values do not affect the result, as they amount to a rotation angle of the arbitrary frame direction $\mathbf{e}_{v_i}$). Both energies are expressed as quadratic functions of ρ, but the integrated deviation D_E does not depend on ρ_{ij} since the contributions from ϵ and τ cancel out along each edge.

4.3.4 Trivial connections

We just described how our definition of a discrete connection can be made as close as possible to the Levi-Civita connection $\bar{\rho}$ through a linear solve. In fact, we can also create a connection as close as possible to *any* metric connection with arbitrary cone singularities at vertices, similar to the *trivial connections* of [26]: in our context, trivial connections are created by using angles $\tilde{\rho}_{e_{ij} \to t_{ijk}} = \bar{\rho}_{e_{ij} \to t_{ijk}} + \alpha_{ij,k}$, where $\alpha_{ij,k}$ is an adjustment angle, and the cone singularity at v_i has a connection curvature

$$K_i = \sum_{t_{ijk}} (\rho_{v_i \to e_{ij}} + \tilde{\rho}_{e_{ij} \to t_{ijk}} - \tilde{\rho}_{e_{ki} \to t_{ijk}} - \rho_{v_i \to e_{ki}}).$$

If the adjustment angles have been picked such that $K_i = 0$ for most vertices, and if we replace $\bar{\rho}_{e \to t}$ in the deviation D_E by $\tilde{\rho}_{e \to t}$, our optimization will lead to trivial connections, thus extending the method of [26] to our primal setup. As we will demonstrate in Sec. 4.6, our optimization of the discrete connection improves the accuracy of all further numerical evaluations. More importantly, we can now formulate in closed-form (pointwise or locally integrated) derivatives *and* their L_2 norms as explained next.

4.4 Connection-based Operators

From a discrete connection ρ and a vector $\mathbf{u}_i = u_i^1 \mathbf{e}_{v_i} + u_i^2 \mathbf{e}_{v_i}^{\perp}$ at each vertex, we can give an exact expression of the vector field $\mathbf{u}$ within each triangle of $\mathcal{T}$. Consequently, any first-order operator or energy will be easy to evaluate using the resulting continuous vector field. The key observation in constructing basis functions of vector fields is that while the basis expressions contain jumps

along edges, the covariant derivative is finite everywhere as these jumps are compensated for by the discrete connection.

4.4.1 Basis functions for vector fields

Given a discrete connection ρ, we define a basis function Ψ_i per vertex v_i. Its expression $\Psi_i|_t$ within each incident triangle t is constructed by first using the rotation $-\rho_{v_i \to t}$ to parallel transport the vector $\mathbf{u}_i$ stored in the local frame $\mathbf{e}_{v_i}$ of v_i to the corner of triangle t; we then parallel transport the resulting vector expressed in the frame $\mathbf{e}_t$ along a straight path from v_i to an arbitrary point $\mathbf{p}$ in t under the connection 1-form τ, which defines a local frame field

$$\Phi_i\Big|_t(\mathbf{p}) = (\mathbf{e}_t, \mathbf{e}_t^{\perp}) \exp\left[-(\rho_{v_i \to t} + \int_{v_i \to \mathbf{p}} \tau) J\right].$$

With these local frame fields, we make use of the scalar basis functions φ_i at $\mathbf{p}$ to blend the parallel transported vectors from each corner of the triangle. Since our connection τ is linear within each triangle, the resulting basis function for a vertex v_i is easily expressed in closed form as:

$$\begin{aligned}\Psi_i\Big|_{t_{ijk}}(\mathbf{p}) &= \varphi_i(\mathbf{p})\Phi_i\Big|_{t_{ijk}}(\mathbf{p}) \\ &= \varphi_i(\mathbf{p})(\mathbf{e}_{t_{ijk}}, \mathbf{e}_{t_{ijk}}^{\perp}) \exp\Big[-(\tau_{ij,k}\varphi_j(\mathbf{p}) + \tau_{ik,j}\varphi_k(\mathbf{p}) + \rho_{v_i \to t_{ijk}})J\Big].\end{aligned}$$

The interpolated vector field $\mathbf{u}$ can then be evaluated anywhere on the mesh via

$$\mathbf{u} = \sum_i \Psi_i \begin{pmatrix} u_i^1 \\ u_i^2 \end{pmatrix}.$$

Note that this interpolation is visually quite similar to a linear interpolation for a discrete as-Levi-Civita-as-possible connection, but can be dramatically different for other connections. Fig. 4.3 shows a vector (in green) locally interpolated by a basis Ψ_i over a non-flat one-ring for an as-Levi-Civita-as-possible connection (*left*) vs. when one of the vertex-to-face connection angles has been doubled (*right*).

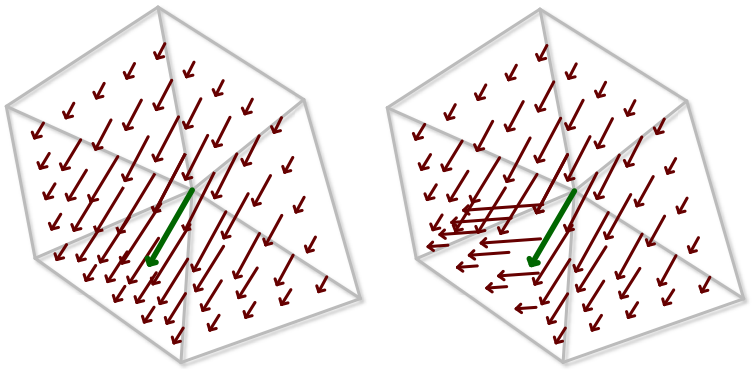


Figure 4.3 **Locally interpolated vector by different connections.**

4.4.2 Discrete covariant derivative

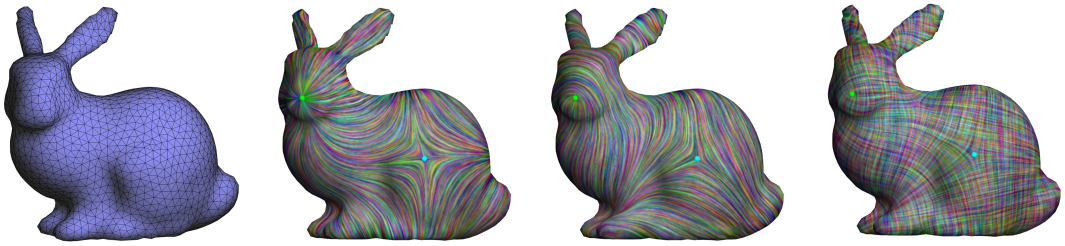
Using the decomposition given in Eq. 2.3 and dropping the basis $(\mathbf{e}_t, \mathbf{e}_t^{\perp})$ for clarity, the covariant derivative of our basis functions within triangle t_{ijk} is formally derived via:

$$\begin{aligned}
\nabla\Psi_i &= \nabla(\varphi\Phi) = \Phi_i \otimes d\varphi_i + \varphi_i \nabla\Phi_i \\
&= \Phi_i \otimes d\varphi_i - J\Psi \otimes (\tau_{ij,k} d\varphi_j + \tau_{ik,j} d\varphi_k) + J\Psi \otimes \tau \\
&= \Phi_i \otimes d\varphi_i - J\Psi \otimes (\tau_{ij,k}(-\varphi_{jk} + \varphi_{ij}) \\
&\qquad + \tau_{ik,j}(\varphi_{jk} - \varphi_{ki})) + J\Psi \otimes \tau \\
&= \Phi_i \otimes d\varphi_i - K_{ijk} J\Psi_i \otimes \varphi_{jk}.
\end{aligned}$$

Note that while the basis functions Ψ_i depend on the choice of vertex-to-triangle impulse rotations $\rho_{v\to t}$, their mass matrix $\int_{\mathcal{T}} \Psi_i \cdot \Psi_j$ (resp., stiffness matrix $\int_{\mathcal{T}} \nabla\Psi_i : \nabla\Psi_j$) does not depend on it, since, e.g.,

$$\Psi_i(\mathbf{p}) \cdot \Psi_j(\mathbf{p}) = \varphi_i(\mathbf{p})\varphi_j(\mathbf{p}) \exp[J(K_{ijk}\varphi_k(\mathbf{p}) + \rho_{ij})].$$

Consequently, the energies E_D, E_H, and E_A do not have this dependence either, and their expressions are similar to the result of [1] except that we use an optimized connection ρ instead of the vertex-to-vertex coefficients derived from the geodesic polar map (Sec. 4.3.1). However, rotations $\rho_{v\to t}$ are necessary for the evaluation of pointwise or integrated first-order derivatives such as divergence, curl, and Cauchy-Riemann operators.



(a) coarse bunny mesh (b) vector field (c) direction field (d) cross field

Figure 4.4 **From vector field to n-vector fields.** A discrete vector field, even on a coarse mesh, can be directly converted into an n-vector or n-direction field by scaling the connection angles. Here, a bunny mesh (a) and a vector field with a source and a saddle on one side (b) is converted into a 2-RoSy (direction) field (c) and a 4-RoSy (cross) field (d).

4.4.3 Discrete operators based on covariant derivative

To derive the integrals of first-order operators per triangle, it is convenient to choose a barycentric-coordinate parametrization $(x(\mathbf{p}), y(\mathbf{p})) = (\varphi_j(\mathbf{p}), \varphi_k(\mathbf{p}))$ in triangle t_{ijk}, for which the metric is

$$g = \begin{pmatrix} \mathbf{e}_{ij} \cdot \mathbf{e}_{ij} & \mathbf{e}_{ij} \cdot \mathbf{e}_{ik} \\ \mathbf{e}_{ij} \cdot \mathbf{e}_{ik} & \mathbf{e}_{ik} \cdot \mathbf{e}_{ik} \end{pmatrix}.$$

The components of $\nabla\Psi_i$ can now be straightforwardly evaluated given any constant frame field $(\mathbf{e}_1, \mathbf{e}_2)$ within the triangle. For instance, if one picks $\mathbf{e}_1 = \frac{1}{g_{11}}\frac{\partial}{\partial x}$, one gets inside triangle t_{ijk}:

$$\begin{aligned} \nabla_{\mathbf{e}_1}\Psi_i &= \Phi_i d\varphi_i(\mathbf{e}_1) - K_{ijk}\varphi_i J\Phi_i(\varphi_j d\varphi_k - \varphi_k d\varphi_j)(\mathbf{e}_1) \\ &= \frac{1}{g_{11}}\left(dx(\frac{\partial}{\partial x}) - K_{ijk}xJ(-yd(x+y))(\frac{\partial}{\partial x})\right)\Phi_i \\ &= \frac{1}{g_{11}}(I + K_{ijk}xyJ)\Phi_i. \end{aligned}$$

The four operators involved in Eq. 2.3 are then assembled via

$$div\ \Psi_i = \mathbf{e}_1 \cdot \nabla_{\mathbf{e}_1}\Psi_i + \mathbf{e}_2 \cdot \nabla_{\mathbf{e}_2}\Psi_i,$$

$$curl\ \Psi_i = \mathbf{e}_1 \cdot \nabla_{\mathbf{e}_2}\Psi_i - \mathbf{e}_2 \cdot \nabla_{\mathbf{e}_1}\Psi_i,$$

$$div\ \Psi_i = \mathbf{e}_1 \cdot \nabla_{\mathbf{e}_1}\Psi_i - \mathbf{e}_2 \cdot \nabla_{\mathbf{e}_2}\Psi_i,$$

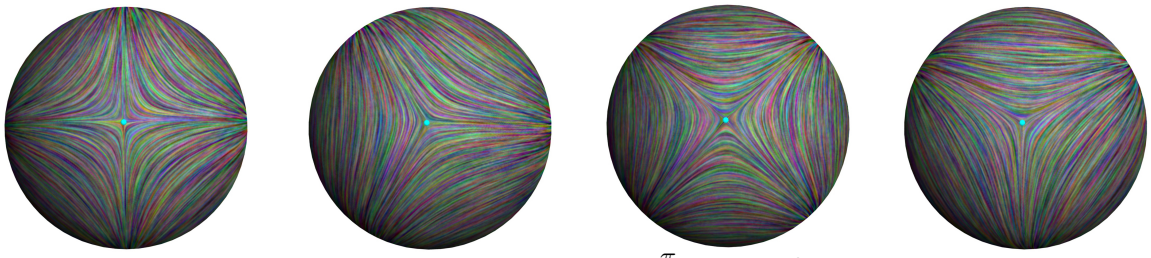
$$curl\ \Psi_i = \mathbf{e}_1 \cdot \nabla_{\mathbf{e}_2}\Psi_i + \mathbf{e}_2 \cdot \nabla_{\mathbf{e}_1}\Psi_i.$$

Note that, as expected, a rotation by θ in the triangle's local frame produces no change in $\mathrm{d}iv$ or $curl$, but it results in a rotation $\exp(J2\theta)$ of the Cauchy-Riemann operator $\bar{\partial}=1/2(div,curl)$. If on the other hand, the connection from a vertex v to an incident triangle t is changed by an angle θ, it results in a redistribution of the four terms $(\mathrm{d}iv_{\text{new}}, curl_{\text{new}})^T = \exp(J\theta)(\mathrm{d}iv, curl)^T$ and $\bar{\partial}_{\text{new}} = \exp(J\theta)\bar{\partial}$, but their combined L_2-norms (E_A and E_H) remain unchanged.

Triangle-based Integrals The discrete versions of these operators are defined as their continuous integrals over triangles as it provides numerically robust local averages:

$$\mathrm{d}iv_t\ \Psi_i = \int_t \mathrm{d}iv\,\Psi_i,\quad curl_t\ \Psi_i = \int_t curl\,\Psi_i,\quad \bar{\partial}_t\Psi_i = \int_t \bar{\partial}\Psi_i.$$

The integration can be done in closed form since it essentially involves terms such as $x\exp(Jx)$. For numerical evaluation, Chebyshev expansion is recommended [1] to handle the expressions when the connection curvature is either small or large. However, with our optimized connection, it is safe to assume that the curvature is small enough to use a simpler Taylor expansion, with essentially the same accuracy. While the integral of our discrete connections on local half-edge cycles (Fig. 4.2) is zero by design, the total integral of the discrete operators we just formed does not necessarily vanish as it should: the triangle integral of divergence reduces to the boundary integral formed by half-edges considered as part of the triangle, which therefore do not account for the edge integrals. Thus, Stokes' theorem for divergence and curl will not hold when we sum triangle integrals. In fact, this discrepancy between integral along the boundary of triangles vs edges is only one of the two sources of inaccuracy: the other source is the deviation of τ from the (trivial) Levi-Civita connection within each triangle. It bears noticing that our optimization target function in Sec. 4.3.3 is precisely a measure of these two discrepancies. Thus, our optimized discrete connections lead to higher quality first-order derivative operators than those induced by the geodesic polar map. The final expressions of our discrete operators are analytically found through symbolic integration, see App. A.3.



(a) vector field with saddle (b) direction field from (a) (c) $\frac{\pi}{4}$-rotation of (a) (d) direction field from (c)

Figure 4.5 **Orientation control for negative index singularities.** From a vector field (a) on a sphere with a saddle point with index -1 (resp., its corresponding 2-RoSy field (b) forming a trisector of index $-1/2$), the user can directly control the orientation (c) of the saddle (resp., the orientation of the trisector (d)) without affecting its position on the surface.

Edge-based Integrals If a precise enforcement of Stokes' theorem is required, the per-triangle integral evaluation of first-order derivatives can be defined via boundary integrals instead: using our edge-based connection ϵ, we can define another set of discrete operators, defined on each triangle as

$$\mathrm{d}iv_t\ \Psi_i = \int_{\partial t} \Psi_i \times d\mathbf{l}, \quad curl_t\ \Psi_i = \int_{\partial t} \Psi_i \cdot d\mathbf{l},$$

where the basis function Ψ is expressed along the edge as:

$$\Psi_i|_{e_{ij}}(\mathbf{p}) = \varphi_i(\mathbf{p}) \exp[-J(\epsilon_{ij}\varphi_j(\mathbf{p}) + \rho_{v_i \to e})].$$

The Cauchy-Riemann operator is defined in a similar fashion via:

$$\bar{\partial}_t \Psi_i = \frac{1}{2} \int_{\partial t} ((F\Psi_i) \times d\mathbf{l}, (F\Psi_i) \cdot d\mathbf{l})^T,$$

where the reflection F is done w.r.t. the frame $\mathbf{e}_t$ in triangle t. The closed-form expressions of these discrete operators are given in App. A.2. Both triangle-based and edge-based discrete approaches to evaluating local integrals of first-order derivatives exhibit similar numerical accuracy, as we will discuss in Sec. 4.6.

4.5 Vector and n-Direction Field Design

The operators and energies we have defined based on our discrete connection are well suited to the design of visually-smooth vector fields on triangle meshes through basic linear algebra, as one

has control over the behavior of their singularities (both position *and* orientation) as well as their alignment. In this section, we present two different approaches to vector field design that build upon and extend previous work through the use of our discrete connections and covariant derivatives. Note that creating a smooth n-vector or n-direction field is also a trivial matter: the exact same vector field design procedure can be used first in a connection where all angles have been multiplied by n, and the resulting vector field is converted to an n-vector field by dividing the angle the vector field makes with each vertex reference direction $\mathbf{e}_{v_i}$ by n (see Fig. 4.4). We can then normalize the resulting n-vector field to make it an n-direction field as proposed in [1].

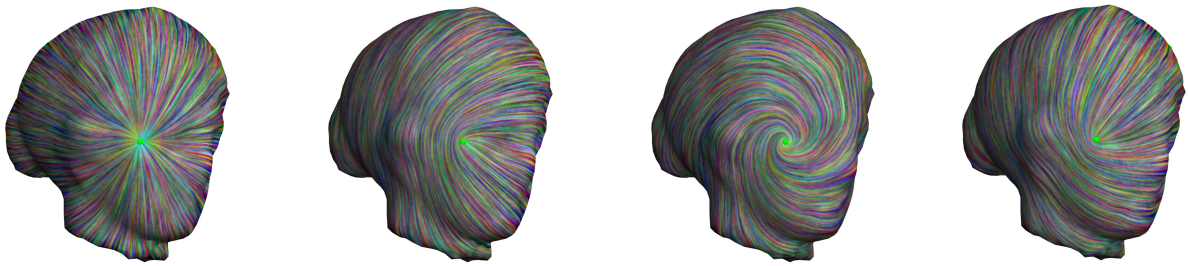
It should be noted here, as it will become important in the course of this section, that for an n-vector field $\mathbf{u}$ with $n \geq 2$, the notions of divergence and curl become dependent on the choice of frame: they now represent the components of an $(n-1)$-vector field $\partial\mathbf{u}$ as we demonstrate in App. A. Conversely, the reflected divergence and reflected curl represent an $(n+1)$-vector field $\bar{\partial}\mathbf{u}$.

4.5.1 Variational approach

The overall procedure of our first approach to design a vector field is based on a quadratic minimization driven by user-specified constraints, extending the approach of [17]. From a globally-optimized discrete connection, we define a penalty energy P for a vector field $\mathbf{u}$ as:

$$P(\mathbf{u}) = \tfrac{1}{2}\int_{\mathcal{T}} (\mathrm{d}iv\,\mathbf{u} - d)^2 + (curl\,\mathbf{u} - c)^2 + (\bar{\partial}\mathbf{u} - \mathbf{s})^2 + w(\mathbf{u} - \mathbf{u}_0)^2,$$

where d prescribes sources/sinks, c controls vortices, $\mathbf{s}$ describes the desired saddle points, $\mathbf{u}_0$ is a guidance vector field, and w is a weight used for local or global alignment constraints. The integration of this quadratic energy can be done on a per-triangle basis, which reduces to a Poisson-like linear system $A\mathbf{U} = b$ for a matrix $A = -2\Delta_\omega + wI$, where Δ_ω can be seen as the discrete version of the connection Laplacian (which handles boundary conditions naturally, unlike the deRham Laplacian used in [17]). This matrix A has the exact same structure as the one in [1], except that we use our optimized ρ_{ij} instead of vertex-to-vertex rotations induced by the geodesic polar map. The right hand side term b relies on the discrete divergence, curl and Cauchy-Riemann operators,



(a) Vector field with a source (b) The "wedge" from(a) (c) A vortex added to (a) (d) Wedge rotated by $\frac{\pi}{3}$

Figure 4.6 **Orientation control of positive index singularities.** By setting a divergence/curl pair $(1, 0)$ on a triangle, a source (singularity of index 1) is formed in the vector field (resp., a wedge singularity of index $1/2$ on the associated 2-RoSy field). Changing this pair to $(\cos(\frac{\pi}{3}), \sin(\frac{\pi}{3}))$, a vortex (c) is added to the source (creating log-spiraling streamlines) while the corresponding orientation field (d) has its wedge rotated by $\pi/3$.

which use our optimized vertex-to-triangles coefficients as well—this term is an extension of the work of [29] for non-flat domains. While we will not explore this possibility here, note that the user can also start from a chosen trivial connection (see Sec. 4.3.3) instead of the Levi-Civita connection for an even greater flexibility in editing.

Controlling singularity orientation Using our penalty energy P, we can control the orientation of positive index singularities, including vortices, sources/sinks, and combinations thereof. This was already possible in the divergence- and curl-based approach of [17]. With our discrete Cauchy-Riemann operator, we now can also control negative index singularities (i.e., saddle points, see Fig. 4.5) and their direction, which was not possible in previous work.

Positively indexed singularities can be constructed by assigning pairs of non-zero values (d_{ijk}, c_{ijk}) on selected triangles (and zero for all others) representing the local divergence and curl that the user desires. Note that the ratio c/d controls the direction of singularities for n-vector fields: while the shape of an index-1 singularity in a vector field is invariant under rotation, changing a pair (d_{ijk}, c_{ijk}) to $\exp(J\theta)(d_{ijk}, c_{ijk})$ when editing an index-$1/n$ singularity in an n-vector field results in a rotation of $\theta/(n-1)$ of the singularity (see Fig. 4.6).

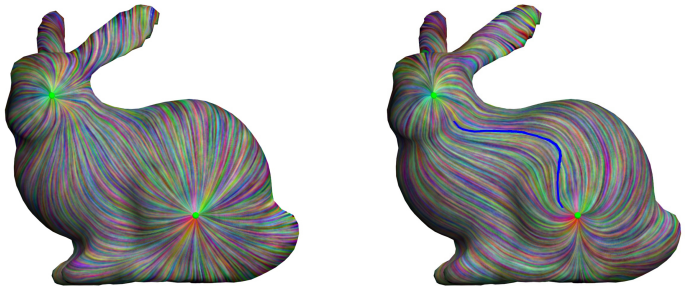
When controlling a saddle point, the ratio between the two components of $\mathbf{s}_{ijk}$ on selected triangles indicates instead the angle that the symmetry axis of the saddle point makes with the

simplicial frame field $\mathbf{e}_{t_{ijk}}$. In this case, $\bar{\partial}\mathbf{u}$ is, itself, a 2-vector field, so rotating the saddle point by $\theta/2$ amounts to using $\exp(J\theta)\mathbf{s}_{ijk}$. For $-1/n$-singularities in n-direction fields, we will get $\theta/(n+1)$ rotations instead. Fig. 4.5 shows an example where a saddle point is rotated by $\pi/3$ by changing the components of $\mathbf{s}_{ijk}$ on the triangle t_{ijk} containing the saddle.

Constraining alignment Vector or n-direction fields can also be modified via alignment constraints, either via an input direction field or via user-drawn strokes. If we are given a target n-vector or n-direction field represented by $\mathbf{u}_0$, we balance the smoothness (and singularity control if needed) and the alignment term via a user-specified weight w as indicated in the last term of energy P. For more local editing, the user can draw strokes on the mesh as an intuitive way to provide control over the design. We essentially follow the approach of [17] to create a locally supported vector field $\mathbf{u}_0$, and enforce it via the same penalty term used above, with its own weight w to let the user decide how closely the resulting vector field should follow the stroke (Fig. 4.7).

4.5.2 Eigen design

While our variational approach to editing is fast and simple, it suffers from two shortcomings: first, one needs to start from an existing vector field to begin the editing process; second, spurious singularities can appear as more constraints are input by the user. Both these issues can be addressed



(a) Original vector field (b) with a user-specified stroke

Figure 4.7 **Design by stroke.** (a) From an n-vector or n-direction field with arbitrary singularities, (b) the user can draw a stroke (blue) in order to easily influence the direction of the field. The result is updated interactively by solving the linear system resulting from the variational approach of Sec. 4.5.1.

using a different approach to vector field editing, where a vector field is provided such that it is the "smoothest" field satisfying the constraints prescribed by the user. Indeed, the authors of [1] noticed that the vector field with the lowest Dirichlet energy for a fixed L_2 norm can be found through a generalized eigenvalue problem (i.e., a Helmholtz equation), which makes use of both the connection Laplacian matrix (computing the Dirichlet energy E_D) and the mass matrix (computing the L_2 norm). We can adopt the exact same method, but now using our discrete optimized connection—resulting in improved eigen vector fields with singularities appearing at more salient locations (see Fig. 4.8).

However, our discrete operators for first-order derivatives offer a much more general extension of this design approach. Indeed, we can now modify the connection Laplacian matrix to add a quadratic penalty on the vector field components *across* user-specified strokes to directly include direction constraints in the eigenvalue problem. Similarly, the mass matrix can be modified to control both singularity placement and orientation using the terms we presented in Sec. 4.5.1. Solving the resulting generalized eigenvalue problem provides the "smoothest" vector field that satisfies user constraints, where smoothest is defined with respect to the notion of connection used to derive the covariant derivative. If the user also changes the discrete connection to be trivial with prescribed singularities as described in Sec. 4.3.3, the vector field will be smoothest for this connection as we demonstrate in Fig. 4.9. From this eigen design, variational editing (Sec. 4.5.1) can be performed if the user wishes to further edit the vector field.

4.6 Results

We present numerical tests of the accuracy of our operators derived from our discrete connection as well as a few vector field design results using our two approaches.

4.6.1 Accuracy of discrete operators

We evaluate the accuracy of the discrete approximations of $\mathrm{d}iv$, $curl$, and $\bar{\partial}$ per triangle. To allow for proper error evaluation, we use a set of triangle meshes interpolating a sphere at various levels of discretization, and use a smooth vector field (namely, a low-order vector spherical harmonic) with a known expression so that we can evaluate its exact divergence and curl everywhere. We then compute the L_2 error between our discrete divergence (resp., curl) evaluation and the real integral value per triangle. The results shown in Table 4.1 demonstrate that our optimization of the connection impacts the accuracy of first-order operators quite significantly compared to a geodesic polar map based connection. The area-based vs. edge-based evaluations of the local first-order derivatives presented in Sec. 4.4.3 are, however, minimally different. We found that the Stokes approach (based on ϵ) often leads to a better accuracy especially on fine meshes; yet, the area-based operators are slightly more robust to noise as they rely on area vs. edge integrals. We used the same setup to evaluate the accuracy of our vector field energies based on our triangle-based first-order derivatives, and once again the optimized connection shows superior numerical accuracy—except on very coarse meshes. Our Dirichlet energy results are also systematically better than the L_2 evaluation provided by [1], even if our optimal vertex-to-vertex connection angles ρ_{ij} are used to improve their results. The difference of the antiholomorphic and holomorphic energies for direction fields is also a good measure of accuracy, as we know that it should evaluate to the Euler characteristic of the mesh times 2π, and the edge-based evaluations using our optimized connections exhibit, once again, significantly improved accuracy as shown in Table 4.2. Our operators are thus well suited to vector field analysis on manifold simplicial complexes.

4.6.2 Vector and n-direction field on meshes

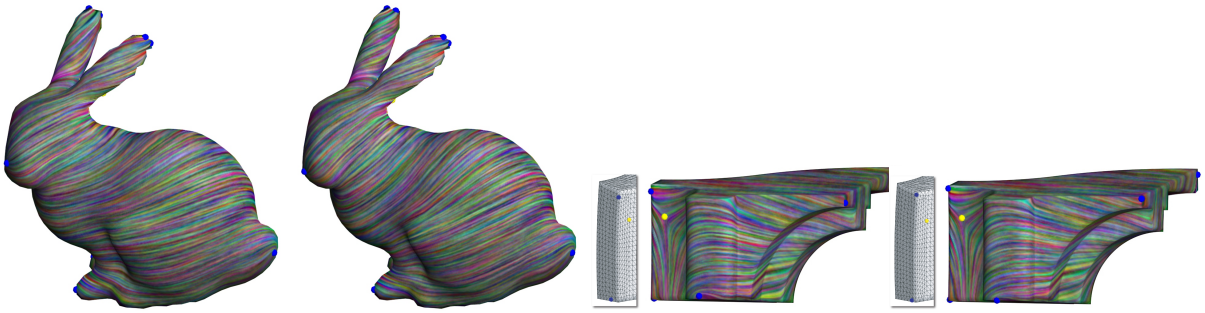
Visualization of fields was done using the code from [24]. We experimented with our variational-based editing approach based on the quadratic energy P. As expected, this simple numerical method (requiring only a linear solve for each new constraint added by the user) offers control not only over

positive singularities, but also over saddle points in the vector field and their principal axes. For instance, a saddle point happening on the side of a mesh (see Fig. 4.5) can be rotated by any angle without changing its position. The same control applies to n-direction fields without any code modification (Fig. 4.4).

