# Discrete Derivatives of Vector Fields on Surfaces – An Operator Approach

OMRI AZENCOT
Technion – Israel Institute of Technology
MAKS OVSJANIKOV
LIX, École Polytechnique
FRÉDÉRIC CHAZAL
Geometrica, INRIA
and
MIRELA BEN-CHEN
Technion – Israel Institute of Technology

Vector fields on surfaces are fundamental in various applications in computer graphics and geometry processing. In many cases, in addition to representing vector fields, the need arises to compute their *derivatives*, for example, for solving partial differential equations on surfaces or for designing vector fields with prescribed smoothness properties. In this work, we consider the problem of computing the *Levi-Civita covariant derivative*, that is, the tangential component of the standard directional derivative, on triangle meshes. This problem is challenging since, formally, tangent vector fields on polygonal meshes are often viewed as being discontinuous, hence it is not obvious what a good derivative formulation would be. We leverage the relationship between the Levi-Civita covariant derivative of a vector field and the directional derivative of its component functions to provide a simple, easy-to-implement discretization for which we demonstrate experimental convergence. In addition, we introduce two linear *operators* which provide access to additional constructs in Riemannian geometry that are not easy to discretize otherwise, including the *parallel transport* operator which can be seen simply as a certain matrix exponential. Finally, we show the applicability of our operator to various tasks, such as fluid simulation on curved surfaces and vector field design, by posing algebraic constraints on the covariant derivative operator.

The authors acknowledge the Marie Curie Career Integration Grant CIG-334283-HRGP, the CNRS chaire d' excellence, the Jean Marjoulet chair, a Google Faculty Research Award, ISF grant 699/12, ISF equipment grant, Marie Curie CIG 303511 and the ANR Top-Data (ANR-13-BS01-0008) grant.

Authors' addresses: O. Azencot (corresponding author), Department of Computer Science, Technion-Israel Institute of Technology, Haifa 3200003, Israel; email: azencot@cs.technion.ac.il; M. Ovsjanikov, LIX, Ecole Polytechnique, 91128 Palaiseau Cedex, France; F. Chazal, Geometrica, INRIA, France; M. Ben-Chen, Department of Computer Science, Technion-Israel Institute of Technology, Haifa 3200003, Israel.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling

General Terms: Algorithms

Additional Key Words and Phrases: Geometry processing, discrete differential geometry, vector field analysis

**ACM Reference Format:**

Omri Azencot, Maks Ovsjanikov, Frédéric Chazal, and Mirela Ben-Chen. 2015. Discrete derivatives of vector fields on surfaces – An operator approach. ACM Trans. Graph. 34, 3, Article 29 (April 2015), 13 pages.
DOI: http://dx.doi.org/10.1145/2723158

## 1. INTRODUCTION

Tangent vector fields are ubiquitous in computer graphics. From fluid simulation to texture synthesis, the need to represent vectorial data arises in many applications. Often, it is necessary to compute the *covariant derivative* of a tangent vector field in an arbitrary tangent direction. For example, when simulating fluid flow using Euler equations, the covariant derivative of the fluid's velocity is the main ingredient in the computation of the time evolution of the flow [Taylor 1996]. Furthermore, some vector fields are characterized by the properties of their derivatives: smooth vector fields [Knöppel et al. 2013] minimize the Dirichlet energy, while geodesic vector fields [Pottmann et al. 2010] are constant length and have symmetric covariant derivative operators. Although specific solutions have been tailored to various applications, there currently exist few works on discrete representations of derivatives of tangent vector fields on polygonal meshes which are applicable to general scenarios.

There are two main challenges in deriving such a discretization. First, even on smooth surfaces, defining derivatives of tangent vector fields is more involved than defining derivatives of functions. Specifically, comparing the values of a function at two points on the surface is trivial, but it is not obvious how, given two tangent vectors at different points, one can determine whether they are "the same", since tangent vectors at different points, are expressed with respect to different reference frames. Hence, one needs a way to transport vectors across tangent planes, a construct encoded by a notion of *parallel transport*. Unfortunately, most theoretical treatments of these topics make heavy use of local coordinates, which makes defining discrete analogues for polygonal meshes difficult.

The second challenge is due to the nature of discrete surfaces, namely polygonal meshes, and the way tangent vector fields are represented. The simplest representation, which is the one we opt for, is of piecewise constant vectors on the faces of the mesh. However, in such a representation vector fields are discontinuous across edges, which a priori can lead to difficulties in computing their derivatives. In this article, we formalize this intuition by showing that, for this choice of vector field representation, there exists no definition of a discrete vector field derivative which satisfies all the properties of the continuous Levi-Civita covariant derivative exactly. Faced with these challenges, we propose a novel approach to discretize the Levi-Civita covariant derivative. We compute the *directional derivatives* of the vector field's *component functions* and take the tangential part of the resulting vector field. In the continuous case, it is well known that such a definition yields the unique Levi-Civita covariant derivative [Morita 2001, page 181]. While being intuitive and easy to implement, our approach offers several conceptual benefits. First, by working with functions instead of vector fields, we overcome the difficulty of comparing vectors in different tangent planes. Second, by projecting the component functions on a multiscale basis, we impose some smoothness on the underlying vector field, which allows us to obtain a stable discretization of the Levi-Civita covariant derivative for which we demonstrate experimental convergence. Finally, we derive a representation of the covariant derivative as an operator acting on vector fields. This allows us to design vector fields with various properties, and to define parallel transport without resorting to the computation of discrete flow lines, simply as a matrix exponential.

## 1.1 Related Work

Unlike the discretization of the directional derivatives of functions which can be reduced to computing gradients and is thus well established (e.g., Botsch et al. [2010] and Azencot et al. [2013]), there exists, to the best of our knowledge, no unified treatment of covariant derivatives of vector fields on meshes. Some derived quantities such as the divergence and the curl have received wide attention [Polthier and Preuss 2003; Wardetzky 2006; Hirani 2003; Meyer et al. 2002], whereas the general case we are interested in, the *Levi-Civita covariant derivative of a tangent vector field*, has not been discretized directly. As a full review of the use of derivatives of vector fields in applications is beyond our scope, we mention a few representative examples.

*Discrete calculus frameworks.* There exist several frameworks for geometry processing and graphics applications that provide discretizations of differential quantities. *Discrete exterior calculus* (DEC) [Hirani 2003] is one of the most extensive and widely used, and provides discrete equivalents for vector field operators such as curl, divergence, gradient, and Hodge Laplacian. In addition, DEC provides a strong theoretical foundation in the discrete setting with theorems which mimic the corresponding statements for smooth surfaces. However, not all operators are supported in DEC, and specifically there is currently no consistent discretization of the covariant derivative of vector fields. Other frameworks, such as surface *Finite Element Methods* (FEM) [Dziuk and Elliott 2013] and finite element exterior calculus [Arnold et al. 2006], have also been proposed, but their focus has traditionally been on solving boundary value problems for differential equations. While these approaches have been successfully used to discretize differential operators including the Laplace-Beltrami operator [Wardetzky 2006; Dziuk and Elliott 2013], discretizing arbitrary differential quantities on unstructured meshes remains challenging.

Another approach is to use a global conformal parameterization to the plane [Lui et al. 2005] together with standard FEM to solve a modified problem which takes into account the distortion introduced by the parameterization. Such methods, however, can be sensitive to the large area distortion induced by conformal maps, which may cause many triangles in the planar mesh to collapse, leading to unstable numerical systems.

*Vector field design.* Vector derivatives are often required for vector field design applications. One of the most prominent requirements is that the resulting vector field is sufficiently smooth, and this calls for a way to relate vectors in nearby tangent spaces. On a triangle mesh, two classes of methods have been proposed to quantify smoothness of vector fields. The first is to use discrete 1-forms instead of vector fields, and rephrase the required operators in terms of DEC [Fisher et al. 2007; Ben-Chen et al. 2010], making use in particular of the Hodge Laplacian operator which provides a measure of smoothness for vector fields in a similar way as the Laplace-Beltrami operator does for functions. However, this limits the scope of applications since, for example, it is not clear how to compute the *directional* derivatives of vector fields, and whether various operators (e.g., the symmetric part of the covariant derivative operator) can be represented in DEC.

Another common method to measure smoothness of vector fields is by prescribing a rule on every edge of the mesh, which allows one to compare vectors on the faces across this edge. Perhaps the most natural instance of this approach is to relate vectors on a pair of neighboring triangles by "unfolding" them into a single plane. Indeed, it is customary to refer to this process as the *discrete Levi-Civita connection* (e.g., Crane et al. [2010]), and various comparison rules have been proposed for different applications (among others, Polthier and Schmies [1998], Crane et al. [2010], Pottmann et al. [2010], and Lai et al. [2010]).

However, this general approach has several significant drawbacks. First, these comparison rules only define directional derivatives in the direction of the dual edges of the mesh, and it is not obvious what the derivative should be in a general direction. If we extend this approach to a general direction by following the discrete geodesic in that direction, it is not clear what happens at a vertex. Furthermore, the resulting definition is not stable: a small change in the direction can change the following face on the geodesic path, yielding a different vector and potentially a large change in the derivative. Finally, in many cases the "unfolding" approach is used to define discrete parallel transport, namely a way to transfer a vector between faces on the mesh. Our method provides a more general definition of parallel transport by allowing to transport a vector field on the flow lines of another vector field. Implementing this using the unfolding approach would require numerically integrating the direction vector field to generate the flow lines and then unfolding the triangles along the flow lines, which are both algorithmically complicated and numerically sensitive operations. Using our method we can compute discrete parallel transport simply using a matrix-vector multiplication.

