DOI: 10.1111/cgf.12174
Eurographics Symposium on Geometry Processing 2013
Yaron Lipman and Hao Zhang
(Guest Editors)

Volume 32 (2013), Number 5

An Operator Approach to Tangent Vector Field Processing

Omri Azencot[1] and Mirela Ben-Chen[1] and Frédéric Chazal[2] and Maks Ovsjanikov[3]

[1]Technion - Israel Institute of Technology
[2] Geometrica, INRIA
[3]LIX, École Polytechnique

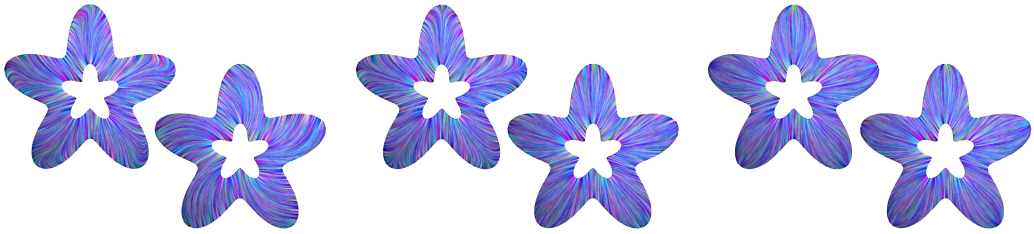


Figure 1: *Using our framework various vector field design goals can be easily posed as linear constraints. Here, given three symmetry maps: rotational (S1), bilateral (S2) and front/back (S3), we can generate a symmetric vector field using only S1 (left), S1 + S2 (center) and S1 + S2 + S3 (right). The top row shows the front of the 3D model, and the bottom row its back.*

Abstract

In this paper, we introduce a novel coordinate-free method for manipulating and analyzing vector fields on discrete surfaces. Unlike the commonly used representations of a vector field as an assignment of vectors to the faces of the mesh, or as real values on edges, we argue that vector fields can also be naturally viewed as operators *whose domain and range are functions defined on the mesh. Although this point of view is common in differential geometry it has so far not been adopted in geometry processing applications. We recall the theoretical properties of vector fields represented as operators, and show that* composition *of vector fields with other functional operators is natural in this setup. This leads to the characterization of vector field properties through commutativity with other operators such as the Laplace-Beltrami and symmetry operators, as well as to a straight-forward definition of differential properties such as the Lie derivative. Finally, we demonstrate a range of applications, such as Killing vector field design, symmetric vector field estimation and joint design on multiple surfaces.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—

1. Introduction

Manipulating and designing tangent vector fields on discrete domains is a fundamental operation in areas as diverse as dynamical systems, finite elements and geometry processing. The first question that needs to be addressed before designing a vector field processing toolbox, is how will the vector fields be represented in the discrete setting? The goal of this paper is to propose a representation, which is inspired by the point of view of vector fields in differential geometry as *operators* or *derivations*.

In the continuous setting, there are a few common ways of defining a tangent vector field on a surface. The first, is to consider a smooth assignment of a vector in the tangent space at each point on the surface. This is, perhaps, the most intuitive way to extend the definition of vector fields from the Euclidean space to manifolds. However, it comes with a price, since on a curved surface one must keep track of the relation between the tangent spaces at different points. A natural discretization corresponding to this point of view (used e.g. in [PP03]) is to assign a single Euclidean vector to each simplex of a polygonal mesh (either a vertex or a face), and to extend them through interpolation. While this representation is clearly useful in many applications, the non-trivial relationships between the tangent spaces complicate tasks such as vector field design and manipulation.

An alternative approach in the continuous case, is to work with *differential forms* (see e.g. [Mor01]) which are linear operators taking tangent vector fields to scalar functions. In the discrete setting this point of view leads to the famous Discrete Exterior Calculus [Hir03, FSDH07], where discrete 1-forms are represented as real-valued functions defined over the edges of the mesh. While this approach is coordinate-free (as no basis for the tangent space needs to be defined), and has many advantages over the previous method, there are still some operations which are natural in the continuous setting, and not easily representable in DEC. For example, the flow of a tangent vector field is a one parameter set of self-maps and various vector field properties can be defined by composition with its flow, an operation which is somewhat challenging to perform using DEC.