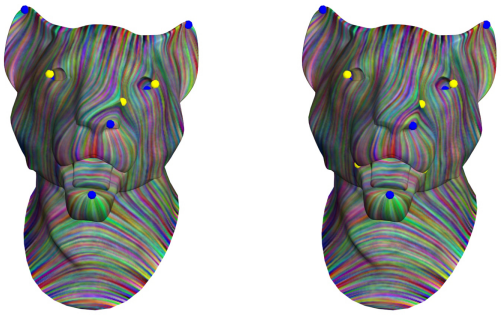
Finally, we tried our eigen approach to vector field design. First, we found that our notion of smoothest vector field for the Levi-Civita connection is quite close to the results of [1], although visual comparisons show from marginal to moderate improvements depending on the complexity of the model (see Fig. 4.8). Where our method really differs is in our ability to handle user constraints in the exact same framework, as well as arbitrary connections as demonstrated in Fig. 4.9.

4.7 Conclusion

We have proposed the construction of a discrete notion of connection and its covariant derivative by exploiting the simplicial nature of triangulated 2-manifolds and picking the lowest-order finite element basis functions we could (to simplify the resulting expressions and make vector field design as efficient as possible) such that derivatives and their L_2 norms are well defined and finite. The resulting discrete covariant derivative is linear and metric preserving by definition, although it fails to exactly satisfies Leibniz's rule as most Whitney-based discrete operators. Our notion of discrete connection was shown to be numerical superior to previous approaches, and applications to vector and direction field design were demonstrated.

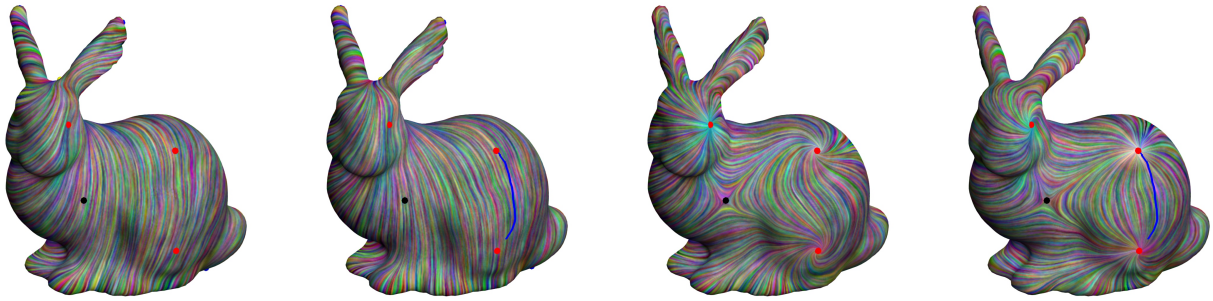


(a) [1]'s... (b) ... vs. our result (c) [1]'s... (d) ... vs. our result



(e) [1]'s... (f) ... vs. our result

Figure 4.8 **Comparisons.** While the method of [1] finds similar singularities, our approach leads to "straighter" vector fields (see neck of bunny (a) & (b); nose of lion (e) & (f)), and the positions of our singularities are found closer to corners (see insets of fandisk, (c) & (d)). Yellow and blue markers indicate the presence of singularities in the vector fields.



(a) Smoothest vector field for the Levi-Civita connection, with no constraints added

(b) Smoothest vector field (Levi-Civita connection) that matches a user-specified stroke

(c) Smoothest vector field for a trivial connection (with prescribed singularities) with no added constraints

(d) Smoothest vector field (for the same trivial connection of (c)) that matches a user-specified stroke

Figure 4.9 **Eigen design.** While an unconstrained generalized eigenvalue problem (a) will result in the smoothest vector field (i.e., with the lowest Dirichlet energy for a fixed L_2 norm) for the as-Levi-Civita-as-possible connection, we can also find the smoothest vector field that matches user-specified strokes (b). Moreover, the user can prescribe a trivial connection (c) with given singularities (both positive and negative ones, placed on the vector field singularities of (a) in this example), and the treatment of stroke constraints remains the same (d).

L_2 error for div	polar map	local optimal	global optimal
sphere	0.2447	0.2239	**0.1809**
sphere Loop 1	0.1586	0.1054	**0.0361**
sphere Loop 2	0.2742	0.1297	**0.0084**
sphere Loop 3	0.7746	0.2749	**0.0020**
L_2 error for curl	polar map	local optimal	global optimal
sphere	0.2453	0.2251	**0.1823**
sphere Loop 1	0.1563	0.1039	**0.0361**
sphere Loop 2	0.2760	0.1300	**0.0083**
sphere Loop 3	0.7765	0.2752	**0.0020**

L_2 error for E_D	polar map	local optimal	global optimal	[1]	[1] w/ optimal ρ_{ij}
sphere	**2.1906**	2.2470	2.4016	2.4161	2.4153
sphere Loop 1	**0.2306**	0.3613	0.6258	0.6300	0.6298
sphere Loop 2	0.8679	0.3259	**0.1581**	0.1592	0.1591
sphere Loop 3	3.0080	1.0464	**0.0396**	0.0399	0.0398

Table 4.1 **Approximation errors.** Using meshes of increasing resolutions that all interpolate a sphere, we evaluate the L_2 errors for the divergence (top left), curl (top right), and Dirichlet energy operators (bottom) evaluated per triangle using our edge-based approach (via Stokes). The sphere mesh has only 162 vertices, and we refine its connectivity via Loop subdivisions, leading to meshes of 642, 2562, and 10242 vertices. We averaged the errors incurred for 100 random vector fields that are linear combinations of the first 40 vector spherical harmonics, normalized to have unit L_2 norm. The optimal (as-Levi-Civita-as-possible) connection systematically produces the smallest error except for extremely coarse resolutions. We also improve on the L_2 norm produced by [1], even if our optimal vertex-to-vertex angles ρ_{ij} are used in their formulae.

$\int_{\mathcal{T}} K$ (mean/std)	polar map	local optimal	global optimal	[1]	global optimal w/ Stokes
sphere	3.9730/0.923e^{-3}	3.9756/0.623e^{-3}	3.9756/0.619e^{-3}	3.9756/0.620e^{-3}	**4.0000**/0.000e^{-3}
sphere Loop 1	3.9878/0.341e^{-3}	3.9897/0.102e^{-3}	3.9898/0.103e^{-3}	3.9898/0.103e^{-3}	**4.0000**/0.000e^{-3}
sphere Loop 2	3.9948/0.455e^{-3}	3.9970/0.057e^{-3}	3.9970/0.057e^{-3}	3.9970/0.057e^{-3}	**4.0000**/0.000e^{-3}
bunny	3.8857/0.850e^{-2}	3.9082/0.706e^{-2}	3.9192/0.741e^{-2}	3.9193/0.742e^{-2}	**3.9995**/0.000e^{-2}
bunny Loop	3.9610/1.199e^{-2}	3.9860/0.696e^{-2}	3.9880/0.725e^{-2}	3.9875/0.725e^{-2}	**4.0003**/0.000e^{-2}
torus	0.0234/0.913e^{-2}	0.0216 /0.371e^{-2}	0.0207/0.381e^{-2}	0.0207/0.381e^{-2}	**-0.0003**/0.000e^{-2}
torus Loop	-0.0025/0.328e^{-2}	-0.0022/0.059e^{-2}	-0.0013/0.065e^{-2}	-0.0016/0.065e^{-2}	**0.0000**/0.000e^{-2}

Table 4.2 **Approximations of Euler characteristic.** For a pointwise unit vector field $\mathbf{u}$, the difference of antiholomorphic and holomorphic energies is $E_{\mathrm{A}}(\mathbf{u}) - E_{\mathrm{H}}(\mathbf{u}) = \int_{\mathcal{T}} K$ (Eq. 2.5). Using random linear combinations of the 30 lowest vector spherical harmonics, we evaluate the difference of our discrete energies E_A and E_H for 100 vector fields (with normalized coordinates at each vertex), divided by π; we indicate both the mean and the standard deviation of these 100 integrations. On various meshes (of genus 0 and 2), our edge-based evaluations exhibit significantly lower errors than all other area-based estimations, including results from [1].

CHAPTER 5

DISCRETE 2-TENSOR FIELDS ON TRIANGULATIONS

5.1 Introduction and related work

While scalar (rank-0 tensor) and vector (rank-1 tensor) fields have been staples of geometry processing, the use of rank-2 tensor fields has steadily grown over the last decade in applications ranging from non-photorealistic rendering to anisotropic meshing. Unlike their lower rank counterparts, there is currently no convenient way to perform computations with arbitrary 2-tensor fields on triangulations. In this chapter, we present a coordinate-free representation of rank-2 tensors suitable for tensor calculus on triangle meshes. Derived from an orthogonal decomposition of planar 2-tensor fields, our resulting discrete tensors extend the notion of discrete differential forms, and are thus compatible with discrete or finite-element exterior calculus in that they define pairing and inner products of arbitrary forms. Additionally, pervasive discrete geometry processing tools such as the weighted Laplace-Beltrami operator are shown to be special cases of our construction, while new operators such as the covariant derivative of discrete 1-forms emerge.

Analysis and visualization. Visualization of fluid motion is often achieved by analyzing the gradient of the velocity field [94]. This 2-tensor field is often split into an antisymmetric part conveying vorticity, and a symmetric part that can be depicted via streamlines tangent to tensor eigenvectors [95]. A unified analysis of arbitrary 2-tensors was proposed in [96] based on complex eigenvalues and eigenvectors. Zhang et al. [97] used, instead, a geometric decomposition of 2-tensors leveraging the trace operator, which was later illustrated as a combination of streamlines and glyphs [98]. In contrast, our work introduces a new orthogonal decomposition of planar 2-tensor fields compatible with discrete exterior calculus.

Metrics. The metric tensor of a Riemannian surface is, itself, a symmetric tensor that defines the length of, and angle between, tangent vectors. While most geometry processing methods use the canonical metric of a mesh induced by its Euclidean embedding, one can use a set of edge lengths to encode piecewise-Euclidean metrics [99, 100, 101]. However, high degrees of anisotropy might not be representable with pure edge lengths as they might not fulfill the triangle inequality everywhere [76, 102]. Recently, a notion of discrete divergence-free metric tensor in the plane was introduced in [78] (representing stress tensors within masonry structures) through not only edge lengths but also additive weights per vertex—which affect both the discrete Hodge star and the Laplacian operator. This augmented metric was further extended to surface meshes in [103]. Our approach offers a generalization of this divergence-free case to arbitrary rank-2 tensors.

Elasticity. Decompositions of differential 2-tensors such as stress and strain are particularly relevant in elasticity [75]. Arnold et al. [104] proposed tensor subspaces (including divergence-free tensors) that form an exact chain dubbed the *elasticity complex*. This sequence further served as the basis for discretizing planar symmetric 2-tensor (stress) fields through mixed finite elements [105] or non-conforming elements [106]. Extensions to tetrahedral meshes were proposed in [107, 108]. Also in the elasticity context, Kanso et al. [109] expressed 2-tensor fields as quadratic tensor products of Whitney 1-forms. While previous methods require high order finite-element spaces, we discretize arbitrary 2-tensor fields through direct differentiation of piecewise-linear Whitney basis functions, which leads to closed-form expressions for discrete tensor calculus on triangulations.

5.1.1 Notations

We will make use of a few specific notations in this chapter.

Continuous setup. We denote by $\mathcal{M}$ a smooth and compact Riemannian 2-manifold, possibly with boundaries $\partial\mathcal{M}$, and endowed with a metric $\mathfrak{g}$ that provides an inner product on (tangent) vector fields. We also use the notion of k-forms ($k = 0, 1, 2$), along with their respective inner

products $\langle ., . \rangle_k$, and the operators d and $\star$ on these forms [72]. We denote by Δ the Laplace-Beltrami operator on functions. From the metric, one can convert a vector field $\mathbf{v}$ into an equivalent 1-form ω using the flat ($\flat$) operator, i.e., $\mathbf{v}^\flat = \omega$; similarly, a 1-form is converted into its equivalent vector field by the sharp ($\sharp$) operator, i.e., $\omega^\sharp = \mathbf{v}$. We call $\mathcal{T}$ the space of tensor fields of rank $(0, 2)$ on $\mathcal{M}$, i.e., 2-tensors acting on vector fields. We also define a local basis of tensors in a given coordinate frame as:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad J = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

We write the area form of $\mathfrak{g}$ as $\mu_\mathfrak{g} = \sqrt{\det \mathfrak{g}}\, J^t$, and the Hodge star on 1-forms as $\star = \sqrt{\det \mathfrak{g}}\, J \mathfrak{g}^{-1}$, indicating a rotation by $\pi/2$ in the tangent plane when applied to a covector. Finally, the (Frobenius) inner product on 2-tensor fields is:

$$\forall \tau_1, \tau_2 \in \mathcal{T}, \quad \langle \tau_1, \tau_2 \rangle_F = \int_\mathcal{M} \operatorname{tr}(\tau_1^t \, \mathfrak{g}^{-1} \tau_2) \, \mu_\mathfrak{g}, \tag{5.1}$$

where $\operatorname{tr}(\tau) = \tau_{ij} \mathfrak{g}^{ij}$ indicates the trace operator on 2-tensors.

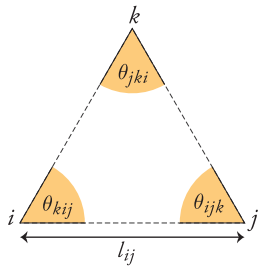


Figure 5.1 **Notations for discrete setup.**

Discrete setup. When dealing with a discrete surface, we use an orientable, compact, and 2-manifold simplicial complex M in $\mathbb{R}^3$, of arbitrary topology (possibly with boundary ∂M). We call $\mathcal{V}$ the set of all its vertices, while the corresponding edge and face sets are denoted by $\mathcal{E}$ and $\mathcal{F}$. Each edge and triangle carries an arbitrary but fixed orientation (index order matters; e.g., ij has the opposite orientation as ji), while vertices have positive orientation by convention. Vertices are given positions $P = \{p_i \in \mathbb{R}^3\}$, which define the surface through linear interpolation over each simplex. The resulting Euclidean measures of edges and triangles are denoted by l_{ij} (length) and

a_{ijk} (area), where the indices refer to the vertex indices, and we assume these to be all nonzero. We denote by θ_{ijk} the angle between edges ij and jk of a triangle ijk (see Fig. 5.1). Discrete k-forms are given as scalars on k-cells [4]. Moreover, we indicate as $\mathbf{d}_0$ the transpose of the incidence matrix of vertices and edges ($|\mathcal{E}|$ rows, $|\mathcal{V}|$ columns), in which each row contains a single $+1$ and -1 for the endpoints of a given edge (the sign being determined from the chosen edge orientation), and zero otherwise; and by $\mathbf{d}_1$ the transpose of the incidence matrix of edges and faces ($|\mathcal{F}|$ rows, $|\mathcal{E}|$ columns), with $+1$ or -1 entries according to the orientation of edges as one moves counter-clockwise around a face. We also use Whitney basis functions for discrete forms [18], indicated as ϕ_i (the usual piecewise linear finite-element function with $\phi_i(p_i) = 1$, $\phi_i(p_j) = 0$) for 0-forms, $\phi_{ij} = \phi_i d\phi_j - \phi_j d\phi_i$ for 1-forms, and $\phi_{ijk} = 2\, d\phi_i \wedge d\phi_j$ for 2-forms. By sharpening 1-form basis functions with the piecewise Euclidean metric, we get the corresponding basis functions for vector fields $\boldsymbol{\phi}_{ij} = \phi_{ij}^{\sharp} = \phi_i \nabla\phi_j - \phi_j \nabla\phi_i$. Hence, our discrete treatment will represent a vector field $\mathbf{u}$ and its associated 1-form $\alpha = \mathbf{u}^{\flat}$ through the same edge values α_{ij}, i.e., through $\mathbf{u} = \sum_{ij} \alpha_{ij} \boldsymbol{\phi}_{ij}$ and $\alpha = \sum_{ij} \alpha_{ij} \phi_{ij}$.

5.2 Tensor Fields over $2D$ Euclidean Space

We now combine the three Berger-Ebin decompositions described in §2.3.3, and derive a new coordinate-free decomposition of 2-tensor fields for the case of a compact region $\mathcal{M}$ in $\mathbb{R}^2$ with boundaries $\partial\mathcal{M}$ and Euclidean metric ($\mathfrak{g} \equiv I$). The continuous picture we present here will be at the core of our discrete approach to deal with 2-tensor fields on arbitrary triangulations. Notice that differential operators simplify considerably for the Euclidean case—e.g., the Hodge-star and the area form reduce to $\star \equiv J$ and $\mu_{\mathfrak{g}} \equiv J^t$, respectively. Yet we keep our original notation in order to discuss extensions and limitations of our results for curved surfaces.

Killing decomposition. Suppose that a 1-form ω is expressed, via Hodge decomposition, as $\omega = df \oplus \star dg \oplus h$, where f and g are scalar functions (df and $\star dg$ represent, respectively, the 1-forms associated to ∇f and $J\nabla g$), and h is a harmonic 1-form. Its Killing operator $\mathcal{K}(\omega)$ can be

decomposed by linearity into terms that are mutually orthogonal with respect to the Frobenius inner product—not only in the plane, but also for surfaces of constant Gaussian curvature as shown in App. B.1:

$$\mathcal{K}(\omega) = \mathcal{K}(df) \oplus \mathcal{K}(\star dg) \oplus \mathcal{K}(h) = \mathrm{Hess}(f) \oplus \mathrm{Tr}^0(g) \oplus \nabla h. \tag{5.2}$$

One can check that $\mathcal{K}(df)$ reduces to the Hessian $\mathrm{Hess}(f) \coloneqq \nabla df$ of the function f, while the term $\mathcal{K}(\star dg)$ is always traceless—we thus denote it as $\mathrm{Tr}^0(g)$. The term $\mathcal{K}(h)$ is simply the covariant derivative ∇h due to the harmonicity of h.

Conjugate Killing decomposition. In a similar fashion, the $\star$-conjugate version of the Killing operator $\overline{\mathcal{K}}(\omega)$ can, itself, be decomposed as:

$$\overline{\mathcal{K}}(\omega) = \star\, \mathrm{Hess}(f)\, \star^{-t} \oplus \star\, \mathrm{Tr}^0(g)\, \star^{-t} \oplus \star\, \nabla h\, \star^{-t}\,. \tag{5.3}$$

We note here that the traceless and harmonic terms in the plane are invariant by $\star$-conjugation; i.e., $\mathrm{Tr}^0(g) = \star\, \mathrm{Tr}^0(g) \star^{-t}$ and $\nabla h = \star \nabla h \star^{-t}$.

Complete decomposition. Using Eqs. (2.7), (5.2) and (5.3), and recalling that divergence-free tensors can be expressed as $\star$-conjugated Hessians [110], we conclude that an arbitrary 2-tensor field τ in the plane is orthogonally decomposed into antisymmetric, divergence-free, curl-free, traceless, and harmonic parts:

$$\tau = \overbrace{\underbrace{s\,\mu_{\mathfrak{g}}}_{\text{2-form}}}^{\in\mathcal{A}} \oplus \overbrace{\underbrace{\mathrm{Hess}(f)}_{\text{curl free}} \oplus \underbrace{\star\,\mathrm{Hess}(w)\star^{-t}}_{\text{div free}} \oplus \underbrace{\mathrm{Tr}^0(g)}_{\text{traceless}} \oplus \underbrace{\star\,\nabla h \star^{-t}}_{\text{harmonic}}}^{\in\mathcal{S}}, \tag{5.4}$$

where the scalar function s describes the antisymmetric part of the tensor field, and the three scalar functions (f, g and w) plus a harmonic 1-form h encode the space of symmetric tensors. Thereby, we obtain a *complete characterization of planar* 2*-tensor fields*, which only requires coordinate-free scalar functions f, g, w, and s.

We further notice that any constant tensor of the form $aI + bB + cC$ can be expressed as the Hessian of a quadratic function f or w. Similarly, a constant tensor $pJ + bB + cC$ can be associated

to the covariant derivative of the rotated gradient of a quadratic function g. We can thus use the three constant scaling a, b, and c to encode a "mean" symmetric tensor field, and then rewrite Eq. (5.4) in a more concise form:

$$\tau = sJ + aI + bB + cC + \mathcal{K}(df + \star dg) + \overline{\mathcal{K}}(dw + h). \tag{5.5}$$

Separating these constant terms will be shown useful in our treatment of 2-tensors on discrete non-flat surfaces in §5.3.2.

Finally, it bears repeating that Eq. (5.4) is only valid for the Euclidean metric ($\mathfrak{g} \equiv I$). Moreover, while the Berger-Ebin decompositions presented in §2.3.3 are valid for any smooth manifold, we show in App. B.1 that the various parts of the Killing operator in Eq. (5.2) are mutually orthogonal only on surfaces of constant Gaussian curvature. To our knowledge, there is no known general coordinate-free decomposition of 2-tensors on arbitrary surfaces.

5.3 Tensor Fields over Triangulations

We now leverage our continuous decomposition of 2-tensor fields in the plane to represent discrete 2-tensor fields on arbitrary triangle meshes via *local, discrete* 0- and 1-*forms*.

5.3.1 Discrete antisymmetric 2-tensors

Differential forms are known to be conveniently discretized using the concept of *cochains* defined in Algebraic Topology [111], and can be interpolated through Whitney form bases [18]. The resulting discrete differential forms [4] and their most relevant operators [65] are well documented by now. In particular, Hodge decomposition of arbitrary forms carries very neatly into the discrete realm in a coordinate-free fashion. The case of discrete 2-forms is a particularly simple subset of this discrete theory: a 2-form μ as used in Eq. (2.7) is simply encoded as its integration over each face ijk

$$\mu_{ijk} = \int_{ijk} \mu. \tag{5.6}$$

This is equivalent to storing a scalar s_{ijk} per face as a discretization of the antisymmetric part in Eq. (5.5), with $s_{ijk} = \mu_{ijk}/a_{ijk}$. Note that this value can be further decomposed using the Laplacian of a dual 0-form q and a constant 2-form p as indicated in §2.3.1. From this set of scalar-per-face μ_{ijk}, a discrete differential 2-form can be interpolated through face-based Whitney basis functions as $\sum_{ijk} \mu_{ijk}\, \phi_{ijk}$.

5.3.2 Discrete symmetric 2-tensors

Unlike the antisymmetric case, symmetric tensors are not entities that are directly "integrable", thus a different discretization approach must be adopted. We introduce a finite-dimensional representation of symmetric tensors via a discrete version of Eq. (5.5).

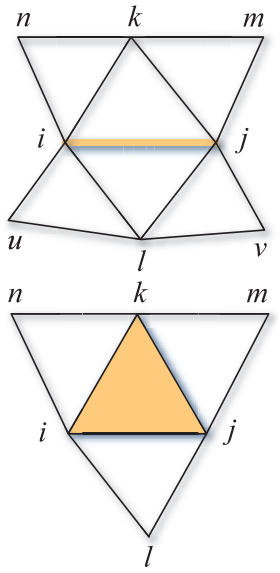


Figure 5.2 **Encoding unit for discrete tensor representation.** We use *one patch per edge* ij defined as the "butterfly stencil" containing the two triangles adjacent to ij and their immediate neighbors (top), as well as *one patch per face* ijk defined as the face and its immediate flaps (bottom).

Encoding. We propose to represent discrete symmetric 2-tensor fields on arbitrary triangle meshes by encoding the forms involved in the decomposition of Eq. (5.5) over small, developable patches (see Fig. 5.2). For each edge-centered and face-centered patch, we store a local approximation of a continuous tensor field as values per oriented simplex (w_i and f_i at nodes, values

g_{ijk} at triangles, and harmonic 1-form values h_{ij} at oriented edges) from which we will be able to derive the symmetric terms of Eq. (5.5). This form-based discretization choice is guided by the usual algebraic topology tools of DEC/FEEC and the resulting discrete Hodge decomposition [4]: in particular, it faithfully discretizes the continuous 1-forms $df + \star dg$ and $dw + h$ as $\mathbf{d}_0 f + \bigstar^{-1} \mathbf{d}_1^t g$ and $\mathbf{d}_0 w + h$, where $\bigstar$ is a discrete Hodge star. Note that we do not need to explicitly encode the constants $a, b,$ and c since, as we discussed in §5.2, they can be incorporated in the scalar fields $f, g, w,$ and s.

Interpolation. As we wish to provide a discrete treatment of 2-tensors that is fully compatible with DEC, the use of Whitney form basis functions [18] is most appropriate: they provide low order, intrinsic interpolation of our discrete forms through $f = \sum_i f_i \phi_i$, $g = \sum_i g_{ijk} \phi_{ijk}$, and $h = \sum_{ij} h_{ij} \phi_{ij}$. With this piecewise continuous reconstruction of forms over each patch, we can formally evaluate the local tensor field approximation for all patches as we describe below. However, our use of piecewise linear basis functions is not amenable to properly capture locally constant 2-tensors: while a quadratic function f has a constant Hessian, a piecewise-linear approximation of f fails to fulfill this property. We thus begin our tensor reconstruction by extracting a local mean tensor per patch, denoted $\overline{\sigma}$, to fully remedy this limitation of low-order form interpolation.

Extracting local mean tensor $\overline{\sigma}$. From f, g, w, h we first evaluate a local constant tensor $\overline{\sigma}$ through local fitting. For each of the 1-forms $\mathbf{d}_0 f$, $\bigstar^{-1} \mathbf{d}_1^t g$, h, and $\mathbf{d}_0 w$ on an edge-based or face-based patch, we find the least-squares fitting linear 1-form over the patch: using a local coordinate system where x is along the oriented edge, we compute the optimal coefficients $\{\nu_i\}_{i=1..6}$ of the 1-form $\nu_1 dx + \nu_2 dy + \nu_3(x dx + y dy) + \nu_4(y dx + x dy) + \nu_5(x dx - y dy) + \nu_6(y dx - x dy)$. The mean tensor $\overline{\sigma} = aI + bB + c\,C$ is then expressed in local coordinates as a sum of contributions $\nu_3 I + \nu_4 B + \nu_5 C$ obtained from each 1-form $\mathbf{d}_0 f$, $\bigstar^{-1} \mathbf{d}_1^t g$, h, and $\mathbf{d}_0 w$. This procedure requires a linear least-squares solve of size 6x6 for both edge- and face-based patches, and is performed on the fly when needed.

Residual edge-based Dirac tensors. The differential terms of Eq. (5.5), representing spatially varying terms around the mean tensor $\overline{\sigma}$, can now be captured within each patch as well. Once we remove from the values of $\mathbf{d}_0 f$, $\bigstar^{-1}\mathbf{d}_1^t g$, $\mathbf{d}_0 w$, and h the linear 1-forms found in the least-squares solution, the residual discrete forms can be formally differentiated to find the corresponding tensor field they encode over each patch. Because of our choice of Whitney basis functions, the resulting residual 2-tensor field turns out to only include edge discontinuities induced by the derivatives of the piecewise linear bases ϕ_i and ϕ_{ij}. The Hessian of f, for instance, is trivially zero everywhere except *across* edges, since the vertex-based basis functions are piecewise linear inside each triangles, and their gradients are discontinuous across an edge. Similarly, the traceless term $\mathrm{Tr}^0(g)$ leads to discontinuities in the piecewise linear 1-form $\bigstar^{-1}\mathbf{d}_1^t g$ across edges. In contrast, the $\star$-conjugate Hessian of w is zero everywhere except *along* edges. We can further compute the tensor term coming from the harmonic 1-form h as a discontinuity along every edge. (Note that the harmonic term could be locally absorbed in the function w due to Poincaré lemma, thereby reducing memory usage if needed; we keep it in our exposition for clarity.) Therefore, our discrete symmetric 2-tensor per patch boils down to a constant tensor $\overline{\sigma}$ plus a sum of impulse tensors on edges expressed as

$$\overline{\sigma} + \sum_{ij} \delta_{ij} \Big[t_{ij}(\mathbf{d}_0 f + \bigstar^{-1}\mathbf{d}_1^t g)\, \mathbf{e}_{ij}^{\perp} \otimes \mathbf{e}_{ij}^{\perp} + t_{ij}(\mathbf{d}_0 w + h)\mathbf{e}_{ij} \otimes \mathbf{e}_{ij} \Big], \tag{5.7}$$

where $\mathbf{e}_{ij}$ is the normalized vector for edge ij, δ_{ij} is the line Dirac function along ij (i.e., $\delta_{ij}(x,y) = \delta(y)$ with δ being the 1D Dirac function and the x-axis being along ij), and t_{ij} is a function linear in its 1-form argument such that, for edge ij between triangles ijk and ijl, $t_{ij} = t_{ij}^k + t_{ji}^l$ with

$$\begin{aligned} t_{ij}^k(\alpha) \;&= \alpha_{ij} \left(\phi_j \cot\theta_{ijk} - \phi_i \cot\theta_{kij} \right) / l_{ij} \\ &\quad + \left(\alpha_{jk}\, \phi_j - \alpha_{ki}\, \phi_i \right) \left(\cot\theta_{kij} + \cot\theta_{ijk} \right) / l_{ij}. \end{aligned} \tag{5.8}$$

These impulse tensors are only well defined in a distributional sense, but they will be systematically integrated against basis functions in §5.4 to obtain weak forms of differential operators.