*Fluid simulation.* The directional derivative of a vector field with respect to itself appears in various PDEs, one of them given by the Euler equations for inviscid incompressible flow. Understanding the solutions to these equations is a research field in itself (see, e.g., Batchelor [2000]), thus we only mention some of the more relevant work in computer graphics, and specifically fluid simulation on surfaces. Existing solutions include parameterization-based techniques [Lui et al. 2005], and methods which assume a particular structure on the mesh, for example, by working with subdivision surfaces [Stam 1999]. These methods have the drawbacks of introducing unwanted errors due to the distortion of the parameterization, and the added complexity of converting a general triangle mesh to a subdivision surface. Note that, on a two-dimensional surface, the Euler equations can be reformulated in terms of the *vorticity* of the flow [Nitschke et al. 2012; Elcott et al. 2007], yielding a simpler representation of the velocity through the *stream function*. However, vortex methods have several limitations, for instance, it can be more difficult to set boundary conditions, and therefore in some cases it is preferable to use a velocity-based method. Finally, a method which is tailored for inviscid and incompressible flows on triangle meshes is provided in Shi and Yu [2004]. This method is based on semi-Lagrangian velocity advection on a triangle mesh, which requires tracing velocity flow lines and triangle unfolding that suffer the drawbacks mentioned previously.

## 1.2 Contributions

Our main contribution is a simple yet efficient method for discretizing the Levi-Civita covariant derivative on triangle meshes. We focus on three aspects in our exposition: properties of the discretization, the novel perspective offered by the operator approach, and sample applications. Note that, since we provide a tool and not a specialized application, we focus on proof-of-concept scenarios to illustrate the possibilities associated with our discretization.

In the following sections we discuss our main contributions:

—the discrete formulation of the Levi-Civita covariant derivative, including experimental convergence results (Section 3);

—a representation of the derivative as a linear operator that takes vector fields to vector fields, whose algebraic properties have geometric meaning, for example, exponentiation leads to an algebraic definition of parallel transport (Section 4); and

—several examples demonstrating the applicability of our discrete derivative: vector field design and fluid simulation on surfaces (Section 5).

## 2. DIRECTIONAL DERIVATIVES OF VECTOR FIELDS

Our main goal is to discretize the directional derivative of a vector field on a surface, also known as the Levi-Civita covariant derivative. We will first discuss the definition of such a derivative and its properties in the continuous case. We provide a brief intuitive introduction to the required concepts in this section. Readers well versed in differential geometry can skim these sections and proceed to the discrete treatment in Section 3. As we focus mostly on the geometric intuition behind the definitions, we refer interested readers to Morita [2001, Chapters 5.2 and 5.3] and do Carmo [1992, Chapter 2] for the detailed treatment.

### 2.1 Notation

In the following we denote a surface by $M \subset \mathbb{R}^3$, uppercase letters (e.g., $U, V, W$) denote tangent vector fields, and lowercase letters (e.g., $f, g$) denote real-valued functions. We denote by $\| \cdot \|$ an operator which takes a tangent vector field and outputs a function of its pointwise norms.

### 2.2 The Levi-Civita Covariant Derivative

To gain some intuition, first consider the motion of a particle in the plane, $\mathbb{R}^2$. Its trajectory forms a path $\gamma(t) \in \mathbb{R}^2, t \in \mathbb{R}$, and

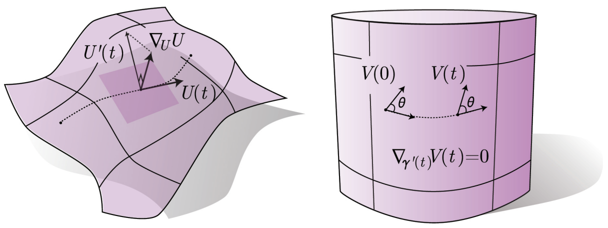

Fig. 1. (left) The velocity $U(t)$ and acceleration $U'(t)$ of a particle traveling along a curve $\gamma(t)$ on a surface, and the tangential component of the acceleration $\nabla_U U$; (right) the parallel transport of $V_0$ along $\gamma$ is the vector field $V(t)$ defined as the unique solution of the differential equation $\nabla_{\gamma'(t)} V(t) = 0$ with $V(0) = V_0$.

its velocity $\gamma'(t) = U(t) \in \mathbb{R}^2$ is a vector tangent to the path. Its acceleration is the vector:

$$U'(t) = \lim_{\Delta t \to 0} \frac{U(t + \Delta t) - U(t)}{\Delta t}. \tag{1}$$

For example, if the trajectory is a straight line and the velocity is not constant, then $U'(t)$ will point in the direction of travel. If the particle travels at constant speed, then the acceleration $U'(t)$ is in a direction orthogonal to the path, since $\langle U(t), U(t)\rangle' = 2\langle U'(t), U(t)\rangle = 0$. Like the velocity, the acceleration vector lies in $\mathbb{R}^2$.

Now, consider the same particle traveling on a curved surface $M \subset \mathbb{R}^3$. Again, its trajectory forms a path $\gamma(t) \in M, t \in \mathbb{R}$, to which its velocity vector $U(t)$ is tangent. However, the acceleration vector $U'(t)$ is no longer tangent to $M$ and decomposes into a component normal to $M$, the *normal acceleration*, and into a component tangent to $M$, the *tangential acceleration* (see Figure 1, left). Intuitively, since the particle is constrained to live on the surface $M$, we can take an intrinsic point of view by considering only the tangential part of the acceleration.

We can similarly compute the tangential component of the derivative of any vector field $V$ defined along a curve, and not necessarily tangent to it, by considering the tangential component of $\lim_{\Delta t \to 0} \frac{V(\gamma(t+\Delta t))-V(\gamma(t))}{\Delta t}$ along the curve $\gamma$. Finally, using the standard $x, y, z$ coordinates in $\mathbb{R}^3$, this definition can be further extended to define the covariant derivative of a tangent vector field $V = (v_x, v_y, v_z)$ on $M$ in a specific direction given by a vector field $U$ on $M$:

$$\nabla_U V(p) = P_p((D_U v_x, D_U v_y, D_U v_z)(p)), \; p \in M, \tag{2}$$

where $P_p$ is the orthogonal projection on the tangent plane to $M$ at $p$ and, for any function $f$, $D_U f =< \nabla f, U >$ denotes the derivative of $f$ in the direction of $U$. Notice that $(D_U v_x, D_U v_y, D_U v_z)(p)$ is a vector in $\mathbb{R}^3$, while $\nabla_U V(p)$ is a tangent vector. The vector field $\nabla_U V$ is known as the *Levi-Civita covariant derivative* of $V$ with respect to $U$ [Morita 2001, page 181].

### 2.3 Parallel Transport

The definition of the covariant derivative is closely related to the notion of parallel transport. Intuitively, parallel transport allows to "carry" a vector along a curve such that it remains "parallel" to itself. For example, the norm of a parallel-transported vector remains fixed, and if the curve is a geodesic then the angle the vector forms with the tangent to the curve also remains fixed. This is formalized using the idea that parallel transport should be the integral of the covariant derivative. Formally, given a curve $\gamma(t)$ in $M$ and a tangent vector $V_0$ at $\gamma(0)$, the parallel transport of $V_0$

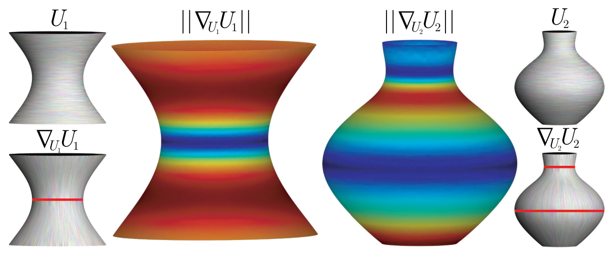

Fig. 2. Constant norm vector fields $U_i$ on a surface of revolution, and their norm $\|\nabla_{U_i} U_i\|$ and flow lines. Note that the norm is zero on the geodesics (marked red), and that the flow lines are orthogonal to $U_i$, since they are constant norm.

along $\gamma$ is defined as the unique solution of the differential equation $\nabla_{\gamma'(t)} V(t) = 0$ with initial condition $V(0) = V_0$ [do Carmo 1992, page 52] see Figure 1 (right).

Before we dive into the properties and the proposed discretization of $\nabla_U V$ we would like to give some intuition as to the quantity we are computing. Consider a surface of revolution like the ones shown in Figure 2, and a constant norm vector field $U$ which is orthogonal to the rotation axis (i.e., it "goes around" the surface). Now consider a particle traveling on the flow lines of $U$ at constant speed. If the flow line is a geodesic, such as the curves marked in red in Figure 2, then traveling at constant speed would yield 0 tangential acceleration. This is seen in the center figures which show the color coding of $\|\nabla_U U\|$. If the particle is not traveling on a geodesic, it has to accelerate to keep "turning". However, since the speed is constant, the acceleration $U'(t)$ would be orthogonal to the direction of travel as seen in the figures showing the flow lines of $\nabla_U U$.

## 2.4 Properties

As we aim for a generic discretization of $\nabla_U V$, which works well in various applications, we would like to assess the properties that are required from such an object. For example, it has been shown in Wardetzky et al. [2007] that, for the Laplace-Beltrami operator and under mild conditions, there is no discretization which fulfills all the defining properties of the continuous operator. In our case, the fundamental theorem of Riemannian geometry guarantees that, if an operator fulfills the following five properties, then it is the *unique* Levi-Civita covariant derivative [do Carmo 1992, page 50 to 55]. Hence, it is of interest to understand these properties and to see whether they are achievable in the discrete case. To make the discussion more concrete, we also denote for each property the application in which it will be required.

*Linearity.* As any derivative, it is a linear operator:

$$\nabla_U(V + W) = \nabla_U V + \nabla_U W. \tag{3}$$

Linearity allows us to represent the operator $\nabla V$ in a basis and construct various energies for vector field design.

*Product rule.*

$$\nabla_U(fV) = f\nabla_U V + V D_U f. \tag{4}$$

Although we do not use this property directly in our applications, the product rule is a fundamental characteristic of any derivative.

*Locality.* The derivative operator is "local" in the direction argument, namely it depends on the value of $U$ at a point, and not on its neighborhood. In other words, if $U_1$ and $U_2$ are vector fields such that $U_1(p) = U_2(p)$ for some point $p$, then $(\nabla_{U_1} V)(p) = (\nabla_{U_2} V)(p)$ for any smooth vector field $V$. This means there are no derivatives of $U$ involved, and therefore this requirement can be rephrased as linearity with respect to functions in the direction argument:

$$\nabla_{fU+gW}(V) = f\nabla_U V + g\nabla_W V. \tag{5}$$

This allows us to represent the operator $\nabla_U$ in a basis, which we use for computing parallel transport.