Finally, another point of view of tangent vector fields in the continuous case is to consider their *action* on scalar functions. Namely, for a given vector field, its *covariant derivative* is an operator that associates to any smooth function f on the manifold another function which equals the derivative of f in the direction given by the vector field. It is well known that a vector field can be recovered from its covariant derivative operator, and thus any vector field can be uniquely represented as a functional operator. We will refer to these operators as functional vector fields (FVFs). Note, that while this point of view is classical in differential geometry, it has so far received limited attention in geometry processing.

In this paper, we argue that the operator point of view yields a useful coordinate-free representation of vector fields on discrete surfaces that is complementary to existing representations and that can facilitate a number of novel applications. For example, we show that constructing a Killing vector field [Pet97] on a surface can be done by simply finding a functional vector field that commutes with the Laplace-Beltrami operator. Furthermore, we show that it is possible to transport vector fields across surfaces, find symmetric vector fields and even compute the flow of a vector field easily by employing the natural relationship between FVFs and functional maps [OBCS*12]. Finally, the Lie derivative of two vector fields is given by the commutator of the two respective operators, and as a result the covariant derivative of a tangent vector field with respect to another can be computed through the Koszul formula [Pet97].

To employ this representation in practice, we show that for a suitable choice of basis, a functional vector field can be represented as a (possibly infinite) matrix. As not all such matrices represent vector fields, we show how to parameterize the space of vector fields using a *basis for the operators*. With these tools in hand, we propose a Finite Element-based discretization for functional vector fields, and demonstrate its consistency and empirical convergence. Finally, we apply our framework to various vector field processing tasks showing comparable results to existing methods, as well as novel applications which were challenging so far.

1.1. Related Work

The body of literature devoted to vector fields in graphics, visualization and geometry processing is vast and a full overview is beyond our scope. Thus, we will focus on the *representation* and discretization of vector fields, as these aspects of vector field processing are most related to our work.

One approach to discretization (e.g. [PP03, TLHD03]) is to use piecewise constant vector fields, where vectors are defined per face and represented in the standard basis in $\mathbb{R}^3$. This approach is simple and allows to define discrete versions of standard operators such as *div* and *curl*, which are consistent with their continuous counterparts (e.g. one can define a discrete Hodge decomposition [PP03]). However, since the representation is based on coordinate frames, it makes vector field design challenging as the relationship between tangent spaces is non-trivial, leading to difficult optimization problems.

An alternative discretization of vector fields was suggested as part of the formalism of Discrete Exterior Calculus (DEC) [Hir03], where vector fields are identified with discrete 1-forms, represented as a single scalar per edge. This approach is inherently coordinate-free, allowing to formulate vector field design as a linear system [FSDH07]. Unfortunately, computing the Lie derivative of vector fields remains a complex task using DEC (as shown in [MMP*11]).

Vector field design and processing applications are also tightly connected to the analysis of rotationally symmetric (RoSy) fields, see e.g. [PZ07, RVAL09, CDS10]. In the latter work, for example, a vector field (or a symmetric direction field) is represented using an angle per edge (an angle valued dual 1-form), which represents how the vector changes between neighboring triangles. While these approaches are also coordinate-free and lead to linear optimization problems for direction field design, it is not clear how vector-field valued operators can be represented in such a setup.

In this paper, we argue that in addition to the existing discretization methods, it is often useful to represent vector fields through their covariant derivatives as linear functional operators. This representation is coordinate-free and, in addition, elucidates the intimate connection between vector fields and self maps of the surface, allowing us to extend the basic vector field processing toolbox to computational tasks which are challenging using existing discretization tools.

Note that the operator representation of vector fields has been used in the context of fluid simulation by Pavlov et al. [PMT*11]. However, in that work, the authors were primarily interested in