5.3.3 Discussion

Our proposed discrete encoding can be seen as a generalization of a number of previous approaches. First, our constant term $\overline{\sigma}$ per edge and per face is akin to the conventional piecewise-constant discretization of tensors, typically done per face or vertex—with the major difference that we do not need to define local frames in which components are stored: they are derived from our local, coordinate-free 0- and 1-forms instead. Second, edge-based Dirac tensors were already identified as relevant for triangulated surfaces (see, e.g., [112, 113]); however, they were directly averaged per local neighborhood before being used for differential computations—instead, we keep the Dirac nature of our reconstructed tensors and formally integrate them to derive differential operators on scalars and vector fields. Lastly, having both constant tensors and Dirac edge tensors captures continuous tensor fields better than limiting the finite-dimensional representation to only one of these two parts.

We finally note that our representation is a generalization of the typical encoding of a stress tensor field in planar elastostatics via the Airy function [78], which is the sum of a paraboloid and a function w—corresponding to a constant tensor field plus the rotated Hessian of w to represent a spatially-varying tensor field. This added non-constant term becomes a sum of Dirac impulses for linear basis functions; higher order Whitney elements (see, e.g., [65, 114]) would remove Dirac distributions—at the cost of requiring larger patches.

5.4 Discrete Differential Tensor-based Operators

Our discrete 2-tensor edge-based representation can now be harnessed to define various operators on vector fields and/or 1-forms. For each tensor-based operator, we present its discrete expression for each of the terms in Eq. (5.5). We introduce the edge integration T_{ij}^k of the line Dirac function

t_{ij}^k (Eq. (5.8)) as this term will appear in most expressions:

$$\begin{aligned} T_{ij}^k(\alpha) &= \int_{ijk} \delta_{ij} t_{ij}^k(\alpha) \\ &= \left(\alpha_{jk} \cot \theta_{kij} - \alpha_{ki} \cot \theta_{ijk}\right) \\ &\quad + \tfrac{1}{2} \left(\mathbf{d}_1 \alpha\right)_{ijk} \left(\cot \theta_{ijk} - \cot \theta_{kij}\right), \end{aligned} \tag{5.9}$$

where α denotes a discrete 1-form. Following the convention for t_{ij}, we use $T_{ij}(\alpha) \coloneqq T_{ij}^k(\alpha) + T_{ij}^l(\alpha)$. Observe that, in the case of exact 1-forms $\alpha = \mathbf{d}_0 f$, Eq. (5.9) simplifies to the non-conforming Laplacian [15] of f restricted to ijk. Consequently, the terms T_{ij} return zero for any linear function f. For a boundary edge ij, we set T_{ij} to zero in order to implement Neumann boundary condition.

5.4.1 Discrete generalized Laplacian $\nabla \cdot (\sigma \nabla)$

As mentioned in §2.3.4, the Laplacian operator of functions, commonly used in geometry processing, is a particular case of a general family of differential operators on functions $\Delta^\sigma(z) = div\ (\sigma \nabla z)$ where σ is a symmetric 2-tensor field [68]. We can define its weak (integrated) form on a discrete scalar function $z = \sum_j z_j \phi_j$ as

$$\langle \Delta^\sigma(z), \phi_i \rangle_0 = \sum_{ij} (z_j - z_i) \int_M \sigma(\nabla \phi_i, \nabla \phi_j) \coloneqq \sum_{ij} (z_i - z_j) \mathbf{H}_{ij}^\sigma. \tag{5.10}$$

This generalized Laplacian operator on discrete 0-forms can thus be expressed as a $|\mathcal{V}| \times |\mathcal{V}|$ matrix of the form

$$\Delta^\sigma = \mathbf{d}_0^t \mathbf{H}^\sigma \mathbf{d}_0, \tag{5.11}$$

where $\mathbf{H}^\sigma$ is a diagonal $|\mathcal{E}| \times |\mathcal{E}|$ matrix. Its coefficients $\mathbf{H}_{ij}^\sigma$ can be evaluated *in closed form* for the various types of σ presented in Eq. (5.5) as spelled out in App. B.3.

We point out that $\mathbf{H}^{\mathrm{Id}}$ matches the diagonal Hodge star $\bigstar^{\mathrm{D}}$ [115] traditionally used in mesh processing (we will discuss discrete Hodge star approximations further in §5.4.4). Also, the resulting elliptic operator Δ^{Id} reduces to the usual cotan-Laplacian operator [116]. Moreover, for the case of $\sigma = \star \operatorname{Hess}(w) \star^{-t}$, the matrix $\mathbf{H}^\sigma$ corresponds to the extra terms used in the *weighted Laplacian operator* [117, 118]. Our formulation thus extends this operator to arbitrary 2-tensors, and due to

our deliberate choice to use the conjugated form of the harmonic part in Eq. (5.4), our generalized Laplacian for $\sigma = \nabla h$ verifies the linear precision of its corresponding continuous operator [78].

Ellipticity. In the continuous setting, the generalized Laplacian is (semi-)elliptic iff the symmetric 2-tensor σ is positive (semi-)definite (PSD). A full characterization of ellipticity of our resulting discrete generalized Laplacian is currently unknown, but many sufficient conditions can be (and have been) formulated. First, notice that a traceless 2-tensor cannot be PSD since the sum of its eigenvalues is zero. For the case $\sigma = \mathrm{Hess}(w)$, a simple sufficient condition on the weights was offered in [117], and it remains valid in our setup. For all other cases, one can directly check whether the discrete operator $\mathbf{H}^\sigma$ is positive definite by testing diagonal dominance and non-negativity.

5.4.2 Pairing through discrete tensors

Discrete symmetric tensors can also be used to pair with vector fields. The integral of this pairing, called total pairing, becomes an $|\mathcal{E}| \times |\mathcal{E}|$ operator on edge values since:

$$\langle \alpha, \sigma \beta^\sharp \rangle_1 = \int_M \sigma(\alpha^\#, \beta^\#) = \sum_{ij,kl} \alpha_{ij} \beta_{kl} \int_M \sigma(\boldsymbol{\phi}_{ij}, \boldsymbol{\phi}_{kl}) = \alpha^t \mathbf{M}^\sigma \beta,$$

where α, β are discrete 1-forms corresponding to the vector fields $\alpha^\#, \beta^\#$. The matrix $\mathbf{M}^\sigma$ is typically referred to as the (generalized) *mass matrix*. App. B.4 lists all the closed-form expressions of the pairing matrix $\mathbf{M}^\sigma$ for the terms of our decomposition in Eq. (5.5).

Inner products with symmetric tensors. A particularly important case of pairing through a symmetric 2-tensor σ is the notion of inner product, when σ is positive definite. As in the generalized Laplacian case, necessary and sufficient conditions on the scalar functions f and w to induce a positive definite matrix $\mathbf{M}^\sigma$ are not trivial to formulate. However, a simple check of the diagonal dominance and non-negativeness of $\mathbf{M}^\mathcal{K}$ and $\mathbf{M}^{\overline{\mathcal{K}}}$ provide a straightforward numerical characterization of inner products. Note that the mass matrix for $\sigma \equiv \mathrm{Id}$ is the inner product of Whitney 1-form basis functions, dubbed the Galerkin Hodge star $\bigstar^{\mathrm{G}}$ [115].

Cross products with antisymmetric tensors. We finally point out that when σ is purely antisymmetric (corresponding to the case $\sigma = \mu$ in Eq. (B.4)), the total pairing becomes an integrated cross product between the two vector fields weighted by the face values μ_{ijk}.

5.4.3 Trace

The discrete trace can also be defined through a weak form based on our discrete tensors, resulting in a value per vertex:

$$[\mathbf{tr}\ \sigma]_i = \langle \phi_i, \mathrm{tr}(\sigma) \rangle_0.$$

Discrete antisymmetric tensors thus have zero discrete trace, as in the continuous world. For a discrete symmetric tensor σ equal to the sum in Eq. (5.7), and using $\int_{ijk} \phi_i \phi_j \mu_{\mathfrak{g}} = a_{ijk}/12$ and $\int_{ijk} \phi_i^2 \mu_{\mathfrak{g}} = a_{ijk}/6$, we find:

$$\mathbf{tr}\ \sigma = \mathbf{d}_0^t \mathbf{H}^{\mathrm{Id}} \mathbf{d}_0 w + \mathbf{d}_0^t \mathbf{M}^{\mathrm{Id}} \omega.$$

The first term is the (primal) cotan-Laplacian of w at vertex i. The remainder is elucidated by noting that a discrete 1-form ω is naturally split based on the discrete Hodge decomposition induced by the Galerkin Hodge star ($\mathbf{M}^{\mathrm{Id}} = \bigstar^{\mathrm{G}}$ defined in Eq. (B.4)) as:

$$\omega = \mathbf{d}_0 f + \left(\mathbf{M}^{\mathrm{Id}}\right)^{-1} \mathbf{d}_1^t g + h,$$

where h is closed and coclosed, i.e., $\mathbf{d}_1 h = 0$ and $\mathbf{d}_0^t \mathbf{M}^{\mathrm{Id}} h = 0$. Since we know that $\mathbf{d}_0^t \mathbf{d}_1^t$ is null, the second term simplifies to the cotan-Laplacian of f, while the discrete versions of the traceless terms of the continuous decomposition (Eq. (5.4)) are zero with this discrete trace operator. Thus, the discrete trace recovers the exact same two non-zero terms as its continuous counterpart.

5.4.4 Choice of discrete Hodge stars

The continuous Hodge star can be approximated in the discrete setting in various ways. In our setup where discrete vector fields and 1-forms are expressed using edge values and Whitney basis functions, the most natural discrete Hodge star is arguably the Galerkin Hodge star $\bigstar^{\mathrm{G}} \equiv \mathbf{M}^{\mathrm{Id}}$ (Eq. (B.4)). However, the diagonal Hodge $\bigstar^{\mathrm{D}} \equiv \mathbf{H}^{\mathrm{Id}}$ (Eq. (B.2)) is a sparser alternative often

preferred in graphics as it offers a less computationally intensive approximation—in particular, the discrete Hodge decomposition for this sparse star now only involves diagonal matrices. In fact, these two Hodge stars are well-known to be related in the sense that the primal Laplacian is the same whether Galerkin or diagonal approximation is used [115], i.e., $\mathbf{d}_0^t \bigstar^{\mathrm{D}} \mathbf{d}_0 = \mathbf{d}_0^t \bigstar^{\mathrm{G}} \mathbf{d}_0$. Our extensions of $\bigstar^{\mathrm{D}}$ and $\bigstar^{\mathrm{G}}$ to arbitrary symmetric 2-tensors (resp., $\mathbf{H}^\sigma$ and $\mathbf{M}^\sigma$) precisely maintain this property as we prove in App. B.2. We thus note that one can opt for either one, but the use of the computationally-attractive diagonal approximation loses the traceless property of the terms in g and h described in §5.4.3 since $\mathbf{H}\mathbf{M}^{-1}$ no longer simplifies.

5.5 Applications

We now employ our discrete differential tensor-based operators to derive computational tools for covariant derivative, Lie bracket, and anisotropic geodesics on triangle meshes.

5.5.1 Discrete covariant derivative

The covariant derivative provides a generalization of directional derivatives on arbitrary surfaces. With this concept, one can measure the rate of change of a 1-form α along a vector field $\beta^\sharp$ (asso-

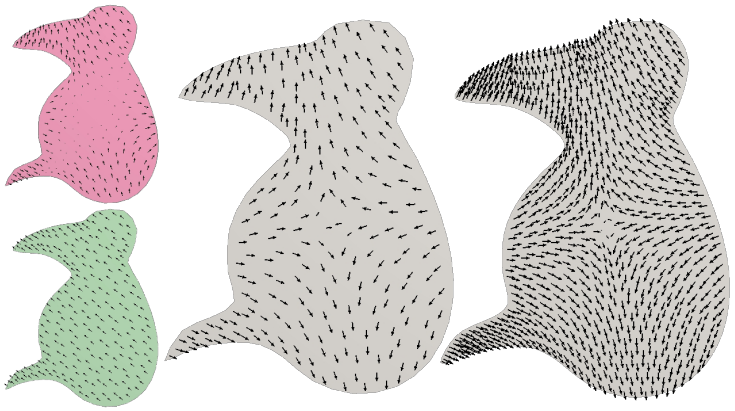


Figure 5.3 **Discrete covariant derivative for planar meshes.** Covariant derivative of a 1-form $\alpha = -2xydx - x^2dy$ (top-left) along $\beta = dx$ (bottom-left) for a planar mesh with concave boundary. Resulting 1-form ω has a numerical residual w.r.t. the analytical solution of 0.7% (center, $|\mathcal{V}| = 173$) and 0.1% (right, $|\mathcal{V}| = 609$), respectively. Vector fields are displayed by interpolating 1-forms at triangle barycenters.

ciated to a 1-form β) as the contraction of $\beta^\sharp$ with the 2-tensor $\nabla\alpha$:

$$\nabla_{\beta^\sharp}\alpha \coloneqq (\nabla\alpha)\beta^\sharp = \omega,$$

where ω is the resulting 1-form. In the discrete realm, we make use of the piecewise linear basis function ϕ_{ij} to evaluate the directional derivative in a weak form as:

$$\forall ij,\ \langle\phi_{ij},\omega\rangle_1 = \langle\phi_{ij},(\nabla\alpha)\beta^\sharp\rangle_1.$$

By leveraging Eq. (2.13) and the mass matrices derived in §5.4.2, we convert this weak formulation into a sparse linear system:

$$\mathbf{M}^{\mathrm{Id}}\omega = \left(\mathbf{M}^{\mathcal{K}(\alpha)} - \tfrac{1}{2}\mathbf{M}^{\mathbf{d}_1\alpha}\right)\beta, \tag{5.12}$$

Therefore, we define the discrete covariant derivative of the 1-form α as the matrix:

$$\nabla\alpha = \left(\mathbf{M}^{\mathrm{Id}}\right)^{-1}\left(\mathbf{M}^{\mathcal{K}(\alpha)} - \tfrac{1}{2}\mathbf{M}^{\mathbf{d}_1\alpha}\right).$$

Note that the mass matrices are computed by combining the mean tensor $\overline{\sigma}$ extracted from α in each patch, the edge-based residual tensor defined by α, and the antisymmetric tensor $\mathbf{d}_1\alpha$. Matrix $\mathbf{M}^{\mathrm{Id}}$ is sparse, positive-definite, and depends only on the triangle mesh, it can thus be efficiently pre-factorized (our implementation uses `eigen` [119]). One can then compute directional derivatives for different 1-forms β through simple sparse matrix-vector multiplication and back substitution. Fig. 5.3 illustrates a directional derivative on a planar mesh with concave boundaries for the case $\alpha = -2xydx - x^2dy$ and $\beta = dx$. The resulting discrete 1-form ω provides an approximation of the analytical solution with a relative error of 0.7% for a coarse mesh and 0.1% for a 4x finer mesh. Fig. 5.4 exemplifies the robustness of the discrete covariant derivative to meshes with variable resolution (even in the presence of obtuse angles), while Fig. 5.5 shows directional derivatives on surfaces of complex shape and non-trivial topology. Finally, Fig. 5.9 presents convergence tests of our results with symmetric and asymmetric tensor fields; we restricted our analysis to a disk, a planar concave mesh and a sphere, since their directional derivatives have known analytical expressions. We also tested the contribution of the impulse tensors in Eq. (5.7) versus using simply the mean tensor $\overline{\sigma}$ per patch in our computations, and observed a systematic decrease in the relative residual from 2% to 27% depending on the tensor fields.

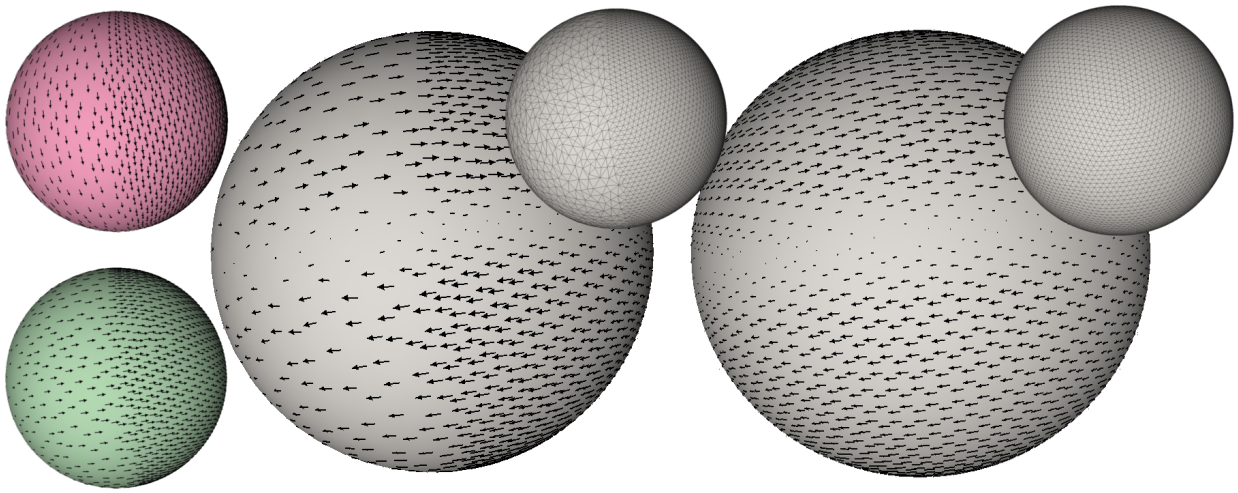


Figure 5.4 **Discrete covariant derivative for sphere meshes.** For 1-forms $\alpha = \sin(\theta)d\theta$ (top left) and $\beta = \sin(\theta)d\phi$ (bottom left) (expressed in spherical coordinates), our discrete covariant derivative $\omega = \nabla_{\beta^\sharp}\alpha$ on an irregular mesh (center) is consistent with the result on a uniform mesh (right) (meshes shown as insets). Vector fields displayed by interpolating 1-forms at barycenters of a subset of triangles.

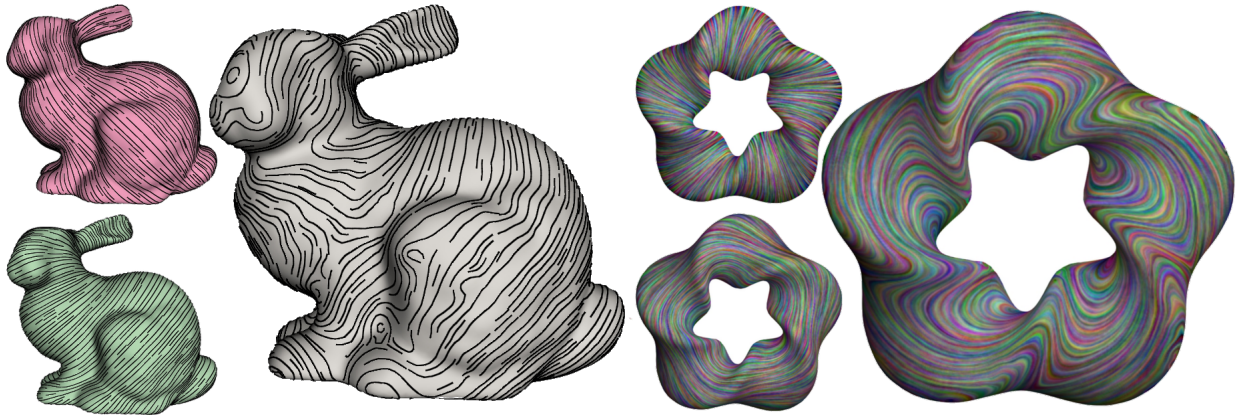


Figure 5.5 **Discrete covariant derivative on meshes of arbitrary shape and topology.** We chose α (top) and β (bottom) as the smoothest 1-forms from the 1-form Laplacian [4].Central images show the resulting 1-form $\nabla_{\beta^\sharp}\alpha$, visualized with sampled integral curves (bunny) and line integral convolution [5] (twisted torus).

5.5.2 Discrete Lie bracket

Our discrete treatment of the covariant derivative also leads to a Lie bracket (also known as the commutator) of vector-fields. By flattening vector fields to 1-forms, the Lie bracket of two 1-forms

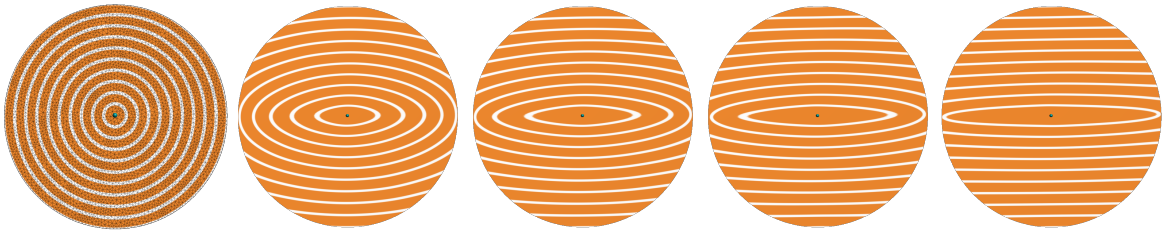


Figure 5.6 **Anisotropic geodesics on planar meshes.** Our tensor-based discrete differential operators can be used to compute anisotropic geodesics. We tested our method on a disk with constant tensors of various anisotropy ratio (from left to right: 1, 0.5, 0.3, 0.2, and 0.1), with the larger magnitude along the x-axis. Notice that the iso-levels stretch to ellipses with the anisotropy as expected.

α and β returns a 1-form γ evaluated as:

$$[\alpha, \beta] := \nabla_{\beta^\sharp} \alpha - \nabla_{\alpha^\sharp} \beta = \gamma.$$

From this definition, we directly reuse Eq. (5.12) and compute the discrete Lie bracket 1-form γ through a similar sparse linear system. Fig. 5.7 validates our discrete Lie bracket on the "torus example" proposed in [6].

5.5.3 Anisotropic heat method

Discrete 2-tensors are also suitable to compute anisotropic geodesic distances based on a simple extension of the heat method [3]. For geodesics induced by the Euclidean embedding space, the heat method requires two linear system solves involving the cotan-Laplace operator: the first step diffuses an impulse heat function z_0 isotropically for a short time interval ϵ into a function z, while the second step finds the potential ψ whose gradient best approximates the normalized gradient of z under the surface metric. We instead propose to replace the isotropic diffusion in the first step by an anisotropic diffusion [77] induced by a PSD symmetric 2-tensor field σ. We thus compute the function z using the generalized Laplacian operator $\mathbf{H}^\sigma$ (see §5.4.1):

$$\left(\bigstar_0 + \epsilon \, \mathbf{d}_0^t \mathbf{H}^\sigma \mathbf{d}_0 \right) z = \bigstar_0 z_0,$$

where $\bigstar_0$ is the diagonal Hodge-star for 0-forms using vertex areas [4]. We further set the time step ϵ to the square of the averaged edge length measured with respect to the tensor σ. The second

step remains unchanged, solving for the potential ψ based on the cotan-Laplacian matrix $\mathbf{d}_0^t \mathbf{H}^{\mathrm{Id}} \mathbf{d}_0$. For boundary conditions, we used Robin boundary conditions as advocated in [3].

In Fig. 5.6, we demonstrate the accuracy of our method by illustrating isodistances for various anisotropy ratios on a unit disk mesh. We also compute anisotropic distances driven by curvature in Fig. 5.8: we first compute an average shape operator Λ per edge as in [112], and set the mean tensor $\overline{\sigma}$ to $(I+0.1\Lambda^2)^{-1}$, such that distances evolve more slowly in regions of large curvature magnitudes.

5.6 Future Work

Our discrete symmetric 2-tensors and their associated operators on vector fields require further numerical analysis, just like usual operators in geometry processing [120]. We are also investigating a discrete notion of Frobenius inner product consistent to our discrete 2-tensors, with which one can compute Killing vector fields [20] and smoothness energies [1]. Finally, extending our discrete treatment of 2-tensors to tet meshes is another topic left for future work.

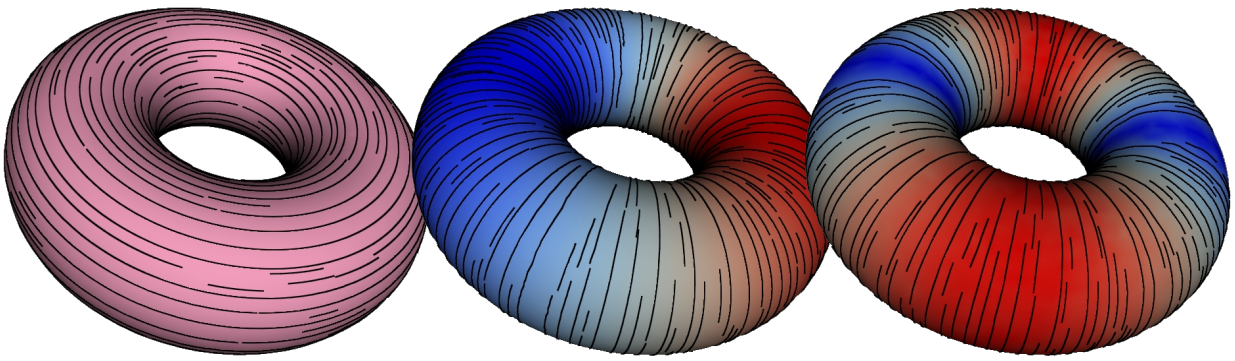


Figure 5.7 **Discrete Lie bracket operator.** Our discrete notion of Lie bracket reproduces the torus example presented in [6] both qualitatively and quantitatively. For two 1-forms α (left) and $z\beta$ (middle), where z is a scaling function, the resulting Lie bracket $\gamma = [\alpha, \beta]$ (right) is parallel to β, as expected in the smooth case. Pseudo-colors indicate the norm of $z\beta$ and γ, respectively.

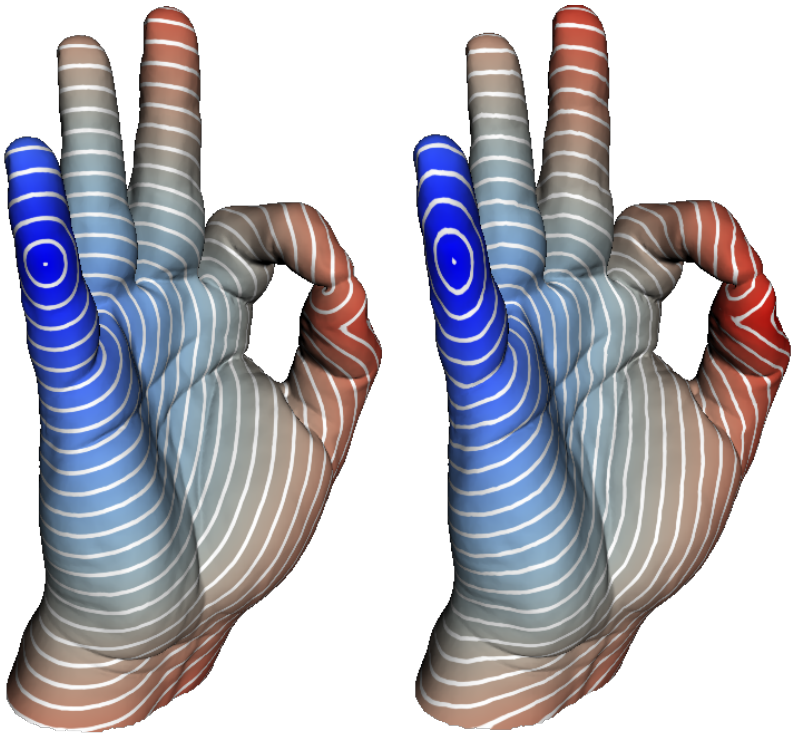


Figure 5.8 **Anisotropic geodesics on surface meshes.** Anisotropic geodesics can be computed guided by the curvature tensor of a surface. Left: isotropic geodesic distance generated by the heat method [3]. Right: anisotropic (curvature-aware) geodesic distance computed with our generalized Laplacian operator (see §5.4.1).