*Metric compatibility.* This property relates the derivative of a vector field to the derivative of its norm. Similar to the case of a particle in $\mathbb{R}^2$ where we had $\langle V(t), V(t)\rangle' = 2\langle V'(t), V(t)\rangle$, in general, $D_U\langle V, V\rangle = 2\langle\nabla_U V, V\rangle$. Note that, together with linearity, this implies that, for any pair of vector fields $V$ and $W$,

$$D_U\langle V, W\rangle = \langle\nabla_U V, W\rangle + \langle V, \nabla_U W\rangle. \tag{6}$$

*Symmetric Hessian.* Finally, the last property relates to the second derivatives of functions. In the Euclidean case, the Hessian matrix is symmetric since partial derivatives commute. The generalization of the Hessian to the surface is the bilinear operator: $H(f)(U, V) = \langle\nabla_U \nabla f, V\rangle$ [do Carmo 1992, page 142]. The last property requires that this operator is symmetric:

$$\langle\nabla_U \nabla f, V\rangle = \langle\nabla_V \nabla f, U\rangle. \tag{7}$$

A consequence of this property is that $[U, V] = \nabla_U V - \nabla_V U$ for any vector fields $U$ and $V$, where $[\cdot]$ represents the *Lie bracket* operator [do Carmo 1992, page 27]. We use this operator to design local parameterizations.

In the following section we investigate the discretization of the covariant derivative. We first address the question of how vector fields are represented on a mesh, and discuss our choices. Then we consider the challenges for our choice of representation in the discrete setting. We show that, for piecewise constant vector fields, under some mild conditions, it is not possible to define a discrete version of the covariant derivative operator which is both linear and fulfills the metric compatibility property. Finally, we propose a simple approach that is based on the recently introduced multiscale discretization of the directional derivative of functions [Azencot et al. 2013], and we demonstrate experimental convergence of the previously mentioned properties under mesh refinement when both the vector fields and functions are smooth.

## 3. DISCRETIZATION

### 3.1 Vector Field Representation

The definition of a derivative of a vector field is closely linked with the way vector fields are represented in the discrete setting. One option is to use discrete 1-forms [Hirani 2003], which would require using the flat and sharp operators for converting from vector fields to 1-forms and back. Another option is to define a smooth atlas on the mesh through a parameterization of the 1-ring of each vertex (e.g., as in Zhang et al. [2006] and Knöppel et al. [2013]), effectively turning the mesh into a smooth manifold. If a vector field is continuous and piecewise smooth in the atlas, it is possible to define first weak derivatives. Further, recent work by Ray and Sokolov [2013] and Myles et al. [2014] showed how a combinatorial data structure can be used to represent vector fields while ensuring that field flow lines do not merge.

While these options can be a potential starting point for discretizing the covariant derivative, they require a somewhat complicated definition of a discrete vector field. We, on the other hand, choose the most simple discretization of a tangent vector field, namely *piecewise constant on faces*. Such vector fields occur often in applications. For example, scalar functions are often discretized as piecewise linear on the vertices of the mesh, and their gradients are piecewise constant vector fields. Furthermore, in mesh parameterization and mesh quadrangulation applications [Kälberer et al. 2007; Bommes et al. 2009, 2013; Campen et al. 2012; Myles and Zorin 2013; Myles et al. 2014; Panozzo et al. 2014] piecewise constant vector fields are often given as constraints for controlling the alignment of the result. Hence, as we work directly with piecewise constant vector fields without requiring additional conversions to 1-forms or atlas-based representations, our approach is simpler, more intuitive, and easier to implement.

## 3.2 Notation

We represent surfaces with triangle meshes, given by $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, which denote the vertices, edges, and faces, respectively. Functions are represented as piecewise constant on the faces, namely $f : \mathcal{F} \to \mathbb{R}, f = \{f^i, i \in \mathcal{F}\}$. Tangent vector fields are given as piecewise constant on triangles, namely $U : \mathcal{F} \to \mathbb{R}^3, U = \{U^i = (u^i_x, u^i_y, u^i_z), i \in \mathcal{F}\}$, such that $U^i$ is parallel to the plane containing the $i$-th face. Discrete operators are represented with a "tilde", for instance, $\tilde{D}_U : (\mathcal{F} \to \mathbb{R}) \to (\mathcal{F} \to \mathbb{R})$ is the discrete directional derivative for functions and $\tilde{\nabla}_U : (\mathcal{F} \to \mathbb{R}^3) \to (\mathcal{F} \to \mathbb{R}^3)$ is the discrete covariant derivative for vector fields. In what follows, we assume to be given a function $f$ and tangent vector fields $U, V$.

## 3.3 Challenges in the Discrete Setting

As mentioned, we choose to represent vector fields as piecewise constant on the faces. Such a representation, while simple and intuitive, leads to an inherent difficulty in defining a meaningful notion of covariant derivatives since, intuitively, the derivatives of piecewise constant vector fields should be zero at the faces.

Indeed, inside a triangle, taking derivatives of piecewise constant vector fields is futile. Thus, a bigger patch must be taken into account. This, however, would require constructing a mechanism for transporting vectors across triangles. Moreover, it is easy to see that, given the prior discretization of vector fields and functions, the product rule (Eq. (4)), cannot hold exactly for every pair of functions and vector fields. This, however, is true for many notions of discrete derivatives.

Unfortunately, there exists a more fundamental difficulty in discretizing the Levi-Civita covariant derivative, which holds not only for our discretization, but even if functions do not "live on the same domain" as the vector fields, such as functions that are piecewise linear. In particular, even in this case, two of the defining properties of the covariant derivative, namely linearity and metric compatibility, cannot be both satisfied exactly in the discrete setting, under some mild conditions. To state this precisely, since the inner product $\langle U, V\rangle$ produces a function on the faces of the triangle mesh, to allow discrete functions to live on a different domain we can use an averaging operator $\mathcal{A}$ that takes functions on faces and produces functions on vertices, edges, or faces. We will assume that $\mathcal{A}$ is linear, nonnegative, and maps constant functions to constant functions. This leads to the following formulation of the metric compatibility condition:

$$\tilde{D}_X \mathcal{A}(\langle U, V\rangle) = \mathcal{A}(\langle \tilde{\nabla}_X U, V\rangle + \langle \tilde{\nabla}_X V, U\rangle). \tag{8}$$

Here $\tilde{D}_X$ is a directional derivative for functions with respect to the vector field $X$. That is, $\tilde{D}_X$ takes a function defined on some domain (e.g., vertices, edges, or faces) and produces a function defined on the same domain. $\tilde{\nabla}_X U$ is the covariant derivative for vector fields, and the inner product is the standard inner product of vector fields in $\mathbb{R}^3$. Under these conditions, we have the following result (proved in the supplemental material).

LEMMA 1. *If $\tilde{D}_X$ is a linear operator such that $\tilde{D}_X f = 0$ if $f$ is a constant function, and the covariant derivative for vector fields is linear: $\tilde{\nabla}_X(U_1+U_2) = \tilde{\nabla}_X U_1 + \tilde{\nabla}_X U_2$, then the metric compatibility condition (Eq. (8)), implies that $\tilde{D}_X f = 0$ for all $f$ in the range of $\mathcal{A}$, that is, $\tilde{D}_X \mathcal{A}(h) = 0$ for any $h$.*

We note that, although this lemma is stated for vector fields that are constant on the faces, the proof is actually quite general and can be adapted to other settings as well. Hence, as we cannot hope to achieve the exact properties of the smooth covariant derivative, we opt for a simple discretization which is based on the directional derivative of the *component functions*, as given by Eq. (2). Using this definition, it is possible to show that all the properties of the Levi-Civita covariant derivative (except the symmetry of the Hessian) are all consequences of the product rule for functions [Morita 2001, page 181]. Therefore, if the operator $\tilde{D}_U f$ provides a better approximation to the product rule as the mesh resolution increases, so we can expect that the operator $\tilde{\nabla}_U V$ will give a better approximation to Properties 3–6 under mesh refinement, although the metric compatibility condition will never be satisfied exactly.

It has recently been shown in Azencot et al. [2013] that it is possible to discretize the directional derivative of functions $\tilde{D}_U f$ using a *multiscale* basis, such that the error in the product rule property experimentally decreases with the increase in the mesh resolution. We choose a similar discretization for the directional derivative of functions defined on the faces of the mesh, and thus get experimental convergence of the product rule for the component functions of the vector field. This in turn, in the convergence experiments we performed, leads to experimental convergence of the covariant derivative properties.

## 3.4 Directional Derivative of Functions

In the discrete differential geometry literature, functions are commonly discretized either as scalars on the vertices or as scalars on edge midpoints, which are then linearly interpolated to the faces. These are known as conforming and nonconforming linear elements, respectively. In both cases, the gradient operator is well defined as piecewise constant on the faces (see, Wardetzky [2006, Chapter 2] for a full discussion).

Contrary to the common setting, our functions are defined on faces, thus we need to extend the notion of a discrete gradient. Given a function $f$, we define an averaging operator $\mathcal{A}$ and define $\tilde{\nabla} f = \nabla \mathcal{A} f$, where $\mathcal{A}$ averages the values of $f$ to the edges, and $\nabla$ is the discrete gradient for nonconforming elements. Potentially, it is possible to define $\mathcal{A}$ such that it averages values to the vertices instead of edges. However, then $\mathcal{A}$ will be of size $|\mathcal{V}| \times |\mathcal{F}|$, and therefore its range will be smaller than its domain. Thus, there will necessarily be two functions on the faces which are mapped to the same function on the vertices. This will lead to difficulties, as it can introduce nonzero vector fields whose interpolation to the vertices leads to a zero vector field. If, on the other hand, $\mathcal{A}$ averages to the edges, its size is $|\mathcal{E}| \times |\mathcal{F}|$, and therefore the range is larger than the domain and this problem is potentially avoided. It is easy to see that a positive local averaging operator $\mathcal{A}$ will have an empty kernel in general, and in particular for any mesh that has at least one odd-degree vertex (see the proof in the supplemental material).