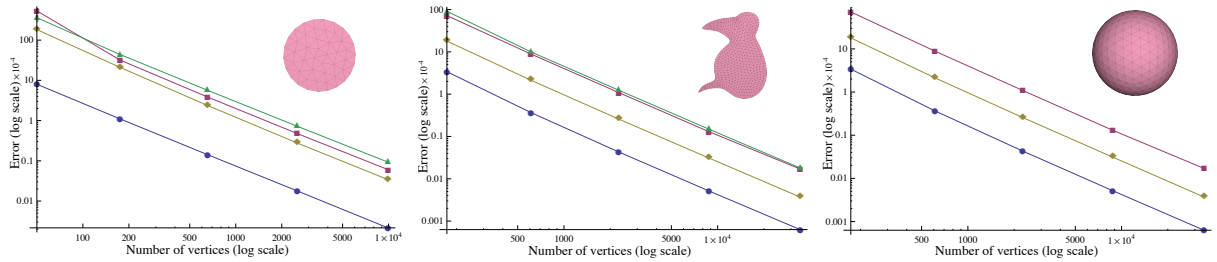


Figure 5.9 **Convergence of the error plot.** Error plot in log-log scale of the residual of the discrete covariant derivative w.r.t. its analytical solution, indicating linear convergence on different meshes. For a disk mesh (left) and a concave shape (middle), we analyzed four scenarios: [in blue] $\alpha = (x-y)dx+(x+y)dy$, $\beta = xdx+ydy$, $\nabla_{\beta^\sharp}\alpha = (x-y)dx+(x+y)dy$; [in pink] $\alpha = -2xydx-x^2dy$, $\beta = dx$, $\nabla_{\beta^\sharp}\alpha = -2(ydx+xdy)$; [in yellow] $\alpha = sin(2x)dx+cos(2y)dy$, $\beta = cos(2x)dx$, $\nabla_{\beta^\sharp}\alpha = 2cos(2x)^2dx$; [in green] $\alpha = x^2dx-2xydy$, $\beta = dx$, $\nabla_{\beta^\sharp}\alpha = 2(xdx-ydy)$. For the sphere (right), we evaluated three cases which are, in spherical coordinates, [in blue] $\alpha = sin(2\theta)d\theta$, $\beta = cos(2\theta)d\theta$, $\nabla_{\beta^\sharp}\alpha = 2cos^2(2\theta)d\theta$; [in pink] $\alpha = sin(2\theta)d\phi$, $\beta = cos(2\theta)d\theta$, $\nabla_{\beta^\sharp}\alpha = 2cos^2(2\theta)d\phi$; [in yellow] $\alpha = sin(2\theta)d\theta+sin(2\theta)d\phi$, $\beta = cos(2\theta)d\theta$, $\nabla_{\beta^\sharp}\alpha = 2cos^2(2\theta)d\theta+2cos^2(2\theta)d\phi$. Errors measured using only the mean tensor $\overline{\sigma}$ (with no edge-based residual tensors) can be up to 27% larger.

CHAPTER 6

SPECTRAL VECTOR CALCULUS FOR VARIATIONAL FLUID SIMULATION

6.1 Introduction

In the previous chapters, we discussed spatial representations of vector fields and tensor fields. Such representations may lead to costly processing procedures for vector fields that are time-dependent, e.g., in fluid simulation. Thus, in this chapter, we focus instead on a drastically different spectral representation of vector fields, the associated operators on such representations, and the resulting efficient update rules in fluid simulation.

While current research of incompressible fluid simulation mainly focuses on achieving realistic visual effects with minimum computation cost, this relentless quest for efficiency has often resulted in time integrators that exhibit large numerical viscosity [36], as they proceed via operator splitting through advection followed by divergence-free projection. The incurred numerical dissipation has also, besides its obvious visual artifacts, the unintended consequence that previews on coarse spatial and temporal resolutions are far from predictive of the final, high-resolution run. Non-dissipative methods have been proposed more recently [40]; however, they require large, non-linear solves, hampering efficiency. On the other hand, model-reduced integrators [121] manipulate a smaller set of degrees of freedom found via Galerkin dimension reduction to capture the main components of the flow efficiently, at the cost of excessive vorticity smearing. Similarly, regular spatial grids are often preferred due to their significantly lighter data structures and sparser stencils—yet, their use conflicts with the proper treatment of boundary conditions over complex, non-grid-aligned domains. While pressure projection with subgrid accuracy have been recently proposed on Cartesian grids [44, 7], non-dissipative methods still require boundary-conforming meshes.

In this chapter, leveraging a novel spectral analysis of the DEC style vector field representation, we introduce a variational model-reduced Eulerian fluid solver with sub-grid accuracy which

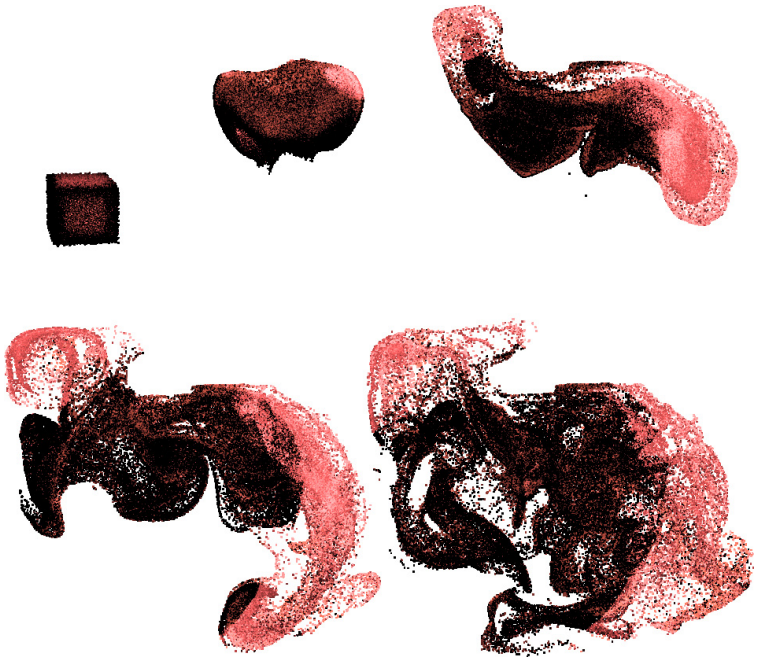


Figure 6.1 **3D bunny buoyancy test**: A hot cube of air initially located at the center of a 3D bunny-shaped domain is advected through buoyancy. Computations were performed using a modified Hodge star on a $42\times42\times32$ grid, with only 100 modes.

bypasses the traditional numerical curses of Galerkin projected dynamics, while keeping the efficiency of Cartesian grid-based simulation. Our contributions are numerical in nature. They do not target improvements in visual complexity, but in efficiency (through embedding of arbitrary boundaries on Cartesian grids, §6.3.4), generality (arbitrary reduced bases can be employed, §6.3.6), and controllability (energy cascading and viscosity are consistent across temporal and spatial scales, §6.3.5).

6.1.1 Related work for model-reduction methods

While most Eulerian methods use a finite-dimensional description of the fluid using DOFs on cell faces or centers, model reduction was introduced in an effort to approximate the fluid motion using only a small number of basis functions. The early days of computational fluid dynamics for atmospheric simulation proposed to reduce complexity by discarding high frequencies through the use of a low number of modes (typically, harmonics or spherical harmonics) to describe the vector

field [122, 123], while pseudo-spectral methods leveraged fast conversion between modal coefficients and spatial representation via the Fast Fourier Transform for highly symmetric domains [124]. Dimensionality reduction was first introduced for fluid animation by Treuille et al. [121] through Galerkin projection onto a reduced set of basis functions computed through principal component analysis of a training set of fluid motions. Their method was demonstrated on regular grids, but is generalizable to tetrahedral meshes. A number of works followed, proposing the use of different bases such as Legendre polynomials [125], trigonometric functions [126], or even non-polynomial Galerkin projection [127]; eventually, Laplacian eigenvectors were pointed out by [9] to be particularly appropriate harmonics as they guarantee divergence free flows and facilitate the conversion between vorticity and velocity, while offering a sparse advection operator for symmetric domains. These eigenfunctions also allow easy implementation of viscosity, and eliminate the need for training sets of velocity fields. For simulations involving moving solids, model reduction can also be conducted on a moving grid [128]. The use of cubature, initially proposed to achieve model-reduced simulation of elastic models [129, 130, 131], can speed up re-simulation of fluids in a reduced subspace as well [132]. However, the gain in efficiency of all such model-reduced simulations is often counterbalanced by (at times severe) energy or vorticity dissipation and the need for unstructured meshes to capture complex boundaries.

6.2 Recap of Variational Eulerian Integration

In order to provide fluid simulations with stable long-term behavior across different space or time resolutions, Pavlov et al. [50] introduced a variational integrator for fluids in Eulerian representation by discretizing the fluid motion as a Lie group acting on the space of functions, and formulating the kinetic energy on its Lie algebra. The motion of an incompressible, inviscid fluid is described in the continuous setting by a volume-preserving flow ϕ_t, i.e., a particle which is at a point p at time $t = 0$ will be found at $\phi_t(p)$ after being advected by the flow. The set of all such possible flows is given by the set of volume-preserving maps ϕ_t from the domain to itself. This set having

the structure of an infinite-dimensional Lie group, it was discretized into a finite-dimensional Lie group for computational purposes. Moving from a Lie group to the associated Lie algebra connects the Lie group viewpoint of flows and "functional maps" [47] to the Lie algebra viewpoint of vector fields, as we now briefly review.

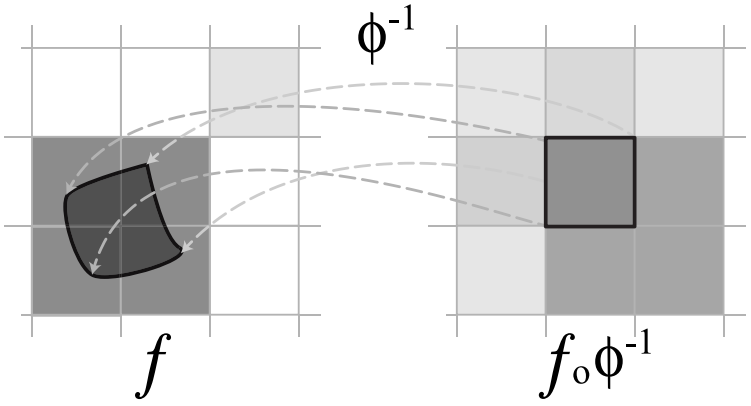


Figure 6.2 **Functional map representation of the fluid flow.** We discretize a continuous function $f(\mathbf{x})$ on our space by taking an average (integrated) value f_i per grid cell i of the mesh, which we arrange in a vector $\mathbf{f}$. This definition of discrete functions allows us to discretize the set of possible flows ϕ_t using a *functional map* (or Koopman operator) $(f \circ \phi_t^{-1})(\mathbf{x}) = f(\phi_t^{-1}(\mathbf{x}))$.

6.2.1 Discretization process

We assume that the fluid domain is discretized as a mesh. Without loss of generality, we restrict our discussion to regular grids for simplicity, as we will show in Sec 6.3.4 how to embed arbitrary domains into a Cartesian grid. In Fig. 6.2 the functional map can be encoded as a matrix q of size the square of the number of cells, representing the action of ϕ_t on any discrete function; that is, the integrated values $\mathbf{f}$ of f per cell become $q\mathbf{f}$ once f is advected by the flow ϕ. Because the discrete flow acts as a functional map, it should always take the constant function to itself. That is, for all q, we require that $q\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ is a vector of ones (see [50] for the equivalent condition on an arbitrary mesh). This is the same as saying that the row sums of q are equal to 1, i.e., q is *signed stochastic*. Since we are simulating an incompressible fluid, we also require that the discrete flow be volume-preserving. This condition is achieved by asking that the discrete flow preserves the

inner product of vectors, that is, q is *orthogonal*, i.e., $q^t = q^{-1}$. Thus, we find that we need to take q to be an element of the Lie group G of orthogonal, signed stochastic matrices. This matrix group represents our discrete fluid configuration, as we describe next.

6.2.2 The Eulerian Lie algebra viewpoint

We can view the finite-dimensional Lie group G as a configuration space: it encodes the space of possible "positions" for the discrete fluid, in that each element of the Lie group represents a possible way that the fluid could have evolved from its initial position. This Lie group represents a Lagrangian perspective as it identifies the fluid particles in a given cell by recording which cells they originally came from. The associated Eulerian perspective is given by the Lie algebra $\mathfrak{g}$ of matrices of the form $\dot{q} \circ q^{-1}$ for $q \in G$. It was shown in [50] that any matrix $A \in \mathfrak{g}$ of this Lie algebra is both *antisymmetric* ($A^t = -A$) and *row-null* ($A\mathbf{1} = \mathbf{0}$), and corresponds to a discrete counterpart of the Lie derivative L_v with respect to the continuous velocity field $v = \dot{\phi} \circ \phi^{-1}$. Thus, the product $A\mathbf{f}$ of such a matrix with a discrete function $\mathbf{f}$ approximates the continuous term $v \cdot \nabla f$. Furthermore, if cells i and j are nearest neighbors, then the matrix element A_{ij} represents the flux of the fluid through the face shared by cells i and j. Thus, an element of the Lie algebra $\mathfrak{g}$ of G is directly linked to the usual flux-based (Marker And Cell) discretization of vector field in fluid simulators [133].

6.2.3 Non-holonomic constraint

Whilst the elements A_{ij} for $A \in \mathfrak{g}$ have a clear physical interpretation in the case where i and j are nearest neighbors, this is not the case for elements representing interactions between cells that are not immediate neighbors. Similarly to a CFL condition, we prohibit fluid particles from skipping to non-neighboring cells, by restricting the Lie algebra to the constrained set $\mathcal{S}$, the set of matrices A such that $A_{ij} = 0$ unless cells i and j share a face (or an edge in 2D). We require the elements of $\mathfrak{g}$ that we use to represent the fluid velocity fields to fall into this constrained set. This has the additional advantage of making the matrices sparse, dramatically decreasing the amount of memory

required and the computational time, as much fewer degrees of freedom need to be updated—and now, the traditional MAC discretization with fluxes corresponds exactly to a Lie algebra element in this constrained set.

Constraining the matrices in this way requires a non-holonomic constraint, because the set $\mathcal{S}$ is not closed under the Lie bracket. That is, interactions between nearest neighbors followed by further interactions between nearest neighbors produce interactions between cells that are two-away from each other, which are therefore not inside the constrained set $\mathcal{S}$.

6.2.4 Creating a variational numerical method

Using this discretization, one can create a variational numerical method for ideal, incompressible fluids through the Euler-Poincaré equations [10] for the Lagrangian given by

$$\mathcal{L}_{\text{Euler}} = \frac{1}{2}\langle A, A\rangle \approx \frac{1}{2}\int v^2 \,\mathrm{d}x, \tag{6.1}$$

and subject to the non-holonomic constraint $A \in \mathcal{S}$. The resulting numerical method exhibits no numerical dissipation, and produces good qualitative behavior over long timescales. Changing the time integration scheme to be time reversible leads to exact energy preservation [40]. With control over dissipation and robustness to time step and grid size, this computational tool greatly facilitates the design of fluid animation. Note that this variational integrator also guarantees that the relabeling symmetry implies a discrete version of Kelvin's circulation theorem, i.e., circulation of velocity field (represented as a Lie algebra element) along a closed loop (represented as a 1-cycle [50]) transported along the fluid flow is invariant, which helps keep the vivid details of vorticity in the fluid simulation without resorting to additional energy-injecting measures such as vorticity confinement, as shown in [19, 40]. However, the time integration requires a quadratic solve based on *all* the fluxes in the domain, making it inappropriate for realtime simulation.

6.3 Model-reduced Variational Integrator

We present an integrator which extends the approach of [50], using a different functional-map Lie group, similarly interlinked with an Eulerian velocity-based Lie algebra picture. Our method offers the additional advantage of fast computations on arbitrary domains: we use *reduced coordinates* to encode the most significant components of the spatial scalar and vector fields, and perform subgrid accurate precomputations on simple *regular grids*. We will focus on Euler equations first, before discussing variants such as Navier-Stokes and magnetohydrodynamics (MHD).

6.3.1 Spectral bases

We first define the discrete, *reduced* scalar and velocity fields on which our functional map Lie group will act. Extending what was advocated in [9], we use the orthonormal bases for 2-forms and 3-forms given by the eigenfunctions of the deRham-Laplacian operators on an arbitrary discrete mesh $\mathcal{M}$. These are calculated using the discrete operators of Discrete Exterior Calculus [134, 65], allowing us to leverage the large literature on their implementation and structure-preserving properties. From this small set of basis functions, we efficiently encode through reduced coordinates the full-space fields typically used in the MAC scheme, i.e., fluxes through cell boundaries (discrete 2-forms) to represent velocity fields, and densities integrated in each cell (discrete 3-forms) to represent scalar fields (such as smoke density or geostrophic momentum in rotating stratified flow [135]).

Choice of bases. We denote the i-th eigenfunction of the 3-form Laplacian Δ_3 as Φ_i, with associated eigenvalue $-\mu_i^2$,

$$\Delta_3 \Phi_i = -\mu_i^2 \Phi_i.$$

The eigenfunctions corresponding to the $M_3{+}1$ *smallest* μ_i can be assembled into a low-frequency basis

$$\{\Phi_0, ..., \Phi_{M_3}\}.$$

Note that depending on the boundary condition, $\mu_0 = 0$ may correspond to more than one harmonic function; but these remain stationary when advected by divergence-free velocity fields with zero flux across the boundary, and are thus omitted in our discussion.

Similarly, we denote the i-th eigenfunction of the 2-form Laplacian Δ_2 as Ψ_i, with its associated eigenvalue $-\kappa_i^2$:

$$\Delta_2 \Psi_i = -\kappa_i^2 \Psi_i.$$

We also assemble the first M_2 eigenvector fields (corresponding to the M_2 smallest κ_i) into a finite dimensional low-frequency basis,

$$\{\Psi_1, ...\Psi_{M_2}\}.$$

Some of the 2-form eigenfunctions are not divergence-free, and these eigenfunctions can be identified as gradient fields, $\nabla\Phi_i/\mu_i$ (see §C.1). Thus, we can reorder the eigenfunctions of Δ_2 into

$$\{h_1, ..., h_{\beta_1}, \frac{\nabla\Phi_1}{\mu_1}, ..., \frac{\nabla\Phi_{M_3}}{\mu_{M_3}}, \Psi_1, ...\Psi_{M_C}\},$$

where h_i are harmonic 2-forms (corresponding to frequency $\kappa_i = 0$) with β_1 being the first Betti number determined by the topology of the domain (basically, the number of tunnels plus the number of connected components of the boundary minus one), and $M_C = M_2 - M_3 - \beta_1$ denoting the number of non-harmonic but divergence-free basis functions.

Discretization. Computing our spectral bases requires a proper discretization of the Laplacian operators and of boundary conditions. Both topics are well studied, and many implementations can be leveraged [57, 136]. In particular, we note that discrete Laplacians are typically *integrated* Laplacians, meaning that the two eigenvalue problems mentioned above are discretized as two generalized eigenvalue problems

$$(\star_3 \Delta_3)\Phi_i = -\mu_i^2 \star_3 \Phi_i \quad \text{and} \quad (\star_2 \Delta_2)\Psi_i = -\kappa_i^2 \star_2 \Psi_i$$

respectively, to make the discrete operators symmetric and thus allow for efficient numerical solvers. We provide a detailed guide to discretization on *arbitrary unstructured meshes* in §C.1 to explicate

how to enforce no-transfer and free-slip conditions (corresponding, respectively, to $v_n|_{\partial\mathcal{M}}=0$ and $\partial v_t/\partial n\,|_{\partial\mathcal{M}}=0$ if the continuous velocity field is decomposed at the boundary into its normal and tangential components, $v=v_n+v_t$). Note that only two operators are required: the exterior derivative d and the Hodge star $\star$. The first operator is purely topological, while the second is just a scaling operation per edge, face, and cell. Moreover, we will see in §6.3.4 that this latter operator can be trivially modified to handle arbitrary fluid domains without having to use anything else but a regular grid. From these two operators, both Laplacians are easily assembled, and low-frequency eigenfields are found via Lanczos iterations.

6.3.2 Spectral Lie group

While earlier methods [50, 10] have defined scalar fields using a spatial representation through linear combinations of locally-supported piecewise-constant basis functions, we use a spectral representation through linear combinations of the aforementioned spectral basis functions Φ_i, allowing us to drastically reduce the number of degrees of freedom the integrator will have to update, while still conforming to the shape of the domain (see Fig. 6.3).

Lie group. We encode the fluid motion through a time-varying Lie group element $q(t)$ that represents a functional map induced by the fluid flow ϕ_t, mapping a function $f(\mathbf{x})=\sum_i f_i\Phi_i(\mathbf{x})$ linearly to another function $g(\mathbf{x})=\sum_i g_i\Phi_i(\mathbf{x})$ such that $g(\mathbf{x})=f\circ\phi^{-1}(\mathbf{x})$. As the function space is approximated by a finite dimensional space spanned by low-frequency basis functions, q can be encoded by a $(M_3+1)\times(M_3+1)$ matrix. The volume-preserving property of the flow still implies the orthogonality of the matrix, i.e, $q^tq=\mathrm{Id}$. So we are looking for a subgroup of $O(M_3+1)$, or, more accurately, of $SO(M_3+1)$, since we wish to describe gradual changes from the identity. The condition that constant functions are mapped to themselves in this low-frequency Lie group becomes $q_{0i}=\delta_{0i}$ and $q_{i0}=\delta_{i0}$, where δ_{ij} is the Kronecker symbol, since 0-th frequency represents the constant function. This effectively removes one dimension, and the Lie group that we are using is thus isomorphic to $SO(M_3)$. This is much smaller than the full Lie group used for the spatial

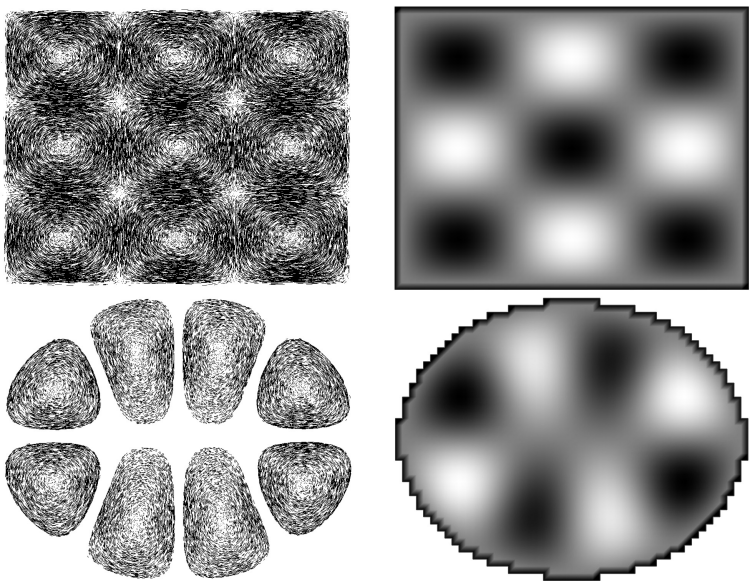


Figure 6.3 **Effect of shape on spectral bases**: The Laplacian eigenvectors depends heavily on the domain Ω. Here, rectangle (top) vs. ellipse (bottom) domains (both computed on 2D rectangular grid of size 60^2) exhibit very different eigenvectors Ψ_{10} and Φ_{10}.

representation [50] which had a dimension proportional to the square of the number of cells of the mesh—a potential reduction of several orders of magnitude.

Lie algebra. We identify each velocity eigenfunction Ψ_m with an element of the Lie algebra of the above Lie group as follows. We take the Lie derivative along the velocity field Ψ_m of a scalar eigenfunction Φ_j, then we project the resulting scalar field onto another scalar eigenfunction Φ_j, producing a matrix A_m for each velocity eigenfunction Ψ_m, with entries

$$A_{m,ij} = \int_{\mathcal{M}} \Phi_i(\Psi_m \cdot \nabla \Phi_j). \tag{6.2}$$

Computing A_m amounts to turning a 3-form into a dual 0-form first with $\star_3$, and then carrying out the integral in the (diamond) volume spanned by each face and its dual edge: this way, the differential of the dual 0-form from Φ_j is multiplied by the 2-form Ψ_m on the face and the average dual 0-form from Φ_i on the face. Notice that we have $\langle A_m, A_n \rangle = \delta_{mn}$ by construction thanks to the basis of Ψ being orthonormal. As in the non-spectral case, the divergence-free condition leads

to the antisymmetry of these matrices since

$$A_{m,ij}+A_{m,ji}=\int_{\mathcal{M}}\Psi_m\cdot\nabla(\Phi_i\Phi_j)=-\int_{\mathcal{M}}\Phi_i\Phi_j\nabla\cdot\Psi_m=0.$$

This is expected, since the Lie algebra $\mathfrak{so}(M_3)$ of $SO(M_3)$ contains only antisymmetric matrices. The Lie algebra has a Lie bracket operator, which is given by the usual matrix commutator $[A_m,A_n]=A_mA_n-A_nA_m$.

Non-holonomic constraint. Just like in [50], not every element of the Lie algebra $\mathfrak{so}(M_3)$ will correspond to a fluid velocity spanned by the eigenfunctions Ψ_m. We force the dynamics on the Lie algebra to remain within the domain of physically-sensible elements using the following non-holonomic constraint, which keeps the velocity within the space spanned by the lowest frequency $M_C+\beta_1$ divergence-free 2-form basis fields:

$$A=\sum_{i=1}^{M_C+\beta_1}v_i\,A_i \tag{6.3}$$

where v_i is a coefficient for A_i representing the modal amplitude of frequency κ_i. This linear condition can thus be seen as an intuitive extension of the one-away spatial constraint on Lie algebra elements used in [50] that we mentioned in §6.2.3.

6.3.3 Spectral variational integrator

The Lagrangian of the fluid motion (i.e., its kinetic energy in the case of Euler fluids) can be written as $\mathcal{L}_{\text{Euler}}=\frac{1}{2}\langle A(t),A(t)\rangle$ as we reviewed in §6.2.4. Thus, the equation of motion can be derived from Hamilton's (least action) principle

$$\int\langle A(t),\delta A\rangle dt=0, \tag{6.4}$$

where $\delta A=\delta(\dot{q}q^{-1})$ is the variation of A induced by variation of q. If we denote $B\equiv\delta qq^{-1}$, one has $\delta A=\delta\dot{q}q^{-1}-\dot{q}q^{-1}\delta qq^{-1}$. Since $\dot{B}=\delta\dot{q}q^{-1}-\delta qq^{-1}\dot{q}q^{-1}$, we find that δA is induced by variations of q only if it satisfies Lin's constraints [10]:

$$\delta A=\dot{B}+[B,A], \tag{6.5}$$

where $B = \sum_i b_i A_i$ is an arbitrary element of the Lie algebra with coordinates $\{b_i\}_i$ in the 2-form basis. Substituting Eq. (6.5) into Eq. (6.4), we then have

$$\begin{aligned} 0 &= \int \langle A, \delta A \rangle \, dt \\ &= \int \sum_{i,k} v_i \dot{b}_k \langle A_i, A_k \rangle + \sum_{i,j,k} v_i v_j b_k \left\langle A_i, [A_k, A_j] \right\rangle \, dt \\ &= \int \sum_k \left(- \sum_i \dot{v}_i \langle A_i, A_k \rangle + \sum_{i,j} v_i v_j \left\langle A_i, [A_k, A_j] \right\rangle \right) b_k \, dt. \end{aligned}$$

Since this last equation must be valid for any b_k, the update rule for the velocity field has to be:

$$\dot{v}_k = \sum_i \dot{v}_i \langle A_i, A_k \rangle = \sum_{i,j} v_i v_j \left\langle A_i, [A_k, A_j] \right\rangle \equiv \mathbf{v}^t \mathbf{C}_k \mathbf{v}, \tag{6.6}$$

where $\mathbf{v}$ is the column vector storing the coefficients v_i of the discrete velocity A (Eq. (6.3)), and $\mathbf{C}_k$ is the square matrix with

$$C_{k,ij} = \left\langle A_i, [A_k, A_j] \right\rangle = \int_{\mathcal{M}} (\nabla \times \Psi_i) \cdot (\Psi_k \times \Psi_j). \tag{6.7}$$

Note that this velocity update do not even require the scalar (3-form) bases used in the definition of the Lie group; however, these bases become important in more general simulations, including magnetohydrodynamics and rotating stratified flows.

Time integrator. The continuous-time update in Eq. (6.6) is then discretized via either a midpoint rule (which will lead to an *energy-preserving* model-reduced variant of [40]) or a trapezoidal rule (which corresponds to a model-reduced variant of the variational method of [50]). Specifically, the midpoint rule is implemented as

$$v_k^{t+h} - v_k^t = h \sum_{i,j} C_{k,ij} \frac{v_i^t + v_i^{t+h}}{2} \frac{v_j^t + v_j^{t+h}}{2}, \tag{6.8}$$

The energy preservation can be easily verified by multiplying $v_k^t + v_k^{t+h}$ on both sides of the above equation, summing over k, and invoking the property of coefficients $C_{k,ij} = -C_{j,ik}$. The trapezoidal rule can, instead, be implemented as

$$v_k^{t+h} - v_k^t = \frac{h}{2} \sum_{i,j} C_{k,ij} (v_i^t v_j^t + v_i^{t+h} v_j^{t+h}), \tag{6.9}$$

which is derived from a temporal discretization of the action with variation of $(\delta q)q^{-1}$ for q along the path to be in the restricted Lie algebra set (to enforce Lin constraints), see App. C.4. Both the energy-preserving and trapezoidal variational rules are time-reversible implicit methods solved through a simple quadratic set of equations with a small number of variables. An explicit forward Euler integration can also be used for small time steps in order to further reduce computational complexity; no guarantee of good behavior over long periods of time is available in this case.

Discussion. Our structural coefficients $\mathbf{C}_k$ (which can be precomputed once the spectral bases are found) are similar to the advection terms mentioned in [9]. However, there are some important differences. Although both expressions converge to the same continuous limit, our variational approach produces coefficients that are exactly antisymmetric in j and k as the Lie bracket is antisymmetric, making our method energy-preserving *without* the artifact-prone energy renormalization step advocated in their work. We also note that the symmetry mentioned in their discretization (specifically, $\kappa_j^2 C_{k,ij} = -\kappa_i^2 C_{k,ji}$) is, in fact, *only* valid in 2D as we explain in §C.2. Moreover, our variational integrator also admits a spectral version of Kelvin's theorem as detailed in §C.3. Finally, our approach is quite different from Azencot et al. [47] even though they, too, use an operator representation of vector fields. Because they explicitly leverage the scalar nature of vorticity in 2D, their work cannot be generalized to 3D. Additionally, their representation of vector (resp., vorticity) fields relies on spatial, piecewise-linear (resp., piecewise constant) basis functions instead of using a reduced set of basis functions.

6.3.4 Embedding complex domains on Cartesian grids

Unstructured meshes can be made to conform to arbitrary domains, and the construction of Laplacians on simplicial meshes is well documented (see §C.1). Therefore, one could use our approach on simplicial meshes directly (see Fig. 6.4 for an example on a non-flat triangle mesh). However, Cartesian (regular) grids always generate *much simpler* data structures and *sparser stencils* for the structural coefficients, so sticking to Cartesian grids is key when efficiency is paramount. Yet,

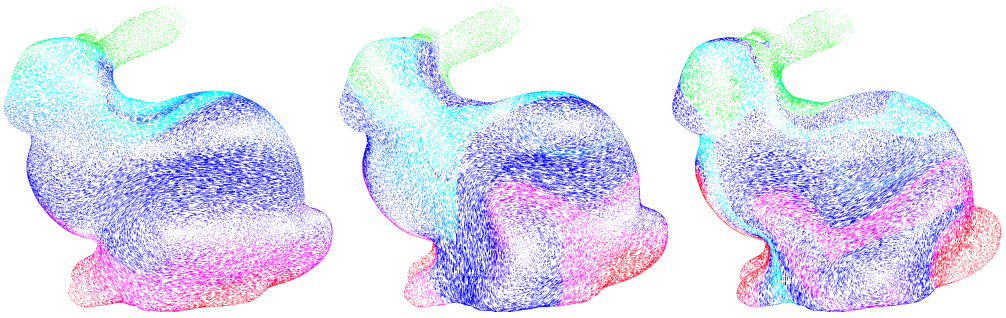


Figure 6.4 **Curved domain**: While all other figures were achieved on a regular grid, our approach applies to arbitrary domains, here on the *surface* of a triangulated domain; a simple laminar flow with initial horizontal velocity smoothly varying along the vertical direction quickly develops vortical structures on this complex surface.

model-reduced fluid methods cannot easily deal with complex domains using only a regular grid to embed it in.

We propose a simple extension of [7] to compute k-form Laplacians of an arbitrary domain, *still using a regular grid*. This renders the implementation of Laplacians and their boundary conditions quite trivial, and removes the arduous task of tetrahedralizing arbitrary domains. This idea was introduced in [44] for their pressure-based projection, and a simple alteration proposed by [7] made the approach robust and convergent. We leverage this latter work by noticing that the modification of the Laplacian Δ_3 that they proposed amounts to a local change to the Hodge star operator $\star_2$.

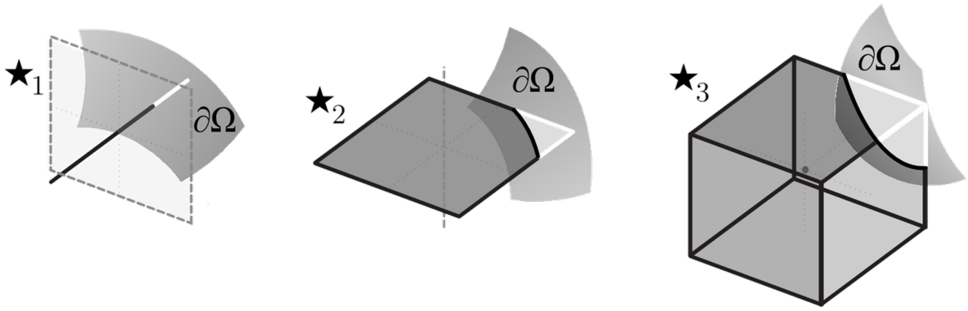


Figure 6.5 **Hodge stars adjusted to the boundary.** The domain Ω is defined implicitly by a function χ via $\Omega = \{\mathbf{x} \mid \chi(\mathbf{x}) \geq 0\}$.

Recall that the diagonal Hodge stars on a mesh $\mathcal{M}$ are all expressed using local ratios of measurements (edge lengths, face areas, cell volumes) on both the primal elements of $\mathcal{M}$ and its dual elements [134]. The changes to the Laplacian operator Δ_3 that Ng et al. [7] introduced can be

reexpressed by an alteration of the Hodge star $\star_2$ where *each primal area measurement only counts the part of the primal face that is inside* Ω, but dual edge lengths are kept unchanged. We extend this simple observation (which amounts to a local, numerical homogenization to capture sub-grid resolution) by computing modified Hodge stars $\hat{\star}_1, \hat{\star}_2,$ and $\hat{\star}_3$ where only the parts of the primal elements (partial lengths, areas, or volumes) that are within the domain Ω are counted (see Fig. 6.5). Note that changing directly the Hodge stars does not affect the symmetry and positive-definiteness of the Laplacians, and thus incurs no additional cost for our method.

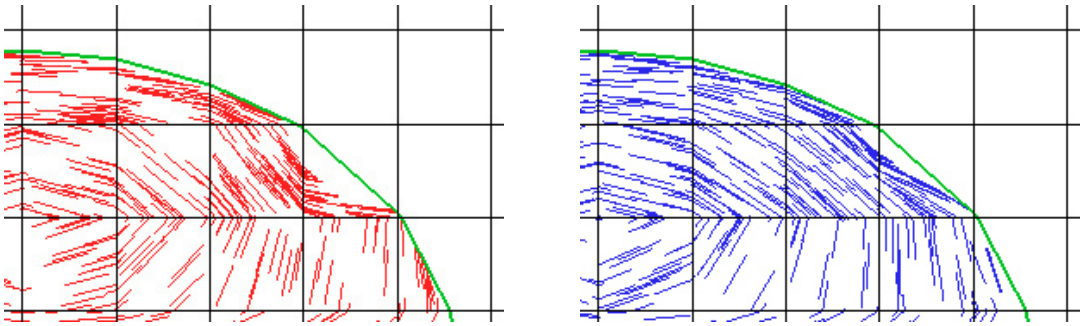


Figure 6.6 **Comparision of the generated eigenvector fields on a coarse grid.** The left figure is generated through a voxelized approximation of the boundary(red) while the right figure is by our Hodge modification(blue).

This straightforward extension allows us to compute our spectral bases on regular grid for arbitrary domains Ω as illustrated in the Fig. 6.6 and in Fig. 6.7 for a basis element of vector fields. We also show the behavior of this Hodge star modification under refinement of the regular grid for a given continuous elliptic domain Ω, resulting in very good approximations of the eigenvectors. Note that the Hodge star operators may involve division by small denominators. A simple and typical thresholding of numbers below the average precision of the floating point representation (i.e., 1e-8) to avoid division by zero is enough, and the eigenfunctions with our specific tangential and normal boundary conditions are computed without any extra preprocessing. In fact, large values of the Hodge stars act as penalty: if a small fraction of the face is inside the domain, then only an accordingly-small flux is allowed.

6.3.5 Variants and extensions

Our approach has been limited to Euler equations so far. However, the use of spectral basis functions, the ability to deal with arbitrary domains on a regular grid, and the functional-map nature of the discretization makes for a very versatile framework in which extensions to the Euler fluid model can be easily incorporated. In all cases, the pseudocode remains identical, as outlined in Algorithm 1.

Algorithm 1 Model-reduced variational integrator

```
Construct 2-form spectral basis with selected frequencies
if scalar fields needed then
    Construct 3-form basis with same selected frequencies
end if
Construct structural coefficients C_{k,ij} with Eq. (6.7)
Initialize simulation
for each time step do
    Time integration through explicit update or Eq. (6.8)/Eq. (6.9)
    If needed, perform scalar advection in current velocity field
end for
```

Viscosity. While the ability to remove spurious energy dissipation is important for the consistency of a numerical integrator with respect to time step size, real fluids exhibit viscosity. Adding viscosity is easily achieved: it corresponds to a dissipation of modal amplitudes by a factor of κ_i^2 since the vector-valued Laplacian is a diagonal matrix in the spectral domain as already leveraged in, e.g., [137]. Thus each modal magnitude v_i evolves as

$$\dot{v}_i = -\nu \kappa_i^2 v_i$$

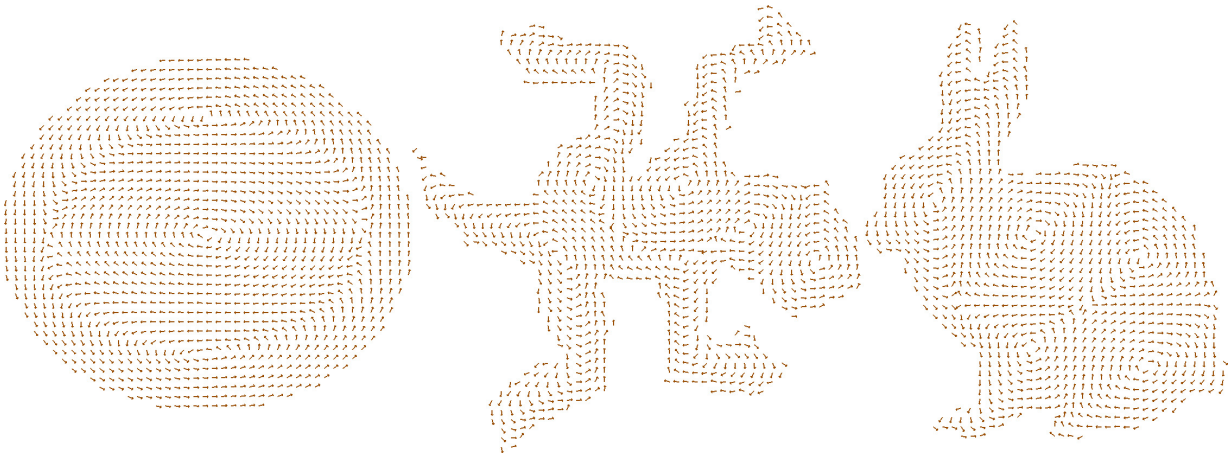


Figure 6.7 **Domain-altered Hodge stars**: Our framework can generate vector bases satisfying prescribed boundary condition for arbitrary domains embedded in a Cartesian grid. Hedge-hog visualization of Ψ_5 on a 256^2 grid for three different 2D domain shapes, obtained through a simple alteration of the Hodge star $\star$ operator.

for a viscosity coefficient of ν. Consequently, Navier-Stokes equations can be handled through an operator-splitting approach by updating the modal magnitudes over each time interval h through

$$v_i^{t+h} \leftarrow e^{-\nu\kappa_i^2 h} v_i^t.$$

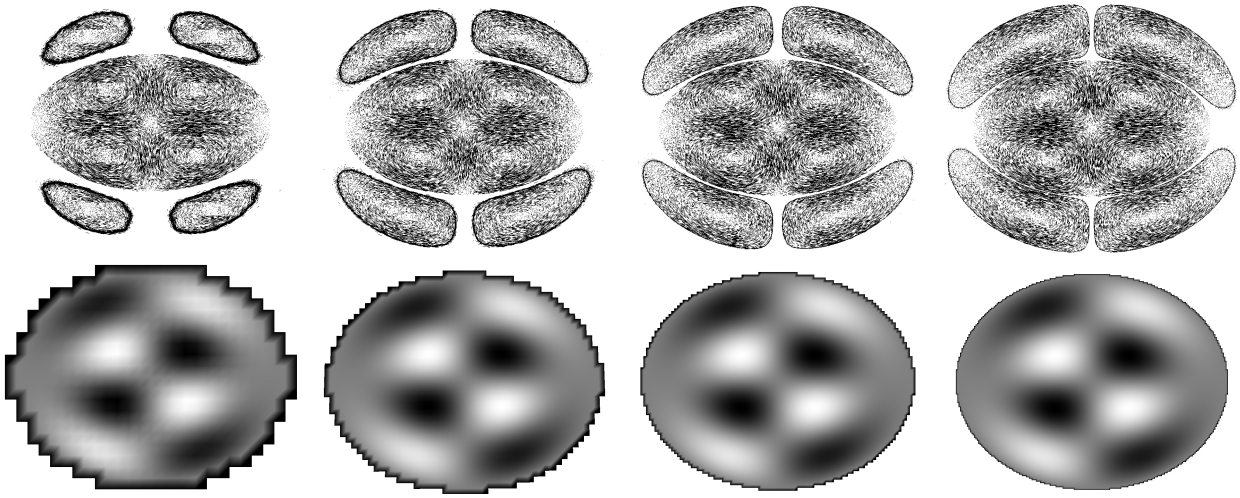


Figure 6.8 **Convergence of Laplacians**: Our discretization of the two Laplacians creates (vector and scalar) eigenfields that converge under refinement of the regular grid used to compute them, extending the linear convergence proved in [7]. Here, particle-tracing visualization of the 15^{th} eigenbasis for vector fields on the ellipse (top) at resolution 30^2, 60^2, 120^2, and 240^2, and 15^{th} eigen function (bottom) at the same resolutions.

Magnetohydrodynamics (MHD). The equations for ideal, incompressible MHD are easily expressed as a modification of the Euler equations. The Lagrangian of MHD is

$$\mathcal{L}_{\text{MHD}} = \frac{1}{2}\langle \mathbf{v}, \mathbf{v} \rangle - \frac{1}{2}\langle \mathbf{F}, \mathbf{F} \rangle$$

in appropriate units [10], where $\mathbf{F}$ is the magnetic field which is advected by the velocity field $\mathbf{v}$. As both $\mathbf{v}$ and $\mathbf{F}$ are divergence-free, they can be discretized with our divergence-free spectral bases. With such discrete velocity and magnetic fields, the update rule in time to simulate the MHD equations closely resembles the Euler fluid case:

$$\dot{v}_k = \mathbf{v}^t \mathbf{C}_k \mathbf{v} - \mathbf{F}^t \mathbf{C}_k \mathbf{F},$$

$$\dot{F}_k = \sum\nolimits_{ij} F_i v_j C_{j,ki},$$

where the second equation performs the advection of $\mathbf{F}$ (encoded in our spectral 2-form basis) in the current velocity v. It is easy to verify that the cross helicity $\int_{\mathcal{M}} \mathbf{v} \cdot \mathbf{F} = \sum_i v_i F_i$ is exactly preserved in our integrator as it is in the continuous world—a numerical property rarely satisfied in existing integrators [10]. The cross helicity is related to the topological linking of the magnetic field and the fluid vorticity. Thus, its preservation prevents spurious changes in the topology of the magnetic field lines over the course of a simulation. This extension to MHD can also be applied to a series of other Euler-Poincaré equations with advected parameters, modeling nematic liquid crystal flows and microstretch continua among others [10]. Note that buoyancy and density fields can be handled in a similar fashion, whether they are coupled with the dynamics or passively advected; they just need to be smooth enough to be captured by low frequencies. Our framework can therefore be used for, e.g., atmospheric simulations as well [135].

Subgrid scale modeling. Direct numerical solvers using Navier Stokes equation require high resolutions to resolve the correct coupling of large and small scale structures. Instead, subgrid scale modeling requires much fewer degrees of freedom to capture the correct large-scale structures by simulating the main effects of the small subgrid scale structures without actually resolving

them. Among the many models that match empirical data well, the LANS-α model (Lagrangian-Averaged-Navier-Stokes, see [8]) advects the velocity in a filtered velocity to better capture the correct energy cascading. Since filtering is achieved through a Laplacian-based Helmholtz operator, we can also use our spectral approach to simulate this model with ease. The kinetic energy (i.e., Lagrangian) is now defined as

$$\mathcal{L}_{\alpha\text{-Euler}} = \int_{\mathcal{M}} \frac{1}{2}\mathbf{v}^2 + \frac{\alpha^2}{2}|\nabla \mathbf{v}|^2.$$

In our spectral bases, the α-model amounts to adding kinetic energy terms scaled by κ_i^2 for the modal amplitudes v_i. Hamilton's principle for the modified Lagrangian leads to essentially the same update rule as Navier-Stokes', except that the structural coefficients are replaced by $(1+\alpha^2\kappa_i^2)C_{k,ij}$. Note that this modification keeps the antisymmetry in j and k intact, and is therefore a trivial alteration of the basic scheme.

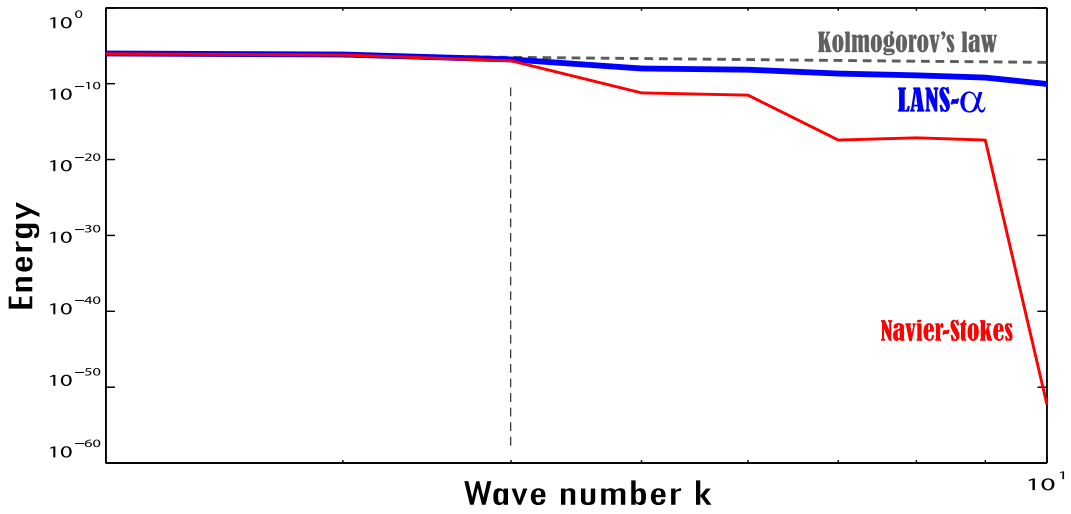


Figure 6.9 **Spectral energy distribution**: With forcing terms keeping the low wave number amplitudes fixed [8], our 3D reduced model applied to the LANS-α model of turbulence produces an average spectral energy distribution (blue) much closer to the expected Kolmogorov distribution (black) than with the usual Navier-Stokes equations (red).

Moving obstacles and external forces. For moving obstacles, instead of calculating additional boundary bases as in [121], we instead follow the simpler solution proposed in [9]: the difference in the normal component of the velocity on the boundary is projected onto the velocity basis and

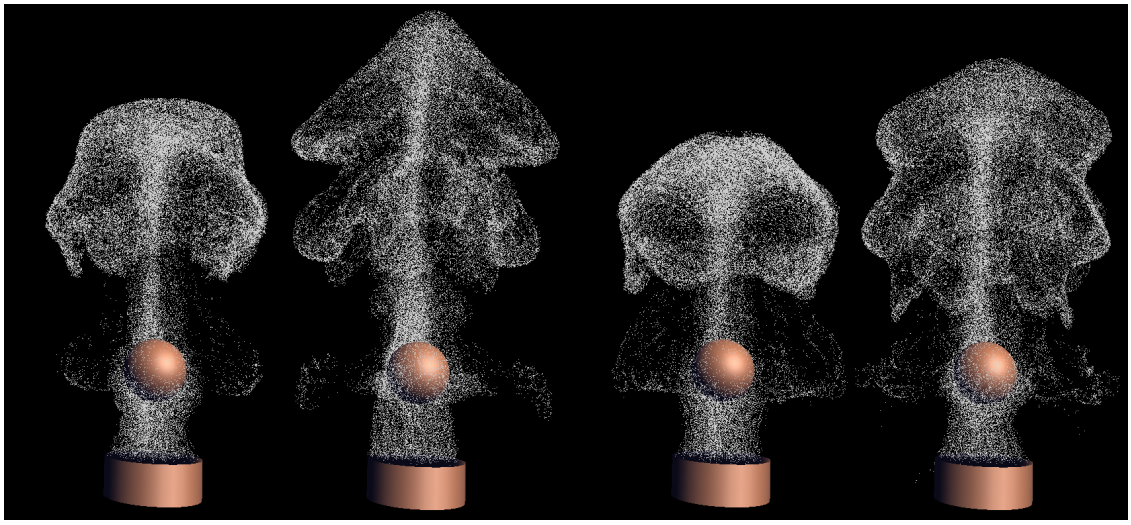


Figure 6.10 **Smoke rising.** Using only 230 modes (about 0.003% of the full spectrum simulation), both [9]'s (left) and our approach already exhibit the expected volutes for a buoyancy-driven flow over a sphere.

subtracted from $\mathbf{v}$, resulting in a low-frequency field roughly satisfying the boundary condition. For any external forces, e.g., buoyancy forces, their effects on the time derivative of the modal amplitude v_i of the i-th frequency are simply calculated by their projection onto Ψ_i. The results are visually correct even on complex shapes, and with minimal computational overhead (see Fig. 6.14).

6.3.6 Generalization to other bases

While we provided detail on the construction of a variational model-reduced integrator for fluid simulation using Laplace eigenvectors, one can easily adapt our approach to arbitrary basis functions, even those extracted from a training set of fluid motions. Suppose that we are given a set of scalar basis elements Φ_i (orthonormalized through the Gram-Schmidt procedure) and a set of velocity basis elements Ψ_i. The Lie derivative matrix A will still be antisymmetric as long as Ψ_i's are divergence-free. This means that one can use existing finite element basis functions instead of our Laplace eigenvectors—or even wavelet bases of $H(\mathrm{div}, \Omega)$ (see, e.g., [138]) if spatially localized basis functions are sought after to get a sparser advection. The key to the numerical benefits of our variational approach is to ensure the anti-commutativity of the Lie bracket in the evaluation of $\langle A_i, [A_j, A_k] \rangle$ (Eq. (6.7)) and the advection of other fields by the exponential map (or approxi-

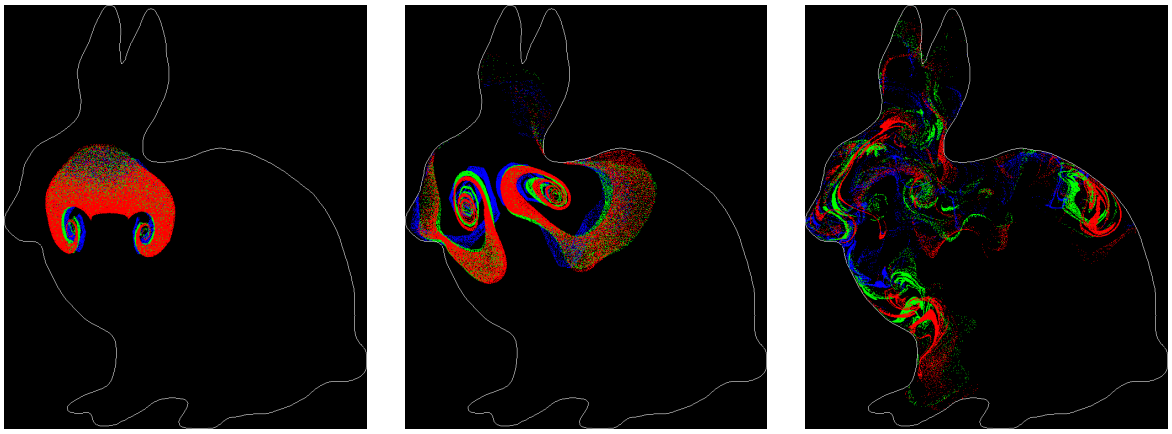


Figure 6.11 **Robustness to resolution**: With the homogenized boundary condition on grids of resolution 40^2 (blue), 80^2 (green), and 160^2 (red), no staircase artifacts are observed, and the simulation results are consistent across resolutions.

mations thereof, see App. C.4) of the matrix representing the Lie derivative as done in MHD and complex fluids [10]. In a way, the original non-spectral variational integrators can be seen as a special case of our framework where Whitney basis functions are used. However, viscosity can no longer be handled as easily in this case as the Laplacian is not diagonal in general bases. Moreover, the required number of degrees of freedom to produce smooth flows may end up being high if the bases are arbitrary.

6.4 Results

Our results were generated on an Intel i7 laptop with 12GB RAM, and visualized using our own particle-tracing and rendering tools.

Reduced vs. full simulation. In order to check the validity of our reduced approach, we performed a stress test in a periodic 2D domain to visualize how the increase in the number of bases used in our spectral integrator impacts the simulation over time. We selected a band-limited initial velocity field at time $t = 0$ that only contains non-zero components for the first 120 frequencies. We then advected the fluid using our integrator, with fluid markers initially set as two colored disks near the center. Because of the propensity of vorticity to go to higher scales, our reduced approach does

not lead to the exact same position of the fluid markers after 12s of simulation if one uses only 120 bases. However, as the number of bases increases to 300 or 500, the simulation quickly captures the same dynamical behavior as a full variational integrator with 256^2 degrees of freedom, see Fig. 6.12. We demonstrate in the accompanying video that our variational integrator also captures the proper qualitative behavior of two merging vortices in contrast to the result using [9] which, instead, generates several vortices (see supplementary video).

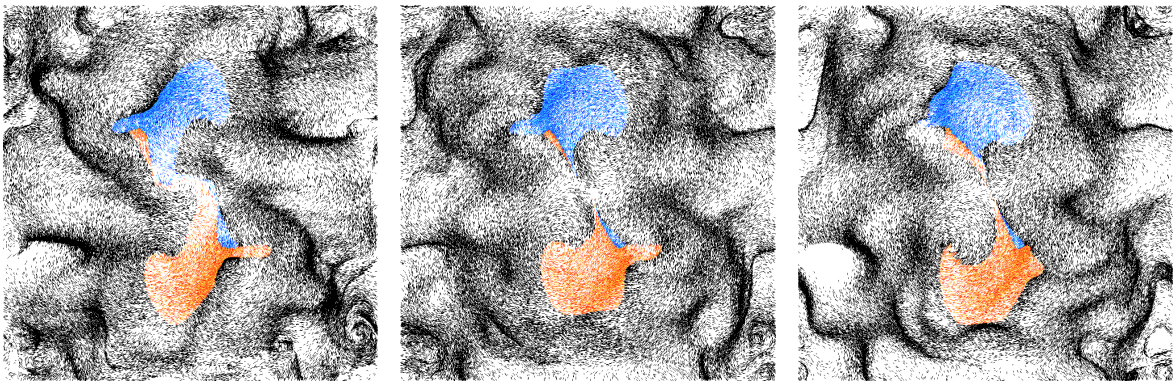


Figure 6.12 **Convergence of simulation**: A flow in a periodic domain is initialized with a band-limited velocity fields with 120 wave number vectors. Fluid markers (forming a blue and red circle) are added for visualization. After 12s of simulation, the results of our reduced approach (left: 120, middle: 300 modes) vs. the full 256^2 dynamics (right) are qualitatively similar.