Formally, we define the directional derivative for functions as

$$\tilde{D}_U f = \langle \nabla \mathcal{A} f, U\rangle, \tag{9}$$

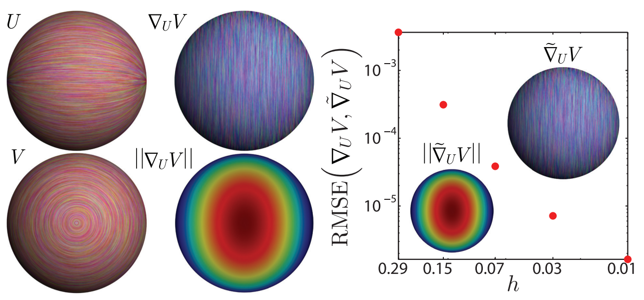

Fig. 3. Comparison of our discretization $\tilde{\nabla}_U V$ with the analytic solution for specific $U$, $V$ on the sphere. We show the convergence graph for the RMSE error for decreasing mean edge length, as well as a visualization of the flow lines and norm of the computed $\tilde{\nabla}_U V$ for the densest mesh.

where $\mathcal{A}_{ij} = w_j / \sum w_k$ if $i$ is an edge in face $j$, and $\mathcal{A}_{ij} = 0$, otherwise. $w_j$ is the area of face $j$ and the sum runs over the faces which share the edge $i$. Now, as $\mathcal{A}f$ is a function on edges, its gradient is piecewise constant per face and has a standard definition (see Polthier [2005, Section 2.3]).

As mentioned previously, we represent the operator $\tilde{D}_U$ in a reduced multiscale basis (the eigenfunctions of the Laplace-Beltrami operator), as this enforces some smoothness on our vector fields.

## 3.5 Covariant Derivative of Vector Fields

Our covariant derivative operator is based on the *extrinsic* definition presented in Eq. (2). Given the discretization for the directional derivative of functions on the faces, the covariant derivative for vector fields follows easily:

$$\tilde{\nabla}_U V(p) = P_p((\tilde{D}_U v_x, \tilde{D}_U v_y, \tilde{D}_U v_z)(p)),\ p \in \mathcal{M}, \tag{10}$$

where $V = (v_x, v_y, v_z)$ and $P_p$ is the projection operator onto the tangent plane of $\mathcal{M}$ at $p$. As the directional derivatives of the components of $V$ are given on the faces, $P_p$ is well defined.

To summarise, given two piecewise constant vector fields $U$ and $V$, we first take the component coordinate functions of $V$, average them onto the edges, and compute the corresponding gradients. These are piecewise constant on the faces, therefore their inner products with $U$ give us three real-valued functions on the faces. We use these functions to construct a vector field in $\mathbb{R}^3$, and project this vector field onto the faces.

To validate our discretization, we experiment with known vector fields $U, V$ on the unit sphere and compare our result with the expected result in the continuous setting. Figure 3 shows the result of this comparison for meshes with decreasing average edge length $h$. We show $U, V$, the analytic result $\nabla_U V$, and the result of our computation $\tilde{\nabla}_U V$. Note that the convergence is polynomial in $h$ and that, for the most dense mesh, the figures of the flow lines and norm are almost indistinguishable from the ground truth. We further demonstrate the convergence results in Figure 4 which shows the log log plot of the RMSE error of Properties 4–7 for ellipsoid meshes with decreasing average edge length $h$. We additionally show the vector fields $U, V, W$ and the functions $f, g$ which were used for the mesh with smallest edge length. The functions $f, g$ are the eighth and tenth eigenfunctions of the area weighted cotangent Laplace-Beltrami operator and the vector fields $U$, $V$, $W$ correspond to eigen 1-forms 4, 3, and 1 of the Hodge Laplacian. Note that the plot suggests a polynomial convergence rate in $h$, where we denote by $m$ the respective slope estimate. Furthermore, given Eq. (10), it is easy to verify that Property 3 holds exactly.

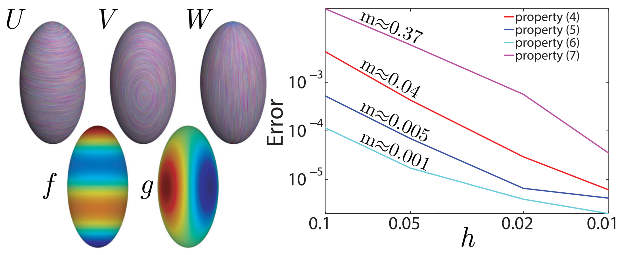

Fig. 4. The behavior of our discretization of the covariant derivative on the Properties 4–7 under mesh refinement for the ellipsoid model. We show the RMSE error of the left-hand side vs. the right-hand side of the equation for decreasing mean edge length $h$. Note that the plot suggests a polynomial convergence rate in $h$, where we denote by $m$ the respective slope estimate. We additionally show the functions and vector fields that were used for the highest mesh resolution. See the text for further details.

## 4. GEOMETRY FROM LINEAR OPERATORS

In addition to computing the quantity $\tilde{\nabla}_U V$, it is often advantageous to fix one of the vector fields and consider the corresponding operator on all possible inputs. For example, we can omit the direction $U$ and consider the operator $\tilde{\nabla} V$, which will provide some information on the derivatives of $V$ in all possible directions. This point of view is useful because it can uncover some hidden structure of $V$ in a global way. As a simple example, the singular vector of $\tilde{\nabla} V$ which corresponds to the smallest singular value will provide the directions in which $V$ changes as little as possible.

This interplay between the algebraic properties of the operators and the geometry of the vector fields they represent is quite useful in practice, because it allows to do global, operations which are traditionally local. For example, manipulating $\tilde{\nabla} V$ is instrumental for vector field design, and $\tilde{\nabla}_U$ allows to easily compute parallel transport.

### 4.1 Preliminaries

*Matrix representation.* While it is possible to analyze these operators directly as abstract linear operators, it is more intuitive to consider their matrix representation. Specifically, we assume, we have a finite orthonormal basis of vector fields $\{\Psi_i, i \in 1, \dots, k\}$, that is, $\int_M \langle \Psi_i, \Psi_j \rangle = 1$ if $i = j$ and 0 otherwise, and such that the vector fields in which we are interested can be represented as $V = \sum_{i=1}^k a_i \Psi_i$ (in Section 5.1 we will elaborate more on our choice of basis). Now, any linear operator $\mathcal{R}$ from tangent vector fields to tangent vector fields can be represented using a $k \times k$ matrix $R$, whose $(i, j)$ entry is $R_{i,j} = \int_M \langle \mathcal{R}(\Psi_i), \Psi_j \rangle$. In the following we will discuss the properties of the operators using their matrix representations. For example, when we mention the operator transpose, we refer to the corresponding matrix transpose.

*Flow of a vector field.* We will need the following definition. The *flow* of a vector field $U$ is a one-parameter family of maps $\Phi_U^t : M \to M$ for $t \in \mathbb{R}$ such that the following holds:

$$\frac{d}{dt}\Phi_U^t(p) = U\left(\Phi_U^t(p)\right), \qquad \Phi_U^0(p) = p.$$

Intuitively, the flow of a vector field encodes what happens to a particle which starts at a point $p \in M$, and its velocity is dictated by the vector field at every point. Hence, it provides a way to recover the trajectory of a particle from its velocity, and thus computing the flow is also known as *integrating* the vector field.

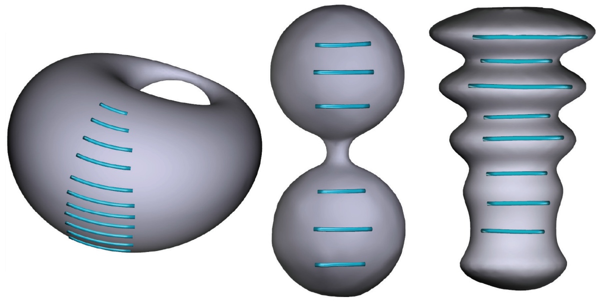
Fig. 5. Approximate killing vector fields computed by minimizing the symmetric part of $\nabla V$.

## 4.2 The Operator $\nabla V$

*Operator action:* $(\nabla V)(U) = \nabla_U V$. Here $V$ is fixed, and we compute its derivative in some direction given as input. This operator is the extension to surfaces of the Jacobian operator of vector fields in Euclidean space, which is simply the matrix of partial derivatives. Its algebraic structure provides information about the nature of the derivatives of $V$ in various directions. For example, as any linear operator, it can be decomposed into symmetric and anti-symmetric parts

$$\nabla V = \frac{1}{2}\left(\nabla V + (\nabla V)^T\right) + \frac{1}{2}\left(\nabla V - (\nabla V)^T\right) = K_V + G_V,$$

where, as discussed previously, we consider the operator as a $k \times k$ matrix representation and thus can compute its transpose. The symmetric and anti-symmetric parts are also linear operators which take tangent vector fields to tangent vector fields and have geometric meaning.

*Symmetric part.* The operator $K_V = \frac{1}{2}(\nabla V + (\nabla V)^T)$ is related to how much the flow $\Phi_V^t$ distorts the metric. Specifically, if $K_V = 0$, then $V$ is called a *killing vector field* (KVF), and its flow $\Phi_V^t$ is an isometry for all $t$ Petersen [2006, Chapter 7.1]. One such example in the plane is $V = (-y, x)$, whose flow is simply a global rotation. Such vector fields are quite rare and exist only on very specific surfaces, however, we can try to minimize $\|K_V\|^2$ for any surface, yielding vector fields whose flow is close to an isometry. Such vector fields are useful in geometry processing applications, as they allow to generate texture and geometric patterns [Ben-Chen et al. 2010].

We use this property to design vector fields which are approximate KVFs by solving a linear system of equations. Note that, as opposed to previous work, we can pose the constraints directly on the derivative operator, without requiring an indirect approach through commutativity with the Laplace-Beltrami operator [Azencot et al. 2013], or reformulation using DEC [Ben-Chen et al. 2010]. Figure 5 shows a few approximate killing vector fields computed this way. Interestingly, KVFs are also related to fluid flow on surfaces, as they provide a steady-state solution to the Euler equations (see Section 5). Furthermore, the killing operator $K_V$ plays a role in the behavior of viscous fluids [Nitschke et al. 2012], which we would like to investigate in future work.