Arbitrary domains. We also show in Figs. 6.3, 6.7, and 6.11 (2D) and Figs. 1.6, 6.1, 6.10, 6.17, and 6.14 (3D) that the use of boundary conditions embedded on regular grids leads to the expected visual behavior near domain boundaries, eliminating the staircase artifacts of traditional immersed-grid methods. Our homogenized boundary treatment obtains results similar to those of unstructured meshes while using only calculations that are directly performed on regular grids—thus requiring significantly simpler, smaller, and more efficient data structures. As shown in Fig. 6.13, our basis fields converge with second order accuracy, much faster than the boundary condition in [44] (the latter may, in fact, not even converge in some cases as discussed in [7]). Moreover, we also extended this approach not just to scalar field, but vector fields. Our fluid dynamics is also consistent across a wide range of temporal and spatial discretizations, see Fig. 6.11. In addition, the regular grid structure also simplifies the interaction with immersed solid objects as demonstrated in Fig. 6.17

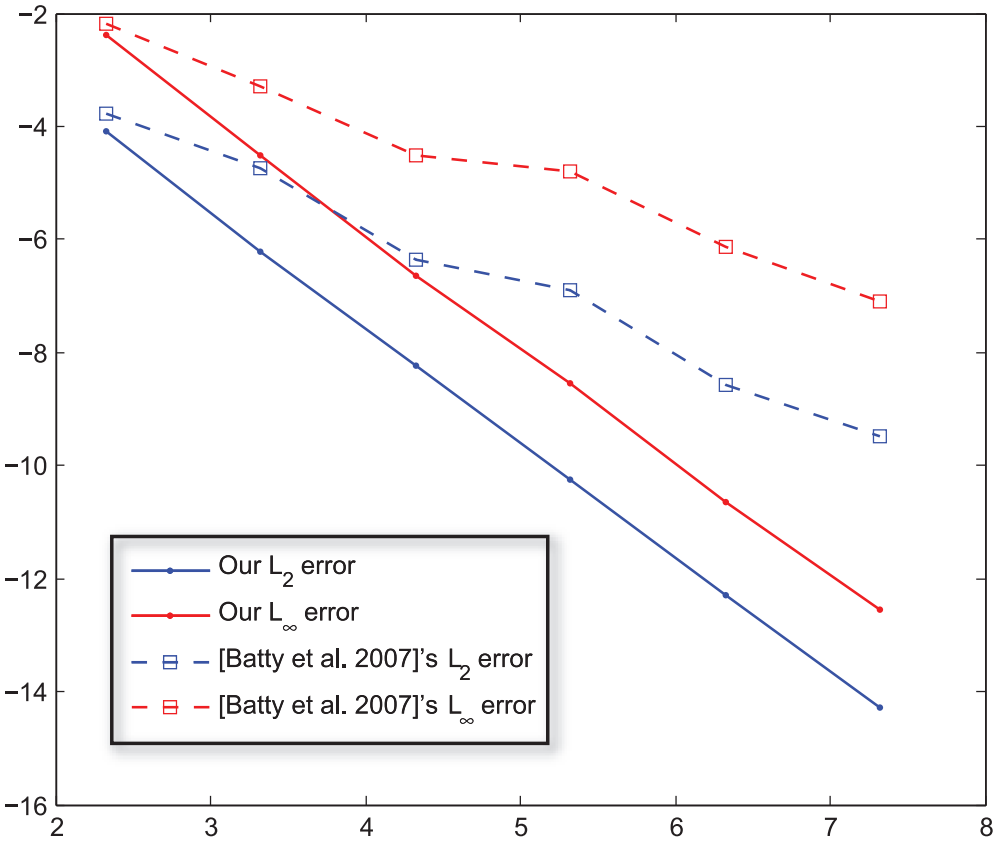


Figure 6.13 **Log(error)-log(resolution) plot for eigenvectors tested on a unit disk.**

through a flow induced by a scripted car turning around a corner; interactive fluid stirring by a paddle manipulated by the user is also easily achieved as shown in Fig. 6.14. We also show in Fig. 6.10 that our method can handle the typical test case of smoke plume past a sphere even at low resolution, and we can incorporate both free-slip or no-slip boundary conditions. Finally, our spectral integrator can be carried out in the same fashion on curved domains as well, since the eigenvectors of the Laplace(-Beltrami) operator are no more difficult to compute on a triangulated surface; Fig. 6.4 shows a simple laminar flow on the *surface* of the bunny model.

Advanced fluid models. We also extended our method to the LANS-α turbulence model to better capture the spectral energy distribution with a small number of modes. On a 3D regular grid, we performed a simulation as described in [8] by holding the low wave number components v_i fixed for $|\kappa_i| < 2$ to act as a forcing term, and running the simulation until $t = 100$. We then extracted

the average spectral energy distribution present between $t = 33$ to $t = 100$. We show in Fig. 6.9 that the Kolmogorov "$-5/3$ law" is much better captured than with the usual Navier-Stokes model, even for the low number of modes used in our spectral context: the α-model produces a decay rate at high wave numbers much steeper than a Navier-Stokes simulation, allowing us to cut off the higher frequencies at a lower threshold without significant deviation from the spectral distribution. This indicates that our approach consisting in a simple scaling of the structural coefficients helps improving fluid simulation on coarse grids.

We also implemented our extension to MHD, and found the expected preservation of cross-helicity and energy. For comparison purposes (see, e.g., [10]), we visualize our results of the typical rotor test with 100 modes in Fig. 6.18. We finally show in Figs. 6.1 and 6.10 that buoyancy forces are also easy to incorporate by adding an upwards force proportional to the local smoke temperature; the curl of this external force is projected onto the 1-form basis functions, and used to update the vorticity.

Computational efficiency. Our use of model reduction via Laplacian eigenbases provides a significantly more efficient alternative to full simulators, obviously. Due to our variational treatment of time integration, we also prevent many shortcomings of the previous reduced models as we ensure consistency of the results over a large spectrum of spatial and temporal discretization rates, and maintain a qualitatively correct behavior even on coarse grids. The efficiency gain compared to the full variational simulation is apparent, be it in 2D, curved 2D, or 3D. For instance, a full-blown 128^2 grid takes around 50s for the variational integrator to update one step (through a Newton solver) in a typical simulation using the trapezoidal rule update, while a 50-mode (resp., 100-mode and 200-mode) simulation with our integrator takes only 0.098s (resp., 0.65s and 2.0s) for complex boundaries (i.e., with dense structural coefficients), and 0.026s (resp., 0.070s, 0.28s) for simple box domains (with sparse coefficients). Our Newton solver normally converges in a couple of iterations depending on the time step size (which determines the quality of the initial guess); for instance, the average in our 3D bunny buoyancy test in Fig. 6.1 is below 3 iterations.

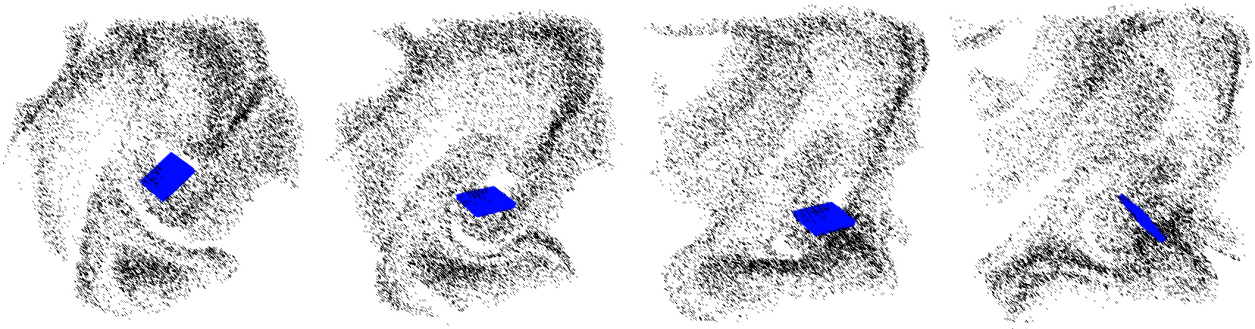


Figure 6.14 **Interactivity**: We can also use the analytic expressions for Ψ_k and $C_{k,ij}$ in a periodic 3D domain to handle a large number of modes directly. The explicit update rule exhibits no artificial damping of the energy as expected, but offers realtime flows.

Selection of frequencies. Depending on how many modes the user is willing to discard (and replace by wavelet noise or dynamical texture for efficiency), the computational gains can be in the order of several orders of magnitude, and this allows us to simulate flows at interactive or realtime rates (see Fig. 6.17, or Fig. 6.14 for an example with a periodic 3D domain where we can compute the eigenbases in closed form). Note however that our model-reduced integrator suffers from the usual limitation of model reduction: the complexity is actually growing quadratically (resp., cubically) with the number of modes for sparse (resp., dense) structural coefficients. So our integrator is numerically efficient only for relatively low mode counts. However, this is exactly the regime for which one can achieve significant computational savings for very little visual degradation. Similar to [9], we also found that, when using very few low frequencies leads to unappealing simulations, adding a few high frequencies (and thus, skipping a large amount of medium frequencies) is enough to render an animation realistic: our integrator can use such a tailored frequency range seamlessly, and the non-linear exchange between low and high frequencies is enough to create much more complex patterns that respect the expected motion of the flow, see Fig. 6.15. Fig 6.10 was also done in this manner: the first 115 modes were used, the next 400 modes were skipped, and we added the next 115 modes to add small scale effects. The use of subgrid scale modeling explained in §6.3.5 is yet another way to make sure that the higher frequencies are properly dealt with and provide a good, visually correct approximation to the fluid equations.

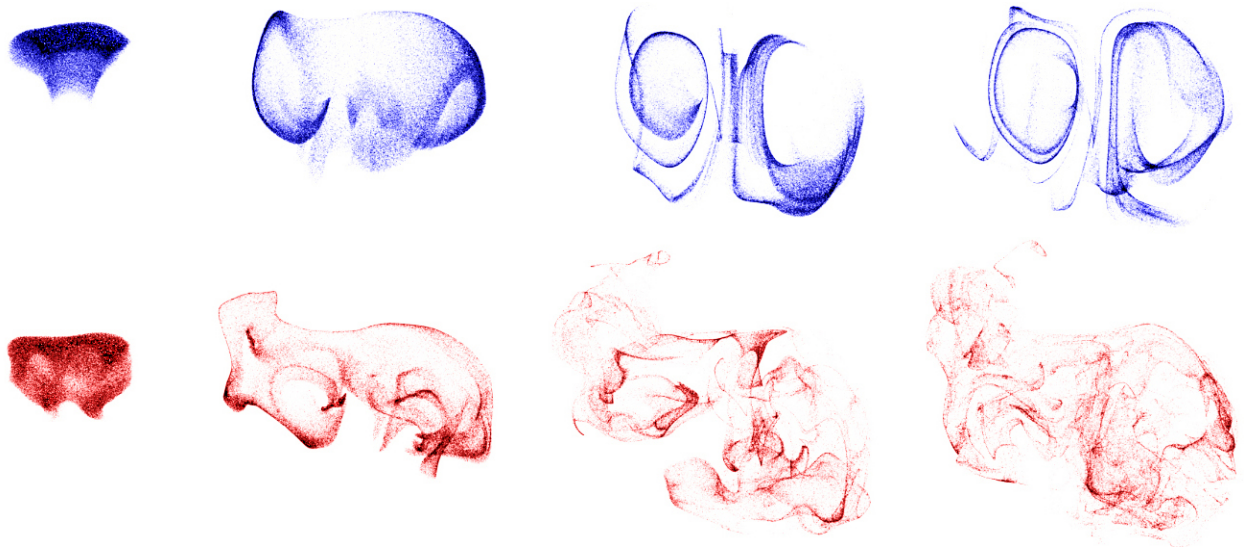


Figure 6.15 **Frequency shaping**: For the same setup as Fig. 6.1, using only the lowest 10 eigenbasis functions for vector fields leads to a very limited motion. However, adding another 10 basis functions of high frequencies creates a much more detailed animation at very little cost, instead of using all the frequencies from low to high.

Time stepping. Finally, an important feature of our model-reduced approach is its ability to handle both explicit and implicit integration. Implicit integration, using the midpoint rule (Eq. (6.8)) or the trapezoidal rule (Eq. (6.9)), come with good numerical guarantees due to the time reversibility. However, explicit integration is also very convenient as it further reduces the time complexity of the simulation. Nevertheless, an explicit integration has very little theoretical guarantees, and should only be used with care.

Quantitative experiments. One way to evaluate a reduced model of fluids is to measure the evolution in time of the error of the velocity field compared to a full-spectrum (spatial) simulation. Using the exact same Laplacian eigenvectors representing only 0.003% of the modes for the 3D simulation of the rising smoke in Fig. 6.10, both our structural coefficients and [9]'s provide a slowly increasing error as shown in Fig. 6.16 (top), with ours showing an improvement of around 20%; the L_∞ shows a more pronounced improvement as well. The same experiment in 2D for 2 vortices exhibits the same trend (Fig. 6.16, bottom), with a more pronounced difference given

that our approach leads to the two vortices merging as in the ground truth simulation, while [9]'s generates multiple vortices (see supplementary video).

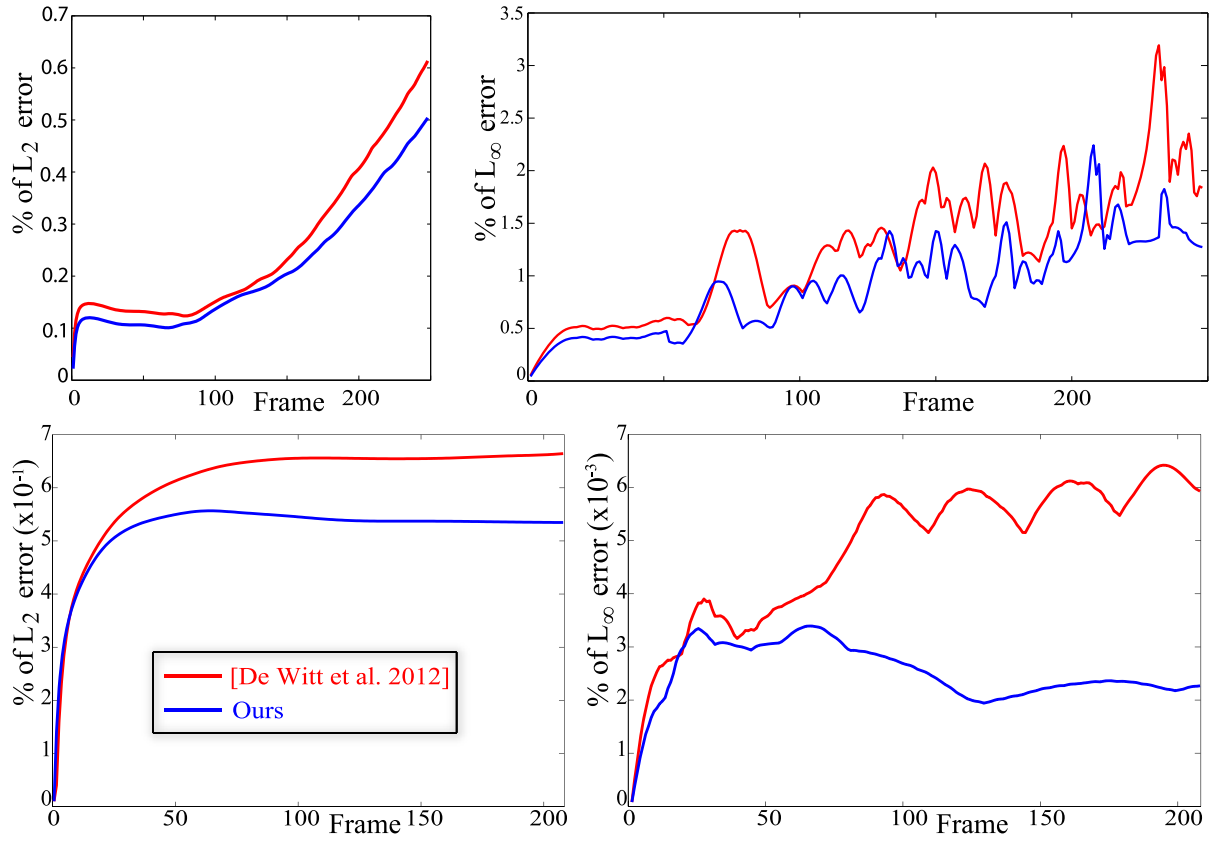


Figure 6.16 **Relative errors.** Relative L_2 (left) and L_∞ errors measured with respect to a full-spectrum (spatial) simulation are systematically improved with our structural coefficients compared to [9], even if the same time integration is used to allow for a fair comparison. Top: errors for the rising smoke example of Fig. 6.10; bottom: errors for two merging vortices (see video).

6.5 Conclusion

We have introduced spectral bases for scalar and vector fields on irregular shapes embedded in regular grids, together with the operators obtained through a simple and novel alteration to offer sub-grid accuracy at no extra cost. Based on the spectral analysis, we introduce a variational integrator for fluid simulation in reduced coordinates. By restricting the variations in Hamilton's principle to a low-dimensional space spanned by low-frequency divergence-free velocity fields, our method exhibits the properties of variational integrators in capturing the qualitatively correct behavior of ideal

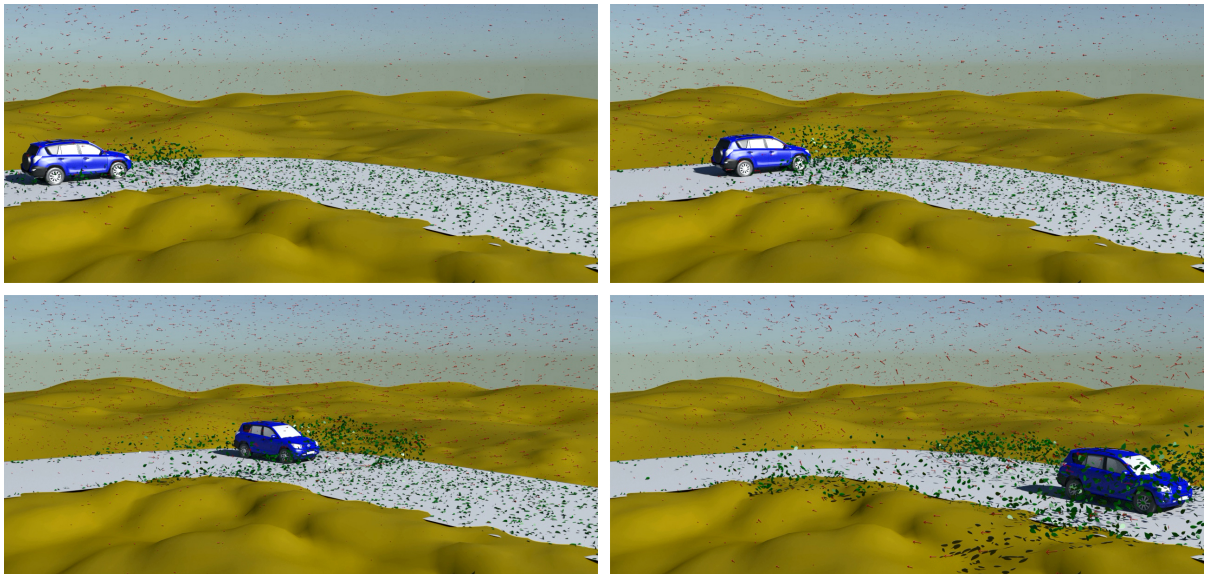


Figure 6.17 **Immersed moving objects.** As the car makes a right turn, the low frequency motion of the air displaced around it lifts the dead leaves. The velocity field above is visualized through arrows.

incompressible fluids (such as Kelvin's circulation and energy preservation) while greatly reducing the computational cost. Finally, we demonstrated the versatility of our integrator by straightforward extensions to moving boundary, magnetohydrodynamics, and turbulence models.

Discussion. Algorithmically, our method resembles all other model-reduced fluid methods. However, it offers a *unified* formulation of model-reduced fluid flows for arbitrary basis functions, and provides structural coefficients without the artifacts of [9]. It only requires a regular grid to encode arbitrary domains with the same convergence rate as [7], but also allows for the computation of vorticity bases. DEC operators for the computation of Laplacians allow full control over boundary conditions, for arbitrary topology; but any other discrete operators can easily be used instead. Our nonlinear update rules enforce a discrete form of Kelvin's theorem and time reversibility—and thus energy preservation. They also offer robustness to time and space discretization rates, an important feature when previewing results. One can also employ explicit integration to further reduce computational complexity. Any application seeking low time complexity of fluid simulation will benefit from our reduced space approach, in particular, real-time interactive simulation or tools for

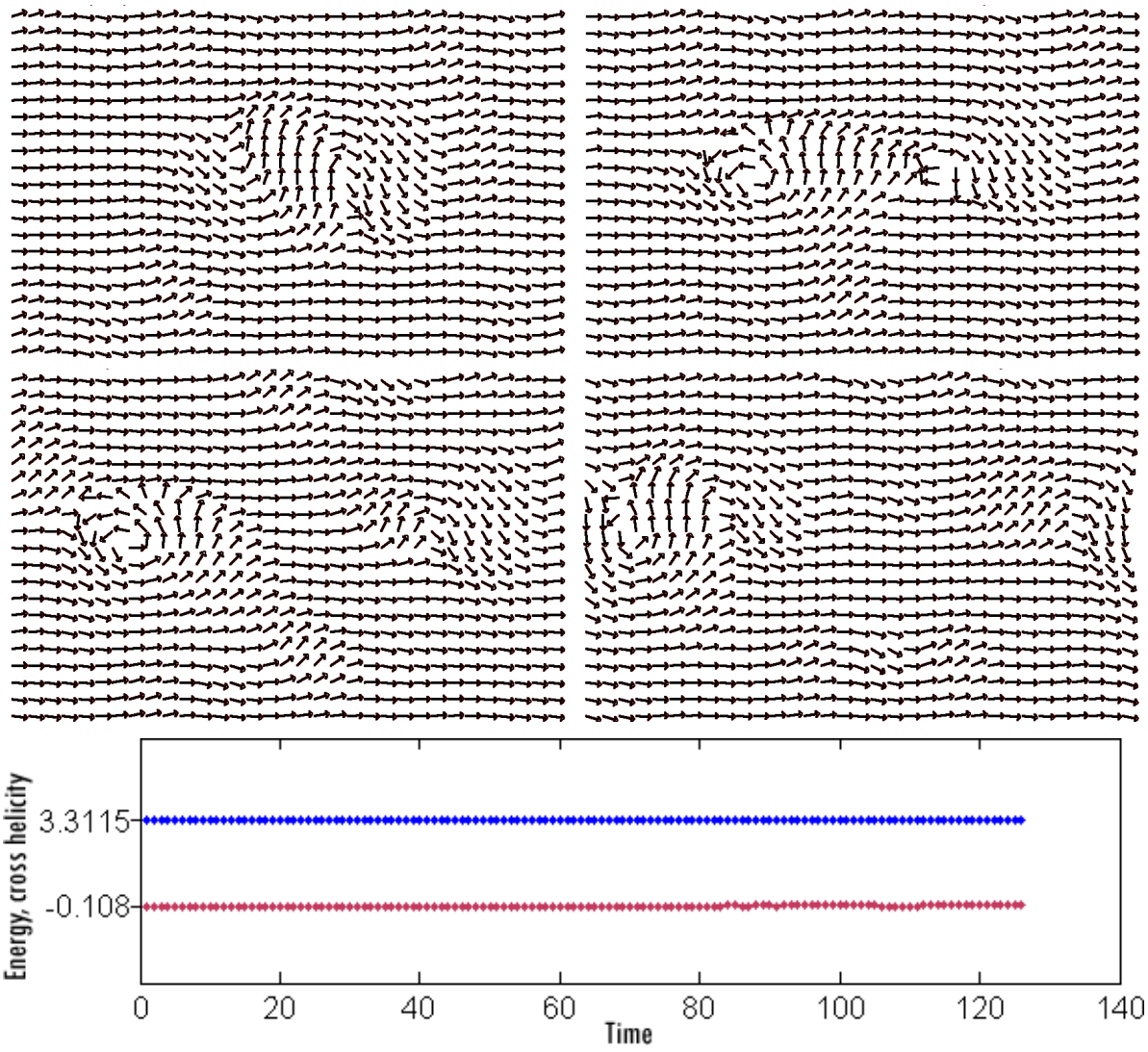


Figure 6.18 **MHD rotor test**: The rotor test for magnetohydrodynamics consists of a dense rotating disk of fluid in an initially uniform magnetic field (left-right, top-middle: $t = 0.042, 0.126, 0.210, 0.336$). Our spectral integrator captures the correct behavior (see full dynamics in [10]) even with only 100 modes. Discrete energy (blue) and cross-helicity (red) are, as predicted, preserved over time (bottom).

designing artist-driven coarse simulation. Even production-quality smoke or fluid animation may be achieved without having recourse to a full resolution simulation through existing post-process curl noise techniques. Moreover, our modified Hodge star can be used in a variety of geometry processing applications where subgrid accuracy on coarse grids is desirable.

CHAPTER 7

SUMMARY AND FUTURE WORK

In this thesis, we present innovative computational tools for vector fields and 2-tensor fields analyses and their applications. Our main contributions are summarized as follows:

- We contributed a theoretical discrete framework for performing covariant derivative (Chapter 4), which is essential to differential calculus on vector fields on triangle meshes and extending the theory of discrete exterior calculus to true vector field analysis. Our notion of discrete connection also leads to a compatible discretization of other first-order derivative operators of vector fields. The resulting computational framework can be applied to n-vector field design, as well as providing the discrete operators numerically superior to previous discretization.

- Our vector field analysis tool is also used in a concrete application, example-based texture synthesis with feature directions aligned to orientation fields (Chapter 3). Our framework provides an intuitive and natural control of the singularities without the need for extra constraints. Spurious singularities are eliminated by incorporating the missing boundary terms into the variational formulation, i.e, by minimizing the Dirichlet energy. Furthermore, we proposed a GPU-friendly seamless synthesis compatible with the orientation field which may contain discontinuity when treated as a vector field. A novel *upside down* mapping style is introduced to enforce the seamless appearance once the discontinuity of the representative vector field is detected.

- Extending coordinate-free representation to rank-2 tensors is achieved by the use of Berger-Ebin decomposition (Chapter 5). We show that an arbitrary 2-tensor (symmetric or non-symmetric) in a plane (or a constant curvature 2-manifold) can be orthogonally decomposed into antisymmetric, divergence-free, curl-free, trace-less, and harmonic parts, each of

which admits coordinate-free representation. This representation facilitates a number of computational tools, including 2-tensor induced covariant derivatives, Lie bracket, anisotropic geodesic calculation. In addition to the practical computation, we believe our representation and operators also provide stepping stones towards a full-blown tensor analysis on simplicial meshes.

- Based on a novel spectral analysis for irregular domains embedded in a Cartesian grid, our computational tool can also be used for incompressible fluid simulation through an innovative model-reduced variational Eulerian integrator (Chapter 6). Our homogenized boundary treatment obtains results similar to those of unstructured meshes and the staircase artifacts of traditional immersed-grid methods are eliminated. Based on such treatment, the scalar and vector valued eigenfunctions of the Laplacian operator can capture the detailed shape without compromising efficiency. The eigenfunctions associated with low frequencies lead to our model-reduced fluid simulator, which achieves realistic animations in significantly less computational time without the numerical viscosity. Consequently, our integrator is robust to coarse spatial and temporal resolutions, providing predictive preview for the final, high-resolution run.

Our research on the vector and 2-tensor analyses and applications offers several directions for future research:

- We have proposed a seamless method for example-based texture synthesis with feature directions adapted for orientation fields. Due to its order-independence, such texture synthesis can be parallelized and thus further accelerate the efficiency. It will be also interesting to explore the effects of combining trivial connection or metric-driven N-RoSy on surfaces with arbitrary topology, generalize the parallel texture synthesis to N-RoSy fields, and implement applications of the method to latent fingerprint enhancement.

- Our constructed smooth discrete connection is interpolated by linear Whitney forms. While

we believe that various applications in geometry processing and even simulation would benefit from a smoother approximation, higher-order connections that still fit our framework could be derived from subdivision-based Whitney forms defined in [114] or from other higher-order Whitney forms—as long as their integrals can be either evaluated in closed form or through quadrature. Similarly, a high order construction for our intrinsic representation of 2-tensor through the subdivision 1-form basis functions would also be interesting. Moreover, our encoding of arbitrary 2-tensors may also find other applications in the context of simulation: the stress and strain tensors used in elasticity could be encoded with intrinsic values—instead of using a piecewise-constant representation induced by the embedding of the triangle mesh as typically done in finite-element methods.

- One intriguing extension for our model-reduced fluid integrator is the use of reduced bases with spatial locality, as our integrator is not restricted to any particular set of bases functions. For instance, using wavelets for vorticity may offer *optimal sparsity in the structural coefficients* if we can address the challenge of adapting the frequency to the local feature size of the domain. Improving scalability (through sparsity in structural coefficients or pseudo-spectral methods) and better adaptivity to moving/deforming solid boundaries (through spatial locality) may then offer a wider applicability for model reduced methods. Another possible extension is to incorporate free surface boundary conditions through our modified Hodge star, combined with wavelet representations for the volume of fluid per cell.

APPENDICES

APPENDIX A

EXPLICIT EVALUATION OF OPERATORS BASED ON COVARIANT DERIVATIVE

In this appendix, we describe how one encodes our discrete operators for a vector field $\mathbf{u}$ as matrices acting on the vector coordinates $(u_i^1, u_i^2)^T$ at each vertex v_i. In this appendix, we adopt the following shorthand notation for clarity: $\rho \equiv \rho_{v_i \to e_{ij}}$, $\epsilon \equiv \epsilon_{ij}$, and $\theta \equiv \angle(\mathbf{e}_{e_{ij}}, \mathbf{e}_{t_{ijk}})$.

A.1 Divergence/curl for nVector Fields

When all the local frames in the neighborhood rotate by $-\alpha$, the representation vector field $\mathbf{v}$ of an n-vector field can be expressed in the new frame as $\mathbf{v'} = \exp(Jn\alpha)\mathbf{v}$. The covariant derivative with respect to an arbitrary vector field $\mathbf{w}$ $\nabla_{\mathbf{w}}\mathbf{v} = (\nabla\mathbf{v})\mathbf{w}$ also changes expression as an n-vector field, yielding:

$$\exp(Jn\alpha)(\nabla\mathbf{v})\mathbf{w} = (\nabla\mathbf{v'})\mathbf{w'} = (\nabla\mathbf{v'})\exp(J\alpha)\mathbf{w}.$$

Applying Eq. 2.3 and noting $F\exp(J\alpha) = \exp(-J\alpha)F$,

$$\begin{aligned}\nabla\mathbf{v'} &= \exp(Jn\alpha)(\nabla\mathbf{v})\exp(-J\alpha) \\ &= \frac{1}{2}\exp(Jn\alpha)(\partial\mathbf{v} + F\bar{\partial}\mathbf{v})\exp(-J\alpha) \\ &= \frac{1}{2}\exp(J(n-1)\alpha)\partial\mathbf{v} + \frac{1}{2}\exp(J(n+1)\alpha)F\bar{\partial}\mathbf{v}.\end{aligned}$$

Thus $\partial\mathbf{v}$ transforms as an $(n-1)$-vector field, while $\bar{\partial}\mathbf{v}$ transforms as an $(n+1)$-vector field.

A.2 Edge-based Operators

Our discrete operators are each represented as a $|F| \times 2|V|$ matrix, assembled based on the contribution of the vector coordinates at each vertex v_i to the integral value of the operator on each

adjacent triangle t_{ijk}. Through integration by parts, we find

$$\begin{aligned}\int_{e_{ij}} \Psi_i dl &= |e_{ij}| \int_0^1 (1-x)\exp(-J(\epsilon x + \rho))dx \\ &= \frac{|e_{ij}|}{\epsilon^2}\exp(-J\rho)[I - J\epsilon - \exp(-J\epsilon)].\end{aligned}$$

We can now evaluate the four discrete operators through the following function:

$$I(\rho,\epsilon) = \frac{|e_{ij}|}{\epsilon^2|t_{ijk}|}[\cos(\rho) - \cos(\rho+\epsilon) - \sin(\rho)\epsilon].$$

If we denote by $\mathrm{op}_{t_{ijk}}^{u_i^m}$ the contribution of the m-th component of $\mathbf{u}_i$ to the integral of op in t_{ijk}, we have (recall that for an n-vector field, divergence and curl operators produce an $(n-1)$-vector field, while the reflected ones produce an $(n+1)$-vector field):

$$\begin{aligned}curl\ {}^{u_i^1}_{t_{ijk}} &= I(n\rho + (n-1)\theta, n\epsilon), \\ curl\ {}^{u_i^2}_{t_{ijk}} &= I(n\rho + (n-1)\theta| + \pi/2, n\epsilon), \\ \mathrm{d}iv\ {}^{u_i^1}_{t_{ijk}} &= I(n\rho + \pi/2 + (n-1)\theta, n\epsilon), \\ \mathrm{d}iv\ {}^{u_i^2}_{t_{ijk}} &= I(n\rho + \pi/2 + (n-1)\theta, n\epsilon), \\ curl\ {}^{u_i^1}_{t_{ijk}} &= I(n\rho + (n+1)\theta, n\epsilon), \\ curl\ {}^{u_i^2}_{t_{ijk}} &= I(n\rho + (n+1)\theta + \pi/2, n\epsilon), \\ div\ {}^{u_i^1}_{t_{ijk}} &= I(n\rho + (n+1)\theta + \pi/2, n\epsilon), \\ div\ {}^{u_i^2}_{t_{ijk}} &= I(n\rho + (n+1)\theta + \pi, n\epsilon).\end{aligned}$$

A.3 Triangle-based Operators

We evaluated the per-triangle integral expressions of our operators through symbolic integration. Note that it leads to expressions with $\tau_{ij,k}$, $\tau_{jk,i}$, and $\tau_{ki,j}$ appearing in the denominator. As these

values are typically close to degenerate, Chebyshev [1] or Taylor expansion is necessary to provide robustness in evaluation.

APPENDIX B

DETAILS FOR 2-TENSOR RELATED OPERATORS

B.1 Decomposition of Killing Operator

We now detail the decomposition of the Killing operator $\mathcal{K}$ for smooth surfaces with Gaussian curvature κ. We first make use of the Bochner technique (see, e.g., [74, 20]), and expand the divergence of the Killing operator as:

$$-\operatorname{div}\ (\mathcal{K}(\omega)) = (2d\delta + \delta d - 2\kappa I)\,\omega, \tag{B.1}$$

where $\delta := -\star\, d\star$ is a shorthand for the co-differential operator. (Note that we assumed a 1-form ω with zero Dirichlet or Neumann boundary condition for simplicity.) By combining Eqs. (2.9) and (B.1), we now compute the inner product of symmetric 2-tensor fields generated by the Killing operator of exact 1-forms df, co-exact 1-forms $\star dg$, and harmonic 1-forms h:

$$\begin{cases} \langle \mathcal{K}(df), \mathcal{K}(\star dg)\rangle_F = 2\langle f, \star(d\kappa \wedge dg)\rangle_0, \\ \langle \mathcal{K}(df), \mathcal{K}(h)\rangle_F = 2\langle f, \star(d\kappa \wedge \star h)\rangle_0, \\ \langle \mathcal{K}(\star dg), \mathcal{K}(h)\rangle_F = 2\langle g, \star(d\kappa \wedge h)\rangle_0. \end{cases}$$

These expressions return zero for arbitrary f, g and h *iff* the Gaussian curvature κ is constant. Therefore, we can decompose the Killing operator $\mathcal{K}$ into an orthogonal direct sum as stated in Eq. (5.2) in the case of planar domains.

B.2 Lumping of Pairing

The proof found in [115] that the diagonal Hodge star is a lumping of the Galerkin Hodge star in the Laplacian operator extends directly to our discrete pairing operators for arbitrary symmetric

tensors since:

$$\begin{aligned}\left(\mathbf{d}_0^t M^\sigma \mathbf{d}_0\right)_{ij} &= \sum_{kl,mn} \left(\mathbf{d}_0^t\right)^{i,kl} \left(\int_M \sigma(\boldsymbol{\phi}_{kl}, \boldsymbol{\phi}_{mn})\right) \mathbf{d}_0^{mn,j} \\ &= \int_M \sigma\left(\sum_{kl} \mathbf{d}_0^{kl,i} \boldsymbol{\phi}_{kl}, \sum_{mn} \mathbf{d}_0^{mn,j} \boldsymbol{\phi}_{mn}\right) \\ &= \int_M \sigma\left(\nabla\phi_i, \nabla\phi_j\right) = \left(\mathbf{d}_0^t H^\sigma \mathbf{d}_0\right)_{ij}.\end{aligned}$$

B.3 Discrete Generalized Laplacian Δ^τ

The matrix $\mathbf{H}^\tau$ is provided in closed form for the various terms of the decomposition in Eq. (5.5). Note that the evaluation stencil for each element requires the "butterfly" patch of edge ij (or part thereof), and we use the naming convention described in the inset of §5.3.2.

• **Case $\tau = I$.** This case corresponds to the well-known cotan formula [116]:

$$\mathbf{H}_{ij}^{\mathrm{Id}} = \tfrac{1}{2}\left(\cot\theta_{jki} + \cot\theta_{ilj}\right) \tag{B.2}$$

• **Case $\tau = \mu$.**

$$\mathbf{H}_{ij}^{\mu} = \tfrac{1}{2a_{jil}}\mu_{jil} - \tfrac{1}{2a_{ijk}}\mu_{ijk}. \tag{B.3}$$

• **Case $\tau = \mathcal{K}(\mathbf{d}_0 f + \bigstar^{-1}\mathbf{d}_1^t g)$.** Using $\omega \equiv \mathbf{d}_0 f + \bigstar^{-1}\mathbf{d}_1^t g$ for conciseness, we have:

$$\begin{aligned}\mathbf{H}_{ij}^{\mathcal{K}} = &-\tfrac{1}{2l_{ij}^2}\left(\cot\theta_{kij}\cot\theta_{ijk} + \cot\theta_{jil}\cot\theta_{lji}\right) T_{ij}(\omega) \\ &+ \frac{1}{4a_{ijk}}\cot\theta_{jki}\left(T_{ik}(\omega) + T_{kj}(\omega)\right) + \frac{1}{4a_{jil}}\cot\theta_{ilj}\left(T_{li}(\omega) + T_{jl}(\omega)\right)\end{aligned}$$

• **Case $\tau = \overline{\mathcal{K}}(\mathbf{d}_0 w + h)$.**

$$\mathbf{H}_{ij}^{\overline{\mathcal{K}}} = \tfrac{1}{l_{ij}^2} T_{ij}(\mathbf{d}_0 w + h)$$

• **Case $\tau = B$.**

$$\mathbf{H}_{ij}^{B} = \frac{cot\theta_{kij} - cot\theta_{ijk}}{2(cot\theta_{kij} + cot\theta_{ijk})} + \frac{cot\theta_{lji} - cot\theta_{jik}}{2(cot\theta_{jil} + cot\theta_{lji})}$$

• **Case $\tau = C$.**

$$\mathbf{H}^C_{ij} = -\frac{1 + cot\theta_{kij} cot\theta_{ijk}}{2(cot\theta_{kij} + cot\theta_{ijk})} - \frac{1 + cot\theta_{jik} cot\theta_{lji}}{2(cot\theta_{jil} + cot\theta_{lji})}$$

B.4 Pairing through Discrete Tensors

The matrix $\mathbf{M}^\tau$ is provided in closed form for the various terms of the decomposition in Eq. (5.5). Note that the evaluation stencil for each element requires *either* the "butterfly" patch of edge ij *or* the patch of a face ijk, and we still use the naming convention described in the inset of §5.3.2.

• **Case $\tau = I$.**

$$\mathbf{M}^{\mathrm{Id}}_{ij,ij} = \tfrac{1}{4}\left(\cot\theta_{jki} + \cot\theta_{ilj}\right) + \tfrac{1}{12}\left(\cot\theta_{kij} + \theta_{ijk}\right) + \tfrac{1}{12}\left(\cot\theta_{jil} + \cot\theta_{lji}\right)$$
$$\mathbf{M}^{\mathrm{Id}}_{ij,jk} = \tfrac{1}{12}\left(\cot\theta_{ijk} - \cot\theta_{kij} - \cot\theta_{jki}\right).$$

This resulting matrix $\mathbf{M}^{\mathrm{Id}}$ corresponds to the Galerkin Hodge star $\bigstar^{\mathrm{G}}$ [115], as further discussed in §5.4.4.

• **Case $\tau = \mu$.**

$$\mathbf{M}^{\mu}_{ij,jk} = -\mathbf{M}^{\mu}_{jk,ij} = -\frac{\mu_{ijk}}{6a_{ijk}}, \quad \mathbf{M}^{\mu}_{ij,ij} = 0. \tag{B.4}$$

• **Case $\tau = B$.** We need to define a local coordinate frame to compute this pairing. For diagonal terms $\mathbf{M}^B_{ij,ij}$, we use the coordinate frame induced by the edge ij of the butterfly patch; for the other terms $\mathbf{M}^B_{ij,jk}$, we pick a random, but fixed frame F_{ijk} per face, and denote by η the angle that rotates the x direction of the local frame F_{ijk} to the direction of edge ki. With this convention, one gets:

$$\mathbf{M}^B_{ij,ij} = \frac{cot\theta_{ijk} - cot\theta_{kij}}{4(cot\theta_{ijk} + cot\theta_{kij})} + \frac{cot\theta_{jil} - cot\theta_{lji}}{4(cot\theta_{jil} + cot\theta_{lji})}$$
$$\mathbf{M}^B_{ij,jk} = sin(2\eta)\frac{1 + cot\theta_{jki} cot\theta_{kij} + (cot\theta_{jki} + cot\theta_{kij})^2}{12(cot\theta_{jki} + cot\theta_{kij})} + cos(2\eta)\frac{cot\theta_{kij} - cot\theta_{jki}}{12(cot\theta_{jki} + cot\theta_{kij})}$$

• **Case $\tau = C$.** Using the same convention as above:

$$\mathbf{M}^{C}_{ij,ij} = \frac{3 + cot\theta_{ijk}cot\theta_{kij} - cot^2\theta_{ijk} - cot^2\theta_{kij}}{12(cot\theta_{ijk} + cot\theta_{kij})} + \frac{3 + cot\theta_{lji}cot\theta_{jil} - cot^2\theta_{lji} - cot^2\theta_{jil}}{12(cot\theta_{jil} + cot\theta_{lji})}$$

$$\mathbf{M}^{C}_{ij,jk} = cos(2\eta)\frac{1 + cot\theta_{jki}cot\theta_{kij} + (cot\theta_{jki} + cot\theta_{kij})^2}{12(cot\theta_{jki} + cot\theta_{kij})} - sin(2\eta)\frac{cot\theta_{kij} - cot\theta_{jki}}{12(cot\theta_{jki} + cot\theta_{kij})}$$

• **Case $\tau = \mathcal{K}(\mathbf{d}_0 f + \bigstar^{-1}\mathbf{d}_1^t g)$.** Using $\omega \equiv \mathbf{d}_0 f + \bigstar^{-1}\mathbf{d}_1^t g$ for conciseness, the closed form expression is:

$$\begin{aligned}\mathbf{M}^{\mathcal{K}}_{ij,jk} = \frac{1}{24a_{ijk}}\Big[&T_{ij}(\omega)\left(2\cot\theta_{ijk} - \cot\theta_{kij}\right) + T_{jk}(\omega)\left(2\cot\theta_{ijk} - \cot\theta_{jki}\right)\\ &- T_{ki}(\omega)\left(\cot\theta_{jki} + \cot\theta_{kij}\right) + \tfrac{1}{2}\cot\theta_{ijk}\left(\cot\theta_{kij} - \cot\theta_{jki}\right)(\mathbf{d}_1\omega)_{ijk}\\ &- \tfrac{1}{2}\cot\theta_{ijk}\left(\cot\theta_{jil} + \cot\theta_{lji}\right)(\mathbf{d}_1\omega)_{jil} + \tfrac{1}{2}\cot\theta_{ijk}\left(\cot\theta_{kjm} + \cot\theta_{mkj}\right)(\mathbf{d}_1\omega)_{kjm}\Big]\end{aligned}$$

$$\begin{aligned}\mathbf{M}^{\mathcal{K}}_{ij,ij} = &\frac{T_{ij}(\omega)}{6l_{ij}^2}\Big[\cot^2\theta_{kij} + \cot^2\theta_{ijk} - \cot\theta_{kij}\cot\theta_{ijk} + \cot^2\theta_{jil} + \cot^2\theta_{lji} - \cot\theta_{jil}\cot\theta_{lji}\Big]\\ &+ \frac{1}{12a_{ijk}}\Big[T_{jk}(\omega)\left(\cot\theta_{ijk} + \cot\theta_{jki}\right) + T_{ki}(\omega)\left(\cot\theta_{jki} + \cot\theta_{kij}\right)\Big]\\ &+ \frac{1}{12a_{jil}}\Big[T_{il}(\omega)\left(\cot\theta_{jil} + \cot\theta_{ilj}\right) + T_{lj}(\omega)\left(\cot\theta_{ilj} + \cot\theta_{lji}\right)\Big]\\ &+ \frac{(\mathbf{d}_1\omega)_{ijk}}{48a_{ijk}}\Big[2\cot\theta_{jki}(\cot\theta_{kij} - \cot\theta_{ijk}) + \cot^2\theta_{lji} - \cot^2\theta_{jil}\Big]\\ &+ \frac{(\mathbf{d}_1\omega)_{jil}}{48a_{jil}}\Big[2\cot\theta_{ilj}(\cot\theta_{lji} - \cot\theta_{jil}) + \cot^2\theta_{kij} - \cot^2\theta_{ijk}\Big]\\ &+ \frac{1}{48a_{ijk}}\Big[(\mathbf{d}_1\omega)_{kjm}(\cot\theta_{kjm} + \cot\theta_{mkj})(\cot\theta_{ijk} + \cot\theta_{jki})\\ &\qquad - (\mathbf{d}_1\omega)_{ikn}(\cot\theta_{nik} + \cot\theta_{ikn})(\cot\theta_{kij} + \cot\theta_{jki})\Big]\\ &+ \frac{1}{48a_{jil}}\Big[(\mathbf{d}_1\omega)_{liu}(\cot\theta_{liu} + \cot\theta_{uil})(\cot\theta_{ilj} + \cot\theta_{jil})\\ &\qquad - (\mathbf{d}_1\omega)_{jlv}(\cot\theta_{vjl} + \cot\theta_{jlv})(\cot\theta_{ilj} + \cot\theta_{lji})\Big]\end{aligned}$$

• **Case $\tau = \overline{\mathcal{K}}(\mathbf{d}_0 w + h)$.**

$$\mathbf{M}^{\overline{\mathcal{K}}}_{ij,jk} = 0, \qquad \mathbf{M}^{\overline{\mathcal{K}}}_{ij,ij} = -\frac{1}{l_{ij}^2}T_{ij}(\mathbf{d}_0 w + h)$$

The similarity of this last diagonal term with $\mathbf{H}^{\overline{\mathcal{K}}}$ is explained in §5.4.4 where Hodge star approximations are discussed.

APPENDIX C

DETAILS FOR FLUID SIMULATION COMPUTATION

C.1 Computing Spectral Bases

In this appendix, we describe how to compute the spectral bases for both vector fields and density fields on a mesh $\mathcal{M}$.

Discrete Laplacians. Finding our spectral bases first requires discretizing both the scalar Laplacian $\nabla\cdot\nabla$ and the vector Laplacian $-\nabla\times\nabla\times+\nabla\nabla\cdot$ on the domain $\mathcal{M}$. Discretization of these operators on arbitrary simplicial complexes is well documented [134, 57], and only involves topological operators d_1 and d_2 deriving from the mesh connectivity, and diagonal "Hodge star" operators $\star_1, \star_2$, and $\star_3$ based on local measures of $\mathcal{M}$ and its circumcentric dual, resulting in the following symmetric second-order operators:

$$\star_3\Delta_3 \equiv \star_3 d_2 \star_2^{-1} d_2^t \star_3, \quad \star_2\Delta_2 \equiv d_2^t \star_3 d_2 + \star_2 d_1 \star_1^{-1} d_1^t \star_2 .$$

Note that d and $\star$ are even simpler on regular grids, even with the alteration we introduced in 6.3.4.

Boundary conditions. The canonical boundary conditions of velocity fields in fluid simulation for graphics purposes are no-transfer (i.e., the normal component v_n of the velocity along $\partial\mathcal{M}$ must be zero) and free-slip (i.e., the derivative of the tangential velocity field along the boundary normal $\partial v_t/\partial n$ must be zero as well). To enforce these conditions, we thus add the conditions that the flux of Ψ_i on *every boundary face* is zero, and that the circulations along the (interior half) boundary of the Voronoi face associated with *each boundary edge* is also zero (i.e, we simply set the values of $\star_1^{-1}$ as zeros for all the edges adjacent to the boundary faces to compute Δ_2). As for the eigenfunctions Φ_i of Δ_3, we use either Dirichlet boundary conditions $f|_{\partial M} = 0$ or Neumann boundary conditions $\frac{\partial f}{\partial n}|_{\partial M} = 0$, by considering boundary cell values or boundary gradients as

null. If other, non-homogeneous boundary conditions (such as influx or outflux conditions) are required, then one must add an additional harmonic (zeroth frequency) component that satisfies the given boundary conditions.

Eigen computations. Once the Laplacians with proper boundary conditions are assembled, we can compute their low-frequency eigenfields using a simple Lanczos algorithm since these operators are symmetric. The constant eigenbases from the kernel of Δ_3 can be safely omitted by setting zero values on boundaries, since a constant scalar function is unchanged when advected by a divergence-free velocity field. Note that, as mentioned in §6.3.1, some of the eigenfields Ψ_i will be of the form $1/\mu_j \delta\Phi_j$: indeed, $\delta\Phi_i$ for $i \neq 0$ is an eigenfunction of Δ_2 since

$$\Delta_2 \delta\Phi_j = \delta\Delta_3\Phi_j = \delta(-\mu_j^2\Phi_j) = -\mu_j^2\delta\Phi_j.$$

These gradient fields are easily identifiable by checking their divergence. Note finally that in theory, there could be cases where $\kappa_i^2 = \mu_j^2$ for multiple pairs of indices i and j. While in practice this is very unlikely to happen, one can protect against this rare event by replacing one of the corresponding Ψ_i by $\delta\Phi_j/\mu_j$, and replacing the other eigenvectors of this eigenvalue through a Gram-Schmidt process to form an orthonormal basis again.

Comments. We note that the approach we described above to compute the eigen bases for our fluid integrator is far from unique. For example, the β_1 harmonic vector fields are obtained as the eigenvectors associated with the eigenvalue 0, but we could have also computed the harmonic function dual to each homology generator via simple sparse linear systems instead [139]. Additionally, the vector field basis Ψ_i could be computed through its vector potential $\boldsymbol{\psi}_i$ instead: indeed, these vector potential 1-forms are eigenvectors of the 1-form Laplacian Δ_1, and boundary edge circulations as well as divergence on boundary dual cell divergence are assumed null to guarantee no-transfer and free-slip conditions. The curl of these 1-form basis functions are then, by construction, the flux-based Ψ_i basis functions. Finally, we point out that our approach is purposely different from what is proposed in [9], as they use an eigendecomposition of $d\delta$ instead (leveraging

the divergence-freeness of the vector fields). However, this simplified operator has a much larger null space that includes also curl-free fields, requiring many more eigenvectors to be computed via Lanczos iterations to generate divergence-free fields.

C.2 Analysis of Structural Coefficients C_k

For simplicity, we use the periodic domain $[0,1]^3$, i.e., the flat 3D torus. The eigenfields can be expressed using complex numbers as

$$\Psi_i(\mathbf{x}) = \mathbf{w}_i e^{\jmath \mathbf{k}_i \cdot \mathbf{x}},$$

where $\jmath$ is the unit imaginary number, $\mathbf{x}$ is the 3D coordinates, $\mathbf{w}_i$ is a unit vector, and $\mathbf{k}_i$ is the wave number vector (i.e., with $|\mathbf{k}_i| = \kappa_i$). The divergence of the basis function is thus

$$\mathrm{div}\Psi_i(\mathbf{x}) = \jmath \mathbf{k}_i \cdot \mathbf{w}_i e^{\jmath \mathbf{k}_i \cdot \mathbf{x}},$$

while the curl is expressed as

$$\mathrm{curl}\Psi_i(\mathbf{x}) = \jmath \mathbf{k}_i \times \mathbf{w}_i e^{\jmath \mathbf{k}_i \cdot \mathbf{x}}.$$

Since $\mathrm{div}\Psi_i = 0$ means that $\mathbf{k}_i \cdot \mathbf{w}_i = 0$, there are two independent $\mathbf{w}$'s for each $\mathbf{k}$ in the basis. We can thus compute the structural coefficients $C_{c,ab}$ from Eq. (6.7) in closed form, by the integral of

$$(\nabla \times \Psi_a) \cdot (\Psi_c^* \times \Psi_b) = \jmath(\mathbf{k}_a \times \mathbf{w}_a) \cdot (\mathbf{w}_c \times \mathbf{w}_b) e^{\jmath(\mathbf{k}_a + \mathbf{k}_b - \mathbf{k}_c)\cdot \mathbf{x}},$$

where superscript * denotes complex conjugation. Note that nonzero coefficients $\jmath(\mathbf{k}_a \times \mathbf{w}_a) \cdot (\mathbf{w}_c \times \mathbf{w}_b)$ only exist when $\mathbf{k}_c = \mathbf{k}_a + \mathbf{k}_b$, and they advect real fields to real fields (whose coefficients satisfy $v_{\mathbf{k},\mathbf{w}} = v^*_{-\mathbf{k},\mathbf{w}}$). It indicates that $|\mathbf{k}_b|^2 C_{c,ab} = |\mathbf{k}_a|^2 C_{c,ba}$ is not true in general in 3D, contrary to the claim in [9]; a simple counterexample is $\mathbf{k}_a = 2\pi(0,2,3)$, $\mathbf{w}_a = (1,0,0)$, $\mathbf{k}_b = 2\pi(1,1,0)$, and $\mathbf{w}_b = (0,0,1)$. Moreover, while $C_{c,aa} = 0$ indeed for this domain since $\mathbf{w}_c \cdot (2\mathbf{k}_a) = 0$, this property will no longer hold for an arbitrary domain. Thankfully, our variational integrator does not depend on the eigenmodes being steady flows, so these symmetries (or rather, lack thereof) are inconsequential.

C.3 Kelvin's Circulation Theorem

Ideal, incompressible fluids have a conserved momentum [140] given by the integrated circulation of the fluid around a closed curve which is advected by the flow. This fact is known as Kelvin's circulation theorem. Methods such as [19] and [50] are constructed so as to conserve a discretized form of this conserved momentum. Our spectral method also obeys a form of Kelvin's theorem as follows. We can define generalized "spectral" curves as spectral dual 1-chains (also called 1-currents [134]) of the form:

$$\Gamma = \sum_i \gamma_i \star_2 A_i. \tag{C.1}$$

The above dual 1-chain expression always represents a closed curve, because each A_i corresponds to a closed (divergence-free) 2-form, which means the dual 1-chain is boundaryless. A pairing between a 2-form and a generalized loop is defined as expected:

$$\langle A, \Gamma \rangle = \left\langle \sum_i v_i A_i, \sum_j \gamma_j \star_2 A_j \right\rangle = \sum_i v_i \gamma_i. \tag{C.2}$$

The Lie advection of the generalized curve along the velocity field

$$\dot{\Gamma} = -[A, \Gamma] \tag{C.3}$$

indicates that the coefficients $\{\gamma_i\}_i$ must evolve such that

$$\begin{aligned}
\dot{\gamma_k} &= -\sum_{i,j} \gamma_i v_j \int_{\mathcal{M}} \Psi_k \cdot (\nabla \times (\Psi_i \times \Psi_j)) \\
&= \sum_{i,j} \gamma_i v_j \int_{\mathcal{M}} (\nabla \times \Psi_k) \cdot (\Psi_j \times \Psi_i) = \sum_{i,j} \gamma_i v_j C_{j,ki}.
\end{aligned}$$

Thus, the spectral version of Kelvin's theorem holds since

$$\begin{aligned}
\frac{d}{dt}\langle A, \Gamma \rangle &= \sum_i (\dot{v_i}\gamma_i + v_i \dot{\gamma_i}) \\
&= \sum_i \mathbf{v}^t \mathbf{C}_i \mathbf{v} \gamma_i + \sum_i v_i \sum_{j,k} v_k \gamma_j C_{k,ij} \\
&= \sum_{i,j,k} v_j C_{i,jk} v_k \gamma_i - \sum_{i,j,k} v_j v_k \gamma_i C_{i,jk} = 0.
\end{aligned}$$

In the above derivation, dummy index variables are swapped and the identity $C_{k,ij} = -C_{j,ik}$ is used.