*Anti-symmetric part.* The operator $G_V = \frac{1}{2}(\nabla V - (\nabla V)^T)$ encodes the failure of $\nabla V$ to be symmetric. We know from Property 7 that if $V = \nabla f$ for some function $f$ then $\nabla V$ is symmetric, hence it is possible to consider $G_V$ as the failure of $V$ to be the gradient of a function. Specifically, minimizing $\|G_V\|^2$ with some additional conditions would provide vector fields which are "as gradient as possible". For example, if we require that $\|V\| = const$ it is possible to show that the flow lines of $V$ are geodesics and $V$ is a *geodesic vector field* (GVF) if and only if $G_V = 0$ [Pottmann et al. 2010], which can be useful in architectural geometry. In the applications section we demonstrate how, by constraining $\nabla V$ to be symmetric, then in addition to the smoothness induced by our framework, we can, using a much simpler setup, achieve similar results, even without adding the constraint on the norm of $V$. Furthermore, our approach allows to combine various constraints, for instance, that the resulting vector field is symmetric with respect to some symmetry map of the surface.

*Uniqueness.* As we discussed, we can design vector fields $V$ which have certain properties by posing constraints (e.g., symmetry or anti-symmetry) on $\nabla V$. This raises the question whether a given $\nabla V$ completely encodes $V$, or whether there can be multiple vector fields with the same $\nabla V$. We have the following lemma.

LEMMA 2. *For a closed oriented surface $M$, $\nabla_U V = 0$ for every smooth $U$ if and only if $V = 0$ or $M$ is a flat torus.*

Hence, if $\nabla V_1 = \nabla V_2$ then $\nabla_U(V_1 - V_2) = 0\ \forall\ U$ which, by the lemma, implies that $V_1 = V_2$, yielding the uniqueness we required.

## 4.3 The Operator $\nabla_U$

*Operator action:* $(\nabla_U)(V) = \nabla_U V$. Here the direction of the derivative is given by a fixed $U$, and we compute the derivative of some vector field $V$ given as input. This operator is closely related to the directional derivative of functions, which we denoted as $D_U$. The scalar directional derivative operator was recently used by Azencot et al. [2013] to represent, analyze, and design discrete vector fields. While this approach is useful in certain applications, it is also limited, since the scalar directional derivative operator $D_U$ does not depend on the metric of the surface, making the computation of metric-dependent operations such as the parallel transport of vector fields impossible without additional structure. As we show shortly, the Levi-Civita covariant derivative, acting on vector fields, shares many useful properties with the functional operator such as uniqueness and decomposition, but also enables more applications including parallel transport in a very compact and convenient manner.

*Uniqueness.* The operator $\nabla_U$ encodes the vector field $U$ uniquely. Hence we can design a vector field $U$ by defining constraints on $\nabla_U$. We have the next lemma.

LEMMA 3. *Two smooth vector fields $U$ and $V$ are equal if and only if $\nabla_U W = \nabla_V W$ for all smooth vector fields $W$.*

*Symmetric part.* The operator $\nabla_U$ allows to easily distinguish divergence-free vector fields as those whose symmetric part of $\nabla_U$ is zero.

LEMMA 4. *Let $M$ be a closed surface. A smooth vector field $U$ is divergence free if and only if $\nabla_U$ is anti-symmetric with respect to the inner product on the surface, that is, if and only if $\int_M \langle \nabla_U V, W\rangle dx = -\int_M \langle \nabla_U W, V\rangle dx$ for all smooth vector fields $V$ and $W$.*

*Parallel transport.* The Levi-Civita covariant derivative, represented as an operator $\nabla_U$, is intimately related to parallel translation along the flow lines of $U$. Suppose we have a vector field $V$ and let $\Phi_U^t(p)$ be the flow of $U$. Now, consider the operator $\Gamma_{U,t}$ which takes a vector field on $M$ and returns a vector field on $M$, which is defined as follows: $\Gamma_{U,t}(V)(p)$ is the vector obtained by parallel transporting the vector $V(\Phi_U^t(p))$ along the flow line from $\Phi_U^t(p)$ to $p$. It is well known (e.g., do Carmo [1992, page 57]) that the following relation between the operators $\nabla_U$ and $\Gamma_{U,t}$ holds:

$$\nabla_U(V)(p) = \frac{d}{dt}(\Gamma_{U,t}(V)(p))|_{t=0}. \tag{11}$$

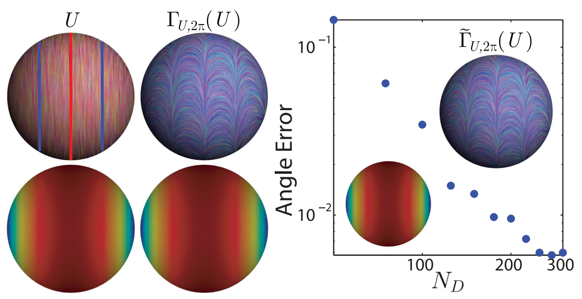

Fig. 6. Parallel transport of a vector field $U$ (left) along its own flow lines comparison to the ground truth on the sphere (middle). Note the three marked singularity curves: the red curve is a geodesic, so vectors transported on it preserve their orientation. The blue curves are two symmetric singularity curves. The vectors transported on them rotate by $\pi$, so they reverse their orientation. The transition between these singularity curves is smooth; (right) convergence graph of the error in the computed angle, and the final result of our computation for the largest number of basis functions.

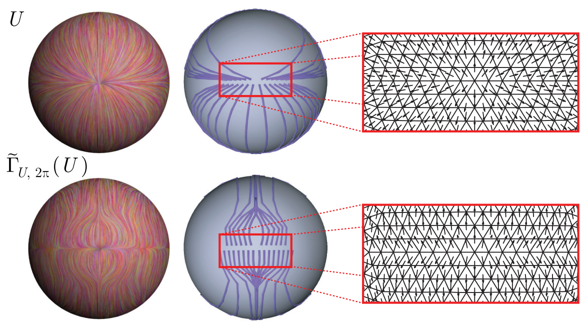

Fig. 7. Parallel translation of $U$ (top row) along the flow lines of $U$. Our discrete parallel transport is robust to merging flow lines, as shown in the result, $\tilde{\Gamma}_{U,2\pi}$ (bottom row).

Hence the $\nabla_U$ operator is the derivative of the backward parallel transport operator at the point $p$. Now, if we consider the discrete version of (11), that is, replace $\nabla_U$ and $\Gamma_{U,t}$ with their discrete matrix-based representations $\tilde{\nabla}_U$ and $\tilde{\Gamma}_{U,t}$, respectively it is easy to check (see supplemental material) that $\tilde{\Gamma}$, given by

$$\tilde{\Gamma}_{U,t} = \exp(t\tilde{\nabla}_U), \tag{12}$$

where exp is the matrix exponentiation, is a solution. By defining $\tilde{\Gamma}_{U,t}$ as in (12) we maintain the relation between the discrete parallel transport and covariant derivative operators which exists in the continuous case, and gain an easy-to-implement matrix-based operator.

This observation allows to compute the parallel transport of vector fields, along the flow lines of other vector fields, simply by using the matrix exponential of $\tilde{\nabla}_U$. This is somewhat remarkable since computing discrete parallel transport on discrete flow lines directly would require to numerically integrate the field $U$ to generate the flow lines, and then compute the discrete geodesic curvature of these flow lines for the transport, such as done in Polthier and Schmies [1998]. This procedure can be cumbersome, computationally heavy, and potentially numerically unstable. For example, the result may not even be a well-defined vector field with multiple vectors in a single face, and some faces not containing any vectors.

On the other hand, when considering the Levi-Civita covariant derivative as an operator acting on vector fields, and representing it as a matrix in a basis, computing parallel transport becomes a standard linear algebra operation involving only matrix exponent and matrix vector multiplication. Note that parallel transporting a vector field $U$ along its own flow lines is closely related to the numerical scheme known as *semi-Lagrangian advection* in fluid simulation [Shi and Yu 2004]. It is therefore possible that our parallel transport matrix operator could be used in such a setup. We leave further investigation of this direction as future work.

In Figure 6 we compare the result of parallel transport done using our approach to the ground truth on the sphere. We take a vector field $U = (0, z, -y)$ which rotates around the sphere, and compute $\tilde{\Gamma}_{U,2\pi}(U)$, the parallel transport of $U$ over itself for time $t = 2\pi$, by taking $\exp(2\pi\tilde{\nabla}_U)U$. In this case, the flow lines are constant latitude lines, and the result of the parallel transport has an analytic expression [do Carmo 1976, page 243].

Figure 6 shows the vector field $U$ (left) and the ground-truth result (center). Our parallel transport operator uses a fixed number of basis vectors, and the parallel-transported vector field is nonsmooth, therefore we expect the result to improve with an increasing number of basis vectors. This is indeed demonstrated in the graph on the right. The graph shows the error in our computation of the angle of the parallel-transported vector field $\tilde{\Gamma}_{U,2\pi}(U)$ with $U$ with respect to using a growing number of basis vectors $N_D$. The two figures in the graph show the flow lines and the norm of $\tilde{\Gamma}_{U,2\pi}(U)$ for the largest number of basis functions. Interestingly, the norm of the parallel-transported vector field can be flown separately using the flow of the operator for functions $\tilde{D}_U$, which leads to more accurate results. Note that the resulting norm and angles are almost indistinguishable from the ground truth.

We provide further evaluation of our discrete parallel transport. It is known that discrete flow lines of vector fields can in some cases merge or split (e.g., Szymczak and Zhang [2012, Figure 4]). In Figure 7 we demonstrate the result of parallel translation of $U$ (top row) along $U$. Notice that, although the flow lines of $U$ might split (see the zoomed area, top, right), our result, $\tilde{\Gamma}_{U,2\pi}(U)$, preserves its smooth behavior.

While matrix exponentiation is itself a difficult problem and the result can be inaccurate for large matrices [Moler and van Loan 2003], note that in our case the matrices are relatively small (on the order of 300), as the vector field is represented a multiscale basis. In our implementation we used Matlab's *expm* function and did not encounter any issues. Furthermore, to compute the parallel transport there is no need to compute the full matrix exponent, but only the matrix vector product $\exp(2\pi\tilde{\nabla}_U)U$ for which more stable and efficient methods exist [Al-Mohy and Higham 2011]. It is possible that more basis vector fields would be required to represent complex vector fields with a large number of singularities, which are common in parameterization and quadrangular remeshing applications. In such cases, it might be instrumental to investigate our operator in the hat basis, which will lead to a sparse representation for which methods such as Al-Mohy and Higham [2011] are still applicable. We leave further study in this direction for future work.

## 4.4 The Operator $[U, \cdot]$

*Operator action:* $[U, \cdot](V) = \nabla_U V - \nabla_V U$. Given two vector fields $U, V$, consider the problem of constructing local texture coordinates $(u, v)$ such that the iso-$v$ and iso-$u$ lines align with $U$ and $V$, respectively. Given $p \in M$, one naïve approach would be to flow along $U$ from $p$ and sample the flow line at fixed constant intervals. Then, starting from the resulting sampled points, we flow along $V$ and sample again. The union of the sampled points forms a grid. Of course, we could reverse the order and flow first on $V$ and then

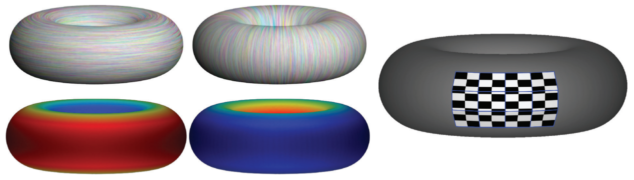

Fig. 8. Given a vector field $U$ (left), we construct local parameterization by optimizing for $V$ (middle) which minimizes the energy $\int_M \|[U, V]\|^2 + \lambda \int_M \| < U, V > \|^2$. The local coordinates are computed by flowing on $U$ and $V$, resulting in a texture-mapped grid marked in blue (right).

on $U$, however, we expect to obtain the same set of sampled points. Formally, this requirement means that the flows of $U$ and $V$ should commute.

The operator $[U, V]$ known as the *Lie bracket* or *Lie derivative* of $U$ and $V$, computes exactly this property: the lack of commutativity of the flows of $U$ and $V$. Specifically, it is possible to construct a local parameterization as described previously around a point $p \in M$ if and only if $U(p)$, $V(p)$ form a basis for the tangent plane and $[U, V] = 0$ (see, e.g., Kolar [1993, Theorem 3.17]).

Using the operators $\nabla_U$ and $\nabla U$ we can represent $[U, \cdot]$, and use it for vector field design. For example, given a vector field $U$, we can construct a matrix representation of $[U, \cdot]$ and compute its singular vectors. Since $[U, U] = 0$, $U$ is always the singular vector corresponding to the 0 singular value. However, the next singular vector $V$ minimizes $\int_M \|[U, V]\|^2$, and would give us the best vector to couple with $U$ to get a parameterization. Note that we can easily add additional terms to the energy, such as $\int_M \| < U, V > \|^2$, if we want $U$ and $V$ to be orthogonal.

Figure 8 demonstrates this for the computation of a local parameterization. We are given $U$ (left), and we minimize the energy $E_{[U,\cdot]}(V) = \int_M \|[U, V]\|^2 + \lambda \int_M \| < U, V > \|^2$. The resulting vector field $V$ (middle) together with $U$ is used to build the local coordinates using the flow method described previously. This yields a textured-mapped grid (right, shown in blue). Note that the vector fields $U$, $V$ are orthogonal but do not have the same norm. Hence, simply rotating $U$ by $\pi/2$ would not have given the same texture coordinates, as the flows would not necessarily commute.

## 5. APPLICATIONS

Until now we have concentrated on the properties of the various operators we can derive from the Levi-Civita covariant derivative, and provided some proof-of-concept applications for the geometric quantities it allows to compute. In this section, we first discuss some implementation details and limitations, and then discuss two concrete applications of this machinery: designing tangent vector fields and simulating fluid flow on surfaces.

### 5.1 Implementation Details

*Choice of basis.* For our basis for $\tilde{D}_U$, we chose the first $N_f$ eigenvectors of the DEC-based 2-form Hodge Laplacian [Hirani 2003]. For $\tilde{\nabla}_U$, $\tilde{\nabla} V$ and all operators acting on vector fields, the basis is given by the first $N_D$ eigenvectors of the DEC-based 1-form Hodge Laplacian [Fisher et al. 2007]. To represent our operators as matrices in the basis, we first convert the 1-forms to piecewise constant vector fields (as in Fisher et al. [2007, Eq. (4)], where we sample at the barycenter of the triangle), then apply the operator to the basis elements and project the result back onto the basis.

*Limitations.* We define the covariant derivative using the embedding in $\mathbb{R}^3$, however, a classical and fundamental property of the covariant derivative in the continuous case is that it is intrinsic, that is, it does not depend on this embedding [Morita 2001, page 181]. In the discrete case, we no longer maintain this property. For rigid deformations, there exists a trade-off between invariance and discretization error. If we use a small number of basis functions, the component functions are smooth, but we lose invariance to rigid transformations. However, the error introduced by the rigid transformation decreases polynomially in the number of basis functions. If, on the other hand, we use the full basis in Eq. (9), the operator will be invariant to rigid transformations (see supplemental material for the proof). For isometric deformations, the averaging operator $\mathcal{A}$ introduces some error even when using the full basis (as it causes averaging of vectors on faces which undergo different rotations), and for a truncated basis we again have an error which decreases polynomially. Despite this limitation, we believe the additional simplicity we gain by using the embedding is worthwhile, especially in applications which use a single nondeforming mesh.

### 5.2 Vector Field Design

As discussed in the previous sections, by using the covariant derivative operators, we can pose various constraints to design tangent vector fields with some prescribed differential properties. Since the operators $\tilde{\nabla}_U$ and $\tilde{\nabla} V$ are linear, each of the optimization problems that we formulate can be solved efficiently by solving a linear system, or by computing a singular value decomposition.

*As-gradient-as-possible vector fields.* We first consider minimizing the energy $\|\tilde{\nabla} V - (\tilde{\nabla} V)^T\|^2$, which quantifies the anti-symmetric part of $\tilde{\nabla} V$. As mentioned in Section 4.2, this energy will be zero if $V$ is a gradient field. Furthermore Pottmann et al. [2010] showed that if additionally the norm of $V$ is constant, then the energy will be zero only if $V$ is a vector field whose flow lines are geodesics, also known as a *geodesic vector fields* (GVFs).

While we do not impose the additional constraint, our results on the Oloid model as shown in Figure 9 are comparable to the results of Pottmann et al. [2010] when weighing the edges according to their mean curvature is not taken into account.

Finally, as we work in the generic framework of functional operators, it is straightforward to combine this energy with additional constraints in a similar manner to Azencot et al. [2013]. For example, we can require the vector field to be symmetric with respect to some symmetry map provided for the surface. By weighing differently the constraints, we can allow the user to explore multiple solutions (see Figure 10) which may be difficult to achieve using other frameworks.

*As-killing-as-possible vector fields.* As mentioned previously, vector fields $V$ for which $\tilde{\nabla} V$ is anti-symmetric are vector fields whose flow preserves the metric, also known as *killing vector fields* (KVFs). These are useful for pattern generation, as shown, for instance, in Ben-Chen et al. [2010]. By minimizing the energy $\|\tilde{\nabla} V + (\tilde{\nabla} V)^T\|^2$, we can construct vector fields that are as close as possible to KVFs, as we demonstrate in Figure 5.

*Smooth vector fields.* As our last design goal we consider the task of computing as-smooth-as-possible vector fields, similarly to what was done in Knöppel et al. [2013]. One way to characterize such vector fields is to minimize the Dirichlet energy $\|\tilde{\nabla} V\|^2$. Figure 11 shows an example of two vector fields computed this way, and Figure 12 compares the vector field computed using our method (left) with that computed by the approach of Knöppel et al. [2013]

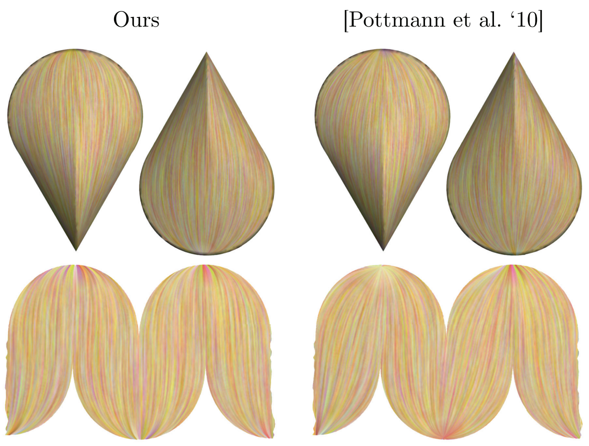

Fig. 9. Approximate geodesic vector field design. We seek a vector field $V$ which minimizes the energy $\|\nabla V - (\nabla V)^T\|^2$, which yields $V$ that is close to a geodesic vector field (top left). The Oloid model has zero Gaussian curvature everywhere except on the creases, hence when it is flattened the flow lines should yield straight lines (bottom left). Compare with the result of Pottmann et al. [2010] (right). Our results are comparable, while our setup is considerably simpler and allows for combination of constraints.

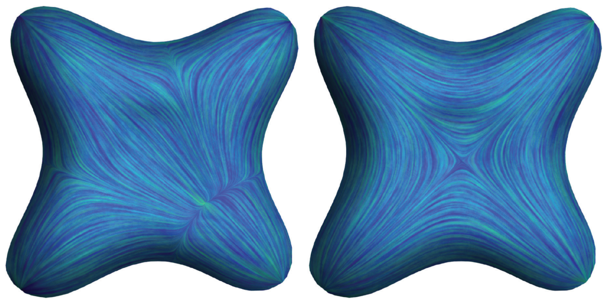

Fig. 10. Trade-off between as-gradient-as-possible vector field constraints and symmetric vector field constraints, with the symmetry constraints weighted higher in the image on the right.