C.4 Temporal Discretization

A fully discrete (in space and time) treatment of our variational integrator is easily achieved using the Hamilton-Pontryagin principle [49], where Lagrange multipliers μ^k enforce that A is indeed the Eulerian velocity of state q. If one denotes by h is the time step, A^k the velocity field between time k & state q^k and time $k+1$ & state q_{k+1}, the discrete Hamilton-Pontryagin action between $t = 0$ and $t = Nh$ is expressed as

$$S_d = \sum_{k=0}^{N-1} \frac{1}{2}\langle A^k, A^k\rangle h + \langle \mu^k, \tau^{-1}(q^{k+1}(q^k)^{-1}) - hA^k\rangle.$$

The map τ must convert an element of the Lie algebra to a Lie group element, thus making A_k the Eulerian velocity between time t_k and t_{k+1}; instead of the usual exponential map which is computationally difficult to handle, we approximate it to be the Cayley transform $\tau(A) = (I - A/2)^{-1}(I + A/2)$, as it efficiently maps antisymmetric matrices to orthogonal matrices. Taking variations with respect to μ^k, we recover the expected group element update rule

$$q^{k+1} = \tau(hA^k)q^k.$$

Variations with respect to A^k show that the multiplier is actually the momentum: $\mu^k = A^k$. Finally variations with respect to q^k restricted to $\delta q^k = B^k q^k$ with B^k in the Lie algebra (to enforce Lin constraints) yield

$$\begin{aligned}&\langle \mu^{k-1}, (I - hA^{k-1}/2)B^k(I + hA^{k-1}/2)\rangle \\ &\quad = \langle \mu^k, (I + hA^k/2)B^k(I - hA^k/2)\rangle.\end{aligned}$$

Omitting the cubic terms in $O(h^2)$ still preserves a discrete Kelvin's theorem, so we follow the suggestion in [10] and simplify the update rule to

$$\forall B^k, \langle A^{k-1}, B^k + \frac{h}{2}[B^k, A^{k-1}]\rangle = \langle A^k, B^k + \frac{h}{2}[B^k, -A^k]\rangle,$$

which reduces to the trapezoidal rule with $A^k = \sum_i v_i^k A_i$ and an arbitrary $B^k = \sum_i b_i^k A_i$.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] F. Knöppel, K. Crane, U. Pinkall, and P. Schröder, "Globally optimal direction fields," *ACM Trans. Graph.*, vol. 32, pp. 59:1–59:10, July 2013.

[2] S. Lefebvre and H. Hoppe, "Appearance-space texture synthesis," *ACM Trans. Graphics*, vol. 25, no. 3, pp. 541–548, 2006.

[3] K. Crane, C. Weischedel, and M. Wardetzky, "Geodesics in heat: A new approach to computing distance based on heat flow," *ACM Trans. Graph.*, vol. 32, no. 5, 2013.

[4] M. Desbrun, E. Kanso, and Y. Tong, "Discrete differential forms for computational modeling," in *Discrete Differential Geometry,* A. I. Bobenko *et al.* (Eds), vol. 38 of *Oberwolfach Seminars*, pp. 287–324, Birkhäuser Basel, 2008.

[5] J. Palacios and E. Zhang, "Interactive visualization of rotational symmetry fields on surfaces," vol. 17, no. 7, pp. 947–955, 2011.

[6] O. Azencot, M. Ben-Chen, F. Chazal, and M. Ovsjanikov, "An operator approach to tangent vector field processing," *Computer Graphics Forum*, vol. 32, no. 5, pp. 73–82, 2013.

[7] Y. T. Ng, C. Min, and F. Gibou, "An efficient fluid-solid coupling algorithm for single-phase flows," *J. Comput. Phys.*, vol. 228, pp. 8807–8829, Dec. 2009.

[8] C. Foias, D. Holm, and E. Titi, "The three dimensional viscous Camassa–Holm equations, and their relation to the Navier–Stokes equations and turbulence theory," *Journal of Dynamics and Differential Equations*, vol. 14, no. 1, pp. 1–35, 2002.

[9] T. De Witt, C. Lessig, and E. Fiume, "Fluid simulation using Laplacian eigenfunctions," *ACM Trans. Graph.*, vol. 31, pp. 10:1–10:11, Feb. 2012.

[10] E. Gawlik, P. Mullen, D. Pavlov, J. Marsden, and M. Desbrun, "Geometric, variational discretization of continuum theories," *Physica D: Nonlinear Phenomena*, vol. 240, no. 21, pp. 1724–1760, 2011.

[11] M. D. P. S. Keenan Crane, Fernando de Goes, "Digital geometry processing with discrete exterior calculus," in *ACM SIGGRAPH 2013 courses*, SIGGRAPH '13, (New York, NY, USA), ACM, 2013.

[12] K. Polthier and E. Preuß, "Variational approach to vector field decomposition," in *Data Visualization 2000* (W. de Leeuw and R. van Liere, eds.), Eurographics, pp. 147–155, Springer Vienna, 2000.

[13] E. Zhang, K. Mischaikow, and G. Turk, "Vector field design on surfaces," *ACM Trans. Graphics*, vol. 25, no. 4, pp. 1294–1326, 2006.

[14] Y. Tong, S. Lombeyda, A. N. Hirani, and M. Desbrun, "Discrete multiscale vector field decomposition," *ACM Trans. Graphics*, vol. 22, no. 3, pp. 445–452, 2003.

[15] K. Polthier and E. Preuß, "Identifying vector field singularities using a discrete hodge decomposition," pp. 113–134, Springer Verlag, 2003.

[16] D. Pavlov, P. Mullen, Y. Tong, E. Kanso, J. Marsden, and M. Desbrun, "Structure-preserving discretization of incompressible fluids," *Physica D: Nonlinear Phenomena*, vol. 240, no. 6, pp. 443–458, 2011.

[17] M. Fisher, P. Schröder, M. Desbrun, and H. Hoppe, "Design of tangent vector fields," *ACM Trans. Graph.*, vol. 26, July 2007.

[18] H. Whitney, *Geometric Integration Theory*. Princeton University Press, 1957.

[19] S. Elcott, Y. Tong, E. Kanso, P. Schröder, and M. Desbrun, "Stable, circulation-preserving, simplicial fluids," *ACM Trans. Graph.*, vol. 26, Jan. 2007.

[20] M. Ben-Chen, A. Butscher, J. Solomon, and L. Guibas, "On discrete killing vector fields and patterns on surfaces," *Computer Graphics Forum*, vol. 29, no. 5, pp. 1701–1711, 2010.

[21] B. Sherlock and D. Monro, "A model for interpreting fingerprint topology," *Pattern Recognition*, vol. 26, no. 7, pp. 1047 – 1055, 1993.

[22] J. Feng, J. Zhou, and A. Jain, "Orientation field estimation for latent fingerprint enhancement," 2013.

[23] E. Zhang, J. Hays, and G. Turk, "Interactive tensor field design and visualization on surfaces," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 13, no. 1, pp. 94–107, 2007.

[24] J. Palacios and E. Zhang, "Rotational symmetry field design on surfaces," *ACM Trans. Graph.*, vol. 26, July 2007.

[25] Y.-K. Lai, M. Jin, X. Xie, Y. He, J. Palacios, E. Zhang, S.-M. Hu, and X. Gu, "Metric-driven rosy field design and remeshing," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 16, no. 1, pp. 95–108, 2010.

[26] K. Crane, M. Desbrun, and P. Schröder, "Trivial connections on discrete surfaces," *Computer Graphics Forum*, vol. 29, no. 5, pp. 1525–1533, 2010.

[27] N. Ray, B. Vallet, W. C. Li, and B. Lévy, "N-symmetry direction field design," *ACM Trans. Graph.*, vol. 27, pp. 10:1–10:13, May 2008.

[28] C. Eisenacher, C. Tappan, B. Burley, D. Teece, and A. Shek, "Example-based texture synthesis on disney's tangled," in *ACM SIGGRAPH 2010 Talks*, SIGGRAPH '10, (New York, NY, USA), pp. 32:1–32:1, ACM, 2010.

[29] B. Liu, Y. Weng, J. Wang, and Y. Tong, "Orientation field guided texture synthesis," *Journal of Computer Science and Technology (CVM)*, vol. 28, no. 5, pp. 827–835, 2013.

[30] Y. Lipman, O. Sorkine, D. Levin, and D. Cohen-Or, "Linear rotation-invariant coordinates for meshes," *ACM Trans. Graph.*, vol. 24, pp. 479–487, July 2005.

[31] S. Kircher and M. Garland, "Free-form motion processing," *ACM Trans. Graph.*, vol. 27, pp. 12:1–12:13, May 2008.

[32] Y. Wang, B. Liu, and Y. Tong, "Linear surface reconstruction from discrete fundamental forms on triangle meshes," *Computer Graphics Forum*, vol. 31, no. 8, pp. 2277–2287, 2012.

[33] A. Myles and D. Zorin, "Controlled-distortion constrained global parametrization," *ACM Trans. Graph.*, vol. 32, pp. 105:1–105:14, July 2013.

[34] F. de Goes, B. Liu, M. Budninskiy, Y. Tong, and M. Desbrun, "Discrete 2-tensor fields on triangulations," vol. (Symposium on Geometry Processing), 2014.

[35] N. Foster and D. Metaxas, "Modeling the motion of a hot, turbulent gas," in *Proc. ACM SIGGRAPH*, pp. 181–188, Aug. 1997.

[36] J. Stam, "Stable fluids," in *Proceedings of ACM SIGGRAPH*, pp. 121–128, Aug. 1999.

[37] J. Steinhoff and D. Underhill, "Modification of the Euler equations for Vorticity Confinement," *Physics of Fluids*, vol. 6, pp. 2738–2744, Aug. 1994.

[38] A. Selle, N. Rasmussen, and R. Fedkiw, "A vortex particle method for smoke, water and explosions," *ACM Trans. Graph.*, vol. 24, pp. 910–914, July 2005.

[39] X. Zhang, R. Bridson, and C. Greif, "Restoring the missing vorticity in advection-projection fluid solvers," *ACM Trans. Graph.*, vol. 34, no. 4, p. Art. 52, 2015.

[40] P. Mullen, K. Crane, D. Pavlov, Y. Tong, and M. Desbrun, "Energy-preserving integrators for fluid animation," *ACM Trans. Graph.*, vol. 28, pp. 38:1–38:8, July 2009.

[41] F. Losasso, J. Talton, N. Kwatra, and R. Fedkiw, "Two-way coupled SPH and Particle Level Set fluid simulation," *IEEE Trans. on Vis. and Comp. Graph.*, vol. 14, no. 4, pp. 797–804, 2008.

[42] A. Golas, R. Narain, J. Sewall, P. Krajcevski, P. Dubey, and M. Lin, "Large-scale fluid simulation using velocity-vorticity domain decomposition," *ACM Trans. Graph.*, vol. 31, pp. 148:1–148:9, Nov. 2012.

[43] B. E. Feldman, J. F. O'Brien, and B. M. Klingner, "Animating gases with hybrid meshes," *ACM Trans. Graph.*, vol. 24, pp. 904–909, July 2005.

[44] C. Batty, F. Bertails, and R. Bridson, "A fast variational framework for accurate solid-fluid coupling," *ACM Trans. Graph.*, vol. 26, pp. 100:1–100:8, July 2007.

[45] R. Howes, C. Schroeder, and J. M. Teran, "A virtual node algorithm for Hodge decompositions of inviscid flow problems with irregular domains," *Methods Appl. Anal.*, vol. 20, no. 4, pp. 439–455, 2013.

[46] J. Stam, "Flows on surfaces of arbitrary topology," in *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, (New York, NY, USA), pp. 724–731, ACM, 2003.

[47] O. Azencot, S. Weißmann, M. Ovsjanikov, M. Wardetzky, and M. Ben-Chen, "Functional fluids on surfaces," *Computer Graphics Forum*, vol. 33, no. 5, pp. 237–246, 2014.

[48] J. E. Marsden and M. West, "Discrete mechanics and variational integrators," *Acta Numerica 2001*, vol. 10, pp. 357–514, 2001.

[49] L. Kharevych, W. Yang, Y. Tong, E. Kanso, J. E. Marsden, P. Schröder, and M. Desbrun, "Geometric, variational integrators for computer animation," in *Symposium on Computer animation*, pp. 43–51, 2006.

[50] D. Pavlov, P. Mullen, Y. Tong, E. Kanso, J. Marsden, and M. Desbrun, "Structure-preserving discretization of incompressible fluids," *Physica D: Nonlinear Phenomena*, vol. 240, no. 6, pp. 443–458, 2011.

[51] J. Stam and E. Fiume, "Turbulent wind fields for gaseous phenomena," in *Proceedings of ACM SIGGRAPH*, pp. 369–376, 1993.

[52] T. Kim, N. Thürey, D. James, and M. Gross, "Wavelet turbulence for fluid simulation," in *ACM Transactions on Graphics (TOG)*, vol. 27, p. 50, ACM, 2008.

[53] Y. Gao, C.-F. Li, B. Ren, and S.-M. Hu, "View-dependent multiscale fluid simulation," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19, no. 2, pp. 178–188, 2013.

[54] H. Schechter and R. Bridson, "Evolving sub-grid turbulence for smoke animation," in *Symposium on Computer animation*, pp. 1–7, 2008.

[55] R. Narain, J. Sewall, M. Carlson, and M. C. Lin, "Fast animation of turbulence using energy transport and procedural synthesis," in *ACM Transactions on Graphics (TOG)*, vol. 27, p. 166, ACM, 2008.

[56] Z. Yuan, F. Chen, and Y. Zhao, "Pattern-guided smoke animation with Lagrangian Coherent Structure," in *ACM Trans. Graph.*, vol. 30, p. 136, 2011.

[57] S. Elcott and P. Schröder, "Building your own DEC at home," in *Discrete Differential Geometry*, ACM SIGGRAPH Courses, pp. 55–59, 2006.

[58] J. B. Frank and P. S. Heckbert, "A pliant method for anisotropic mesh generation," in *International Meshing Roundtable*, pp. 63–76, 1996.

[59] R. Narain, A. Samii, and J. F. O'Brien, "Adaptive anisotropic remeshing for cloth simulation," *ACM Trans. Graph.*, vol. 31, pp. 152:1–152:10, Nov. 2012.

[60] D. Panozzo, E. Puppo, M. Tarini, and O. Sorkine-Hornung, "Frame fields: Anisotropic and non-orthogonal cross fields," *ACM Trans. Graph. (SIGGRAPH)*, vol. 33, no. 4, 2014.

[61] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun, "Anisotropic polygonal remeshing," *ACM Trans. Graph.*, vol. 22, no. 3, 2003.

[62] G. Tewari, J. Snyder, P. V. Sander, S. J. Gortler, and H. Hoppe, "Signal-specialized parameterization for piecewise linear reconstruction," in *Symposium on Geometry Processing*, pp. 55–64, 2004.

[63] R. Zayer, C. Rossl, and H.-P. Seidel, "Discrete tensorial quasi-harmonic maps," in *Shape Modeling International 2005*, pp. 278–287, 2005.

[64] H. N. Iben and J. F. O'Brien, "Generating surface crack patterns," in *Symposium on Computer Animation*, pp. 177–185, Sept 2006.

[65] D. N. Arnold, R. S. Falk, and R. Winther, "Finite element exterior calculus, homological techniques, and applications," *Acta Numerica*, vol. 15, pp. 1–155, 2006.

[66] D. L. Meier, "Constrained transport algorithms for numerical relativity. i. development of a finite-difference scheme," *The Astrophysical Journal*, vol. 595, pp. 980–991, 2003.

[67] A. Kratz, C. Auer, M. Stommel, and I. Hotz, "Visualization and analysis of second-order tensors: Moving beyond the symmetric positive-definite case," vol. 32, no. 1, pp. 49–74, 2013.

[68] G. Alessandrini and V. Nesi, "Area formulas for σ-harmonic mappings," in *Nonlinear Problems in Mathematical Physics and Related Topics I*, vol. 1 of *International Mathematical Series*, pp. 1–21, Springer US, 2002.

[69] P. B. Bochev and J. M. Hyman, "Principles of Mimetic Discretizations of Differential Operators," *IMA Volumes*, vol. 142, pp. 89–119, 2006.

[70] J. M.Lee, *Introduction to Smooth Manifolds*. Springer, 2003.

[71] M. Spivak, *A comprehensive introduction to differential geometry. Vol. II*. Wilmington, Del.: Publish or Perish Inc., second ed., 1979.

[72] R. Abraham, J. Marsden, and T. Ratiu, *Manifolds, Tensor Analysis, and Applications: 2nd Edition*. Springer-Verlag, 1988.

[73] M. Berger and D. G. Ebin, "Some decompositions of the space of symmetric tensors on a Riemannian manifold," *Journal of Differential Geometry*, vol. 3, no. 3-4, pp. 379–392, 1969.

[74] P. Petersen, *Riemannian Geometry*, vol. 171 of *Graduate Texts in Mathematics*. Springer, 2006.

[75] H. C. Hu, "Some variational principles in elasticity and plasticity," *Acta Phys. Sin.*, vol. 10, no. 3, pp. 259–290, 1954.

[76] D. Kovacs, A. Myles, and D. Zorin, "Anisotropic quadrangulation.," *Computer Aided Geometric Design*, vol. 28, no. 8, pp. 449–462, 2011.

[77] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 7, pp. 629–639, 1990.

[78] F. de Goes, P. Alliez, H. Owhadi, and M. Desbrun, "On the equilibrium of simplicial masonry structures," *ACM Trans. Graph.*, vol. 32, no. 4, 2013.

[79] Y. Liu, P. Hao, J. Snyder, W. Wang, and B. Guo, "Computing self-supporting surfaces by regular triangulation," *ACM Trans. Graph. (SIGGRAPH)*, vol. 32, no. 4, 2013.

[80] E. Praun, A. Finkelstein, and H. Hoppe, "Lapped textures," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 465–470, ACM Press/Addison-Wesley Publishing Co., 2000.

[81] G. Turk, "Texture synthesis on surfaces," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 347–354, ACM, 2001.

[82] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: Image and video synthesis using graph cuts," *ACM Transactions on Graphics, SIGGRAPH 2003*, vol. 22, pp. 277–286, July 2003.

[83] Q. Wu and Y. Yu, "Feature matching and deformation for texture synthesis," in *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, (New York, NY, USA), pp. 364–367, ACM, 2004.

[84] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman, "Patchmatch: a randomized correspondence algorithm for structural image editing," *ACM Transactions on Graphics-TOG*, vol. 28, no. 3, p. 24, 2009.

[85] S. Lefebvre and H. Hoppe, "Parallel controllable texture synthesis," *ACM Trans. Graph.*, vol. 24, pp. 777–786, July 2005.

[86] F. González and G. Patow, "Continuity mapping for multi-chart textures," *ACM Trans. Graph.*, vol. 28, pp. 109:1–109:8, Dec. 2009.

[87] G. Turk, "Part ii: texturing surfaces and geometry creation," in *ACM SIGGRAPH 2007 courses*, SIGGRAPH '07, (New York, NY, USA), ACM, 2007.

[88] L.-Y. Wei, S. Lefebvre, V. Kwatra, and G. Turk, "State of the art in example-based texture synthesis," in *Eurographics 2009, State of the Art Report, EG-STAR*, Eurographics Association, 2009.

[89] H. Theisel, "Designing 2d vector fields of arbitrary topology," vol. 21, no. 3, pp. 595–604, 2002.

[90] A. Hertzmann and D. Zorin, "Illustrating smooth surfaces," pp. 517–526, 2000.

[91] N. Ray, B. Vallet, L. Alonso, and B. Levy, "Geometry-aware direction field processing," *ACM Trans. Graph.*, vol. 29, pp. 1:1–1:11, Dec. 2009.

[92] D. Bommes, H. Zimmer, and L. Kobbelt, "Mixed-integer quadrangulation," *ACM Trans. Graph.*, vol. 28, pp. 77:1–77:10, July 2009.

[93] D. Panozzo, Y. Lipman, E. Puppo, and D. Zorin, "Fields on symmetric surfaces," *ACM Trans. Graph.*, vol. 31, pp. 111:1–111:12, July 2012.

[94] D. Laidlaw and J. Weickert, *Visualization and Processing of Tensor Fields: Advances and Perspectives*. Springer Publishing Company, Incorporated, 1st ed., 2009.

[95] T. Delmarcelle and L. Hesselink, "The topology of symmetric, second-order tensor fields," pp. 140–147, 1994.

[96] X. Zheng and A. Pang, "2D asymmetric tensor analysis," pp. 3–10, 2005.

[97] E. Zhang, H. Yeh, Z. Lin, and R. Laramee, "Asymmetric tensor analysis for flow visualization," vol. 15, no. 1, pp. 106–122, 2009.

[98] D. Palke, Z. Lin, G. Chen, H. Yeh, P. Vincent, R. Laramee, and E. Zhang, "Asymmetric tensor field visualization for surfaces," vol. 17, no. 12, pp. 1979–1988, 2011.

[99] T. Regge, "General relativity without coordinates," *Nuovo Cim.*, vol. 19, no. 3, pp. 558–571, 1961.

[100] B. Springborn, P. Schröder, and U. Pinkall, "Conformal equivalence of triangle meshes," *ACM Trans. Graph.*, vol. 27, no. 3, 2008.

[101] F. Luo, "Rigidity of polyhedral surfaces, III," 2010. arXiv:1010.3284v1.

[102] M. Campen, M. Heistermann, and L. Kobbelt, "Practical anisotropic geodesy," *Computer Graphics Forum*, vol. 32, no. 5, pp. 63–71, 2013.

[103] F. de Goes, P. Memari, P. Mullen, and M. Desbrun, "Weighted triangulations for geometry processing," *ACM Trans. Graph.*, vol. 33, 2014.

[104] D. N. Arnold, R. S. Falk, and R. Winther, "Differential complexes and stability of finite element methods II: The elasticity complex," in *Compatible Spatial Discretizations* (D. Arnold, P. Bochev, R. Lehoucq, R. Nicolaides, and M. Shashkov, eds.), vol. 142 of *IMA Vol. Math. Appl.*, pp. 47–68, 2006.

[105] D. N. Arnold and R. Winther, "Mixed finite elements for elasticity," *Numer. Math.*, vol. 92, no. 3, pp. 401–419, 2002.

[106] D. N. Arnold and R. Winther, "Nonconforming mixed elements for elasticity," *Math. Models Methods Appl. Sci.*, vol. 13, no. 3, pp. 295–307, 2003.

[107] D. N. Arnold, G. Awanou, and R. Winther, "Finite elements for symmetric tensors in three dimensions," *Math. Comput.*, vol. 77, pp. 1229–1251, 2008.

[108] D. Arnold, G. Awanou, and R. Winther, "Nonconforming tetrahedral mixed finite elements for elasticity," *Math. Models Methods Appl. Sci.*, vol. 24, pp. 783–796, 2014.

[109] E. Kanso, M. Arroyo, Y. Tong, A. Yavari, J. Marsden, and M. Desbrun, "On the geometric character of stress," *ZAMP*, vol. 58(5), pp. 843–856, 2007.

[110] A. Green and W. Zerna, *Theoretical Elasticity*. Dover, 2002.

[111] J. R. Munkres, *Elements of Algebraic Topology*. Addison-Wesley, 1984.

[112] D. Cohen-Steiner and J.-M. Morvan, "Restricted Delaunay triangulations and normal cycle," pp. 312–321, 2003.

[113] E. Grinspun, Y. I. Gingold, J. Reisman, and D. Zorin, "Computing discrete shape operators on general meshes.," *Comput. Graph. Forum*, vol. 25, no. 3, pp. 547–556, 2006.

[114] K. Wang, Weiwei, Y. Tong, M. Desbrun, and P. Schröder, "Edge subdivision schemes and the construction of smooth vector fields," *ACM Trans. Graph.*, vol. 25, pp. 1041–1048, July 2006.

[115] A. Bossavit and L. Kettunen, "Yee-schemes on a tetrahedral mesh, with diagonal lumping," *Int. J. Num. Model.*, vol. 12, pp. 129–142, 1999.

[116] U. Pinkall and K. Polthier, "Computing discrete minimal surfaces and their conjugates," *Exp. Math.*, vol. 2, no. 1, pp. 15–36, 1993.

[117] D. Glickenstein, "Geometric triangulations & discrete Laplacians on manifolds," 2005. arXiv.org:math/0508188.

[118] P. Mullen, P. Memari, F. de Goes, and M. Desbrun, "HOT: Hodge-optimized triangulations," *ACM Trans. Graph. (SIGGRAPH)*, vol. 30, 2011.

[119] G. Guennebaud, B. Jacob, *et al.*, "Eigen v3." http://eigen.tuxfamily.org, 2010.

[120] K. Hildebrandt, K. Polthier, and M. Wardetzky, "On the convergence of metric and geometric properties of polyhedral surfaces," *Geometriae Dedicata*, vol. 123, no. 1, pp. 89–112, 2006.

[121] A. Treuille, A. Lewis, and Z. Popović, "Model reduction for real-time fluids," *ACM Trans. Graph.*, vol. 25, pp. 826–834, July 2006.

[122] I. Silberman, "Planetary waves in the atmosphere," *J. Meteor.*, vol. 11, pp. 27–34, 1954.

[123] V. Yudovich, "Non-stationary flow of an ideal incompressible liquid," *USSR Computational Mathematics and Mathematical Physics*, vol. 3, no. 6, pp. 1407–1456, 1963.

[124] S. A. Orszag, "Numerical methods for the simulation of turbulence," *Physics of Fluids*, vol. 12, pp. II 250–257, 1969.

[125] M. Gupta and S. G. Narasimhan, "Legendre fluids: A unified framework for analytic reduced space modeling and rendering of participating media," in *Symposium on Computer Animation*, pp. 17–25, 2007.

[126] B. Long and E. Reinhard, "Real-time fluid simulation using discrete sine/cosine transforms," in *Symposium on Interactive 3D Graphics and Games*, pp. 99–106, 2009.

[127] M. Stanton, Y. Sheng, M. Wicke, F. Perazzi, A. Yuen, S. Narasimhan, and A. Treuille, "Non-polynomial Galerkin projection on deforming meshes," *ACM Trans. Graph.*, vol. 32, pp. 86:1–86:14, July 2013.

[128] J. M. Cohen, S. Tariq, and S. Green, "Interactive fluid-particle simulation using translating Eulerian grids," in *ACM Symp. on Interactive 3D Graphics and Games*, pp. 15–22, 2010.

[129] S. An, T. Kim, and D. James, "Optimizing cubature for efficient integration of subspace deformations," *ACM Trans. Graph.*, vol. 27, no. 5, p. Art. 165, 2008.

[130] C. von Tycowicz, C. Schulz, H.-P. Seidel, and K. Hildebrandt, "An efficient construction of reduced deformable objects," *ACM Trans. Graph.*, vol. 32, no. 6, p. Art. 213, 2013.

[131] S. Li, J. Huang, F. de Goes, X. Jin, H. Bao, and M. Desbrun, "Space-time editing of elastic motion through material optimization and reduction," *ACM Trans. Graph.*, vol. 33, no. 4, p. Art. 108, 2014.

[132] T. Kim and J. Delaney, "Subspace fluid re-simulation," *ACM Trans. Graph.*, vol. 32, no. 4, p. 62, 2013.

[133] F. H. Harlow and J. E. Welch, "Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface," *Physics of Fluids*, vol. 8, pp. 2182–2189, Dec. 1965.

[134] M. Desbrun, E. Kanso, and Y. Tong, "Discrete differential forms for computational modeling," in *ACM SIGGRAPH'06 Course Notes on Discrete Differential Geometry*, ACM Press, 2006.

[135] M. Desbrun, E. S. Gawlik, F. Gay-Balmaz, and V. Zeitlin, "Variational discretization for rotating stratified fluids," *Disc. Cont. Dyn. S.*, vol. 34, no. 2, pp. 477–509, 2013.

[136] N. Bell and A. N. Hirani, "PyDEC: A Python library for Discrete Exterior Calculus," 2008. Google Code project at: `http://code.google.com/p/pydec/`.

[137] J. Stam, "A simple fluid solver based on the fft," *J. Graph. Tools*, vol. 6, pp. 43–52, Sept. 2002.

[138] K. Urban, "Wavelet bases for $H(\mathrm{div})$ and $H(\mathrm{curl})$," in *Wavelets in Numerical Simulation*, vol. 22 of *Lecture Notes in Computational Science and Engineering*, pp. 83–107, 2002.

[139] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun, "Designing quadrangulations with discrete harmonic forms," in *Symposium on Geometry Processing*, pp. 201–210, 2006.

[140] A. Chorin and J. Marsden, *A Mathematical Introduction to Fluid Mechanics*. Springer-Verlag, 3rd edition ed., 1979.