(right). Note that the resulting vector fields are comparable in terms of smoothness. Compared to the ground truth on the unit sphere, the Dirichlet energy obtained by Knöppel et al. [2013] is more accurate than ours (1.0017 versus 0.9515, where the analytic solution is 1), potentially due to energy loss incurred by our projection on the basis of vector fields. Furthermore, the method by Knöppel et al. [2013] is more general than ours, as it can handle N-RoSy fields in addition to vector fields.

To conclude, while there exist other specialized methods for posing many of the design constraints mentioned here, such as Azencot et al. [2013], Pottmann et al. [2010], Knöppel et al. [2013], and Ben-Chen et al. [2010], our setup is unique in that it is simple, allows to pose all of these constraints, and to generate a large variety of vector fields, since we have direct access to the $\nabla V$ and $\nabla_U$ operators.

## 5.3 Fluid Simulation on Surfaces

As our last application, we consider the problem of simulating the behavior of an incompressible flow on a curved surface. A fluid can

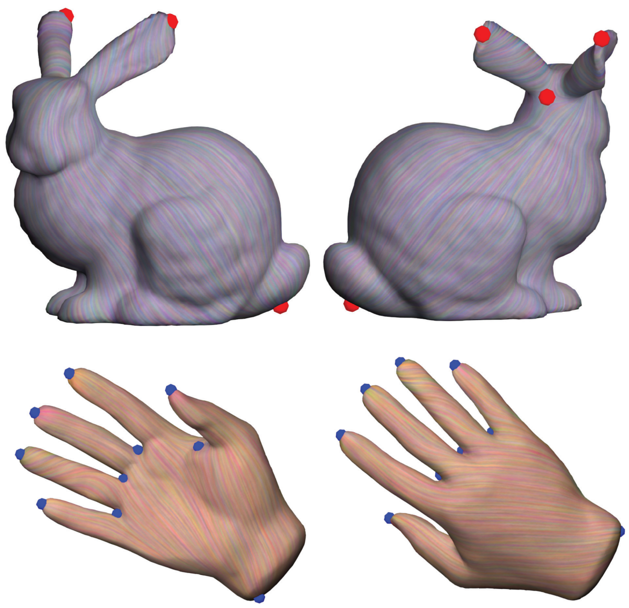

Fig. 11. Designing smooth vector fields by finding vector fields which minimize the energy $\|\tilde{\nabla} V\|^2$.

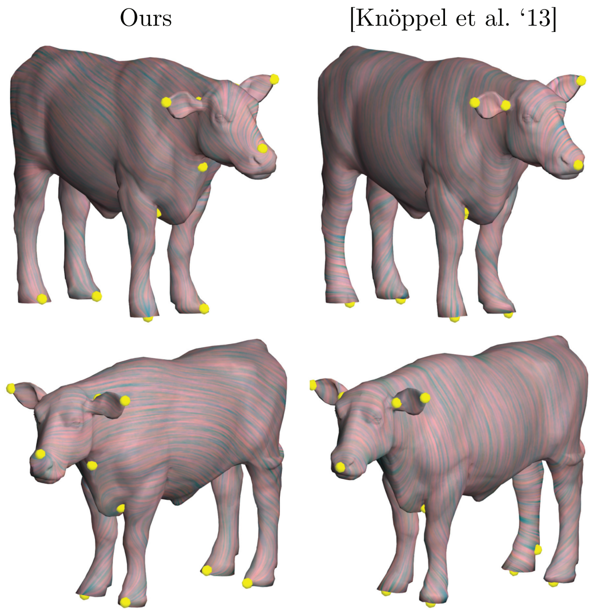

Fig. 12. Our smooth vector field (left) compared to the one obtained by the method of Knöppel et al. [2013] (right).

be described as a time-varying velocity field $U(t)$, whose behavior is governed by the Navier–Stokes equations [Taylor 1996]. We discuss here only incompressible (divergence-free) inviscid (viscosity-free) flows, for which the defining equations are known as the Euler equations Taylor [1996, Eq. (1.10)]:

$$\frac{\partial U}{\partial t} = -P_{curl}(\nabla_U U), \tag{13}$$

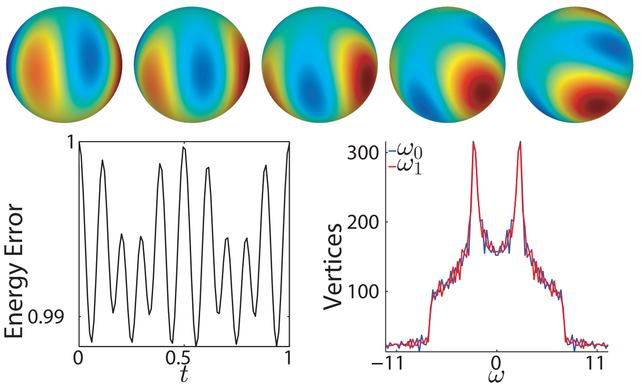

Fig. 13. (top) A few frames from a periodic solution of the Euler equations on the sphere. Note that the vorticity (color coded) is globally rotated, as expected. See the text for details; (bottom, left) the relative kinetic energy $\int_M \|U(t)\|/\int_M \|U(0)\|$ during the simulation. Note that it is periodic and remains within 98% of the original energy; (bottom, right) a histogram of the vorticity for the first (blue) and last (red) frames. Note that the histogram is preserved as expected.

where $P_{curl}$ is the orthogonal projection onto the space of divergence-free vector fields.

Using our discrete definition of the covariant derivative, it is straightforward to compute the time-varying velocity $U(t)$ of a flow, given some initial conditions. We implemented a very simple pipeline using a black-box time integrator (Matlab's ode45 [Dormand and Prince 1980]). One iteration consists of computing $\tilde{\nabla}_U U$ using our operator, followed by projection onto the space-divergence-free vector fields by solving the Poisson equation $\Delta s = -\omega$, where $\omega$ is the vorticity function given by the curl of $U$, projected onto the space of functions spanned by our basis. The change in $U$ is now given by the gradient of $s$ rotated by $\pi/2$ in each face. We use the operator from Polthier and Preuss [2003] for computing the curl of a vector field.

Despite the simplicity of this approach, we found that in most cases it was enough to simulate interesting flows for which we know the analytic solution or expected behavior. We demonstrate some examples in the accompanying video for the simulation of the flows. We stress that this is a proof of concept of the applicability of our operator to fluid simulation on surfaces. We leave further tuning, as well as incorporating a more sophisticated time integrator, as future work.

*Steady-state solutions.* If $U$ is a killing vector field, or $U = J\nabla\phi_i$, where $\phi_i$ is an eigenfunction of the Laplace-Beltrami operator, then $U(t) = U$ is a steady-state solution to Eq. (13) (see Majda and Bertozzi [2001, page 46, Eq. (2.13)], and also the supplemental material for a simple proof). Hence, as a sanity check we compute the average of $\|P_{curl}(\tilde{\nabla}_U U)\|/\|U\|$ for such a vector field $U$. The result can be considered an indicator to the stability of our method, and was on the order of $10^{-4}$ for the unit sphere.

*Periodic solution on the sphere.* On the sphere there exists a periodic time-varying solution, given by $U(t) = U_0 + \sum_i a_i(t) J\nabla\phi_i$, where $U_0$ is a killing vector field, and $\phi_i$ are eigenfunctions of the Laplace-Beltrami operator corresponding to the same eigenvalue. Furthermore, the curl of the velocity field (its vorticity) $\omega(t)$ is advected by this flow isometrically, namely a pure rotation. We are not aware of a reference for this solution in the literature, and thus provide the proof in the supplemental material.

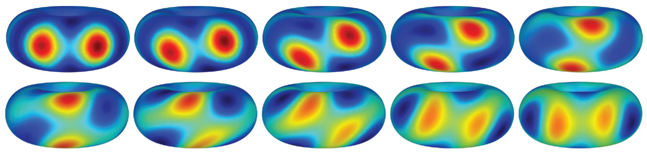

Fig. 14. A few frames from a solution of the Euler equations on the torus for a co-rotating vortex pair.

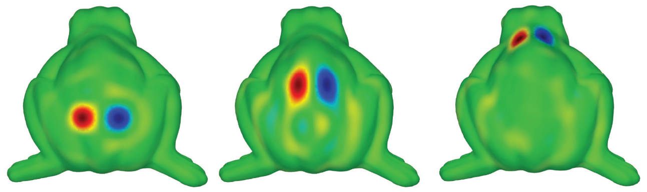

Fig. 15. Three frames from a fluid flow simulation showing a positive/negative vortex pair on a surface.

Figure 13 (top) shows a few frames from such a simulation on the unit sphere, where we took $\phi_i$ to be an eigenfunction in the third group of spherical harmonics. We show the color coding of the vorticity function, which is indeed advected as an isometry. Figure 13 (bottom right) shows the relative kinetic energy $\int_M \|U(t)\|/\int_M \|U(0)\|$ during the simulation. Note that the energy itself exhibits periodic behavior and remains within 98% of the original energy. This indicates the stability of our method, especially since we used a straightforward black-box time integrator for all simulations. Finally, Figure 13 (bottom left) shows a histogram of the vorticity values for the first and last frames of the simulation. Note that the histogram remains fixed, as expected.

*Co-rotating vortex pair.* On a plane, a pair of *point vortices* (namely, singular points where all the vorticity is concentrated) spinning in the same direction should rotate around each other Saffman [1992, page 117]. We generate a similar configuration on a torus, where we take the initial vorticity $\omega_0$ to be constant at all vertices except two vertices $v_i, v_j$, where we take $\omega_0$ to be 1. The constant is set such that $\int \omega_0 = 0$, and then $\omega$ is projected onto the span of our basis functions. Figure 14 shows a few frames from this simulation (see also the accompanying video). Note that the vortices rotate as expected. One limitation of our method is that it is not circulation preserving as is, for example, the method in Elcott et al. [2007]. This is visible in the torus simulation, as some of the vorticity is lost due to numerical dissipation. We leave the exploration of efficient methods to overcome this limitation as future work.

*Counter-rotating vortex pair.* Similarly to the previous experiment, we take two point vortices rotating in opposite directions. In the plane such a configuration translates in a straight line Saffman [1992, page 117] and a similar behavior is demonstrated on the back of the frog model in Figure 15 and in the accompanying video that can be accessed in the ACM Digital Library. The stability of our method is exhibited by the fact that the vortex pair travels intact the whole length of the frog model.

*N-vortex structures.* Here we take a more complicated configuration of vortices. The first includes two pairs of counter-rotating vortices which collide, where the expected behavior is that they continue in a direction orthogonal to the original direction after collision. This is shown in Figure 16 and in the accompanying

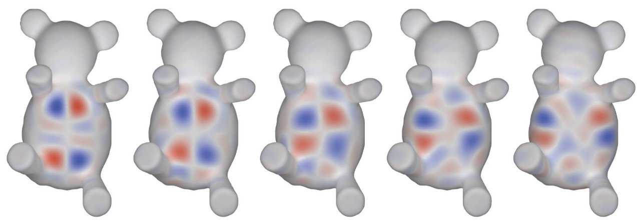

Fig. 16. A few frames from a solution of the Euler equations on the teddy bear for two colliding pairs of counter-rotating vortices.

video on the teddy bear model. The second configuration includes three co-rotating vortices forming an equilateral triangle, where the flow should rotate the three vortices as a single unit Newton [2001, page 78]. We reproduce this behavior, as can be seen in the video. Note that, while two of the vortices merge during the process, they separate again at the end of the flow, returning to a configuration similar to the original one.

## 6. CONCLUSIONS AND FUTURE WORK

In this article, we proposed a novel discretization for the Levi-Civita covariant derivative of vector fields on discrete surfaces, which has various appealing properties. First, it exhibits experimental convergence of the five defining properties of the derivative in the continuous case. Second, it can be represented as a linear operator acting on tangent vector fields, thus allowing to harness tools from linear algebra, such as matrix exponential, to perform geometric operations which were otherwise harder to achieve (e.g., parallel transport of a vector field along the flow lines of another vector field). Finally, we demonstrated the applicability of our discretization to various geometry processing tasks such as local parameterization, vector field design, and fluid simulation.

We believe there is much more left to explore, as we only gave a taste of the possible applications of our formulation. First, the covariant derivative appears in many PDEs on surfaces, and it would be interesting to apply our discretization to additional problems. For example, it is possible to compute the covariant derivative of the normal vector field, thus yielding a novel discretization of the shape operator. Second, our parallel transport approach can potentially be applied to fluid flow simulation to yield a more stable exponential integrator, and the killing operator can be used for simulating viscous flow. Furthermore, we would like to investigate additional operators derived from the covariant derivative, such as the connection Laplacian which can potentially be used for vector field smoothing. To conclude, we believe our discrete covariant derivative will inspire future work that tackles additional challenges in vector field processing, thus providing a stepping stone towards a complete framework for vector calculus on discrete surfaces.

## ELECTRONIC APPENDIX

The electronic appendix to this article is available in the ACM Digital Library.

## ACKNOWLEDGMENTS

Meshes provided by the Stanford Computer Graphics Laboratory, the AIM@SHAPE Shape Repository, Zaha Hadid Architects, London and SHREC.

## REFERENCES

A. H. Al-Mohy and N. J. Higham. 2011. Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM J. Sci. Comput.* 33, 2, 488–511.

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

O. Azencot, M. Ben-Chen, F. Chazal, and M. Ovsjansikov. 2013. An operator approach to tangent vector field processing. *Comput. Graph. Forum* 32, 73–82.

G. K. Batchelor. 2000. *An Introduction to Fluid Dynamics*. Cambridge University Press.

M. Ben-Chen, A. Butscher, J. Solomon, and L. Guibas. 2010. On discrete Killing vector fields and patterns on surfaces. In *Proceedings of the EUROGRAPHICS Symposium on Geometric Processing (SGP'10).* Vol. 29. 1701–1711.

D. Bommes, M. Campen, H.-C. Ebke, P. Alliez, and L. Kobbelt. 2013. Integer-grid maps for reliable quad meshing. *ACM Trans. Graph.* 32, 4.

D. Bommes, H. Zimmer, and L. Kobbelt. 2009. Mixed-integer quadrangulation. *ACM Trans. Graph.* 28, 77.

M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Levy. 2010. *Polygon Mesh Processing*. AK Peters.

M. Campen, D. Bommes, and L. Kobbelt. 2012. Dual loops meshing: Quality quad layouts on manifolds. *ACM Trans. Graph.* 31, 4.

K. Crane, M. Desbrun, and P. Schroder. 2010. Trivial connections on discrete surfaces. *Comput. Graph. Forum* 29, 1525–1533.

M. P. Do Carmo. 1976. *Differential Geometry of Curves and Surfaces*. Prentice-Hall.

M. P. Do Carmo. 1992. *Riemannian Geometry.* Birkhauser.

J. R. Dormand and P. J. Prince. 1980. A family of embedded Runge-Kutta formulae. *J. Comput. Appl. Math.* 6, 1, 19–26.

G. Dziuk and C. M. Elliott. 2013. Finite element methods for surface PDEs. *Acta Numerica* 22, 289–396.

S. Elcott, Y. Tong, E. Kanso, P. Schroder, and M. Desbrun. 2007. Stable, circulation-preserving, simplicial fluids. *ACM Trans. Graph.* 26, 1.

M. Fisher, P. Schroder, M. Desbrun, and H. Hoppe. 2007. Design of tangent vector fields. *ACM Trans. Graph.* 26, 3.

A. N. Hirani. 2003. Discrete exterior calculus. Ph.D. thesis, California Institute of Technology. http://thesis.library.caltech.edu/1885/3/thesis_hirani.pdf.

F. Kalberer, M. Nieser, and K. Polthier. 2007. Quadcover-surface parameterization using branched coverings. *Comput. Graph. Forum* 26, 375–384.

F. Knoppel, K. Crane, U. Pinkall, and P. Schroder. 2013. Globally optimal direction fields. *ACM Trans. Graph.* 32, 4, 59:1–59:10.

I. Kolar. 1993. *Natural Operations in Differential Geometry*. Springer.

Y.-K. Lai, M. Jin, X. Xie, Y. He, J. Palacios, E. Zhang, S.-M. Hu, and X. Gu. 2010. Metric-driven RoSy field design and remeshing. *IEEE Trans. Visual. Comput. Graph.* 16, 1, 95–108.

L. M. Lui, Y. Wang, and T. F. Chan. 2005. Solving PDEs on manifolds with global conformal parameterization. In *Variational, Geometric, and Level Set Methods in Computer Vision*, Springer, 307–319.

A. J. Majda and A. L. Bertozzi. 2001. *Vorticity and Incompressible Flow*, Vol. 27, Cambridge University Press.

M. Meyer, M. Desbrun, P. Schroder, and A. H. Baar. 2002. Discrete differential-geometry operators for triangulated 2-manifolds. In *Proceedings of the VisMath Conference (VisMath'02).* 35–57.

C. Moler and C. Van Loan. 2003. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Rev.* 45, 1, 3–49.

S. Morita. 2001. *Geometry of Differential Forms*. American Mathematical Society, Providence, RI.

A. Myles, N. Pietroni, and D. Zorin. 2014. Robust field-aligned global parameterization. *ACM Trans. Graph.* 33, 4.

A. Myles and D. Zorin. 2013. Controlled-distortion constrained global parameterization. *ACM Trans. Graph.* 32, 4, 105:1–105:14.

P. K. Newton. 2001. *The N-Vortex Problem: Analytical Techniques*. Vol. 145, Springer.

I. Nitschke, A. Voigt, and J. Wensch. 2012. A finite element approach to incompressible two-phase flow on manifolds. *J. Fluid Mechan.* 708, 418.

D. Panozzo, E. Puppo, M. Tarini, and O. Sorkine-Hornung. 2014. Frame fields: Anisotropic and non-orthogonal cross fields. *ACM Trans. Graph.* 33, 4.

P. Petersen. 2006. *Riemannian Geometry*. Springer, New York.

K. Polthier. 2005. Computational aspects of discrete minimal surfaces. *Global Theory Minim. Surf.* 2, 65–111.

K. Polthier and E. Preuss. 2003. Identifying vector field singularities using a discrete Hodge decomposition. *Visual. Math.* 3, 113–134.

K. Polthier and M. Schmies. 1998. Straightest geodesics polyhedral surfaces. In *Mathematical Visualization*, Springer, 135–150.

H. Pottmann, Q. Huang, B. Deng, A. Schiftner, M. Kilian, L. Guibas, and J. Wallner. 2010. Geodesic patterns. *ACM Trans. Graph.* 29, 3.

N. Ray and D. Sokolov. 2013. Tracing cross-free polylines oriented by a n-symmetry direction field on triangulated surfaces. http://arxiv.org/pdf/1306.0706.pdf.

P. G. Saffman 1992. *Vortex Dynamics*. Cambridge University Press.

L. Shi and Y. Yu. 2004. Inviscid and incompressible fluid simulation on triangle meshes. *Comput. Animat. Virtual Worlds* 15, 3–4, 173–181.

J. Stam. 1999. Stable fluids. In *Proceedings of the 26$^{th}$ Annual ACM SIGGRAPH Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'99).* 121–128.

A. Szymczak and E. Zhang. 2012. Robust Morse decomposition of piecewise constant vector fields. *IEEE Trans. Visual. Comput. Graph.* 18, 6, 938–951.

M. Taylor 1996. *Partial Differential Equations*, *Vol. III.* Springer.

M. Wardetzky. 2006. Discrete differential operators on polyhedral surfaces – Convergence and approximation. Ph.D. thesis, Freie Universitat Berlin.

M. Wardetzky, S. Mathur, F. Kalberer, and E. Grinspun. 2007. Discrete Laplace operators: No free lunch. In *Proceedings of the Symposium on Geometry Processing (SGP'07).* 33–37.

E. Zhang, K. Mischaikow, and G. Turk. 2006. Vector field design on surfaces. *ACM Trans. Graph.* 25, 4, 1294–1326.

Received July 2014; revised December 2014; accepted January 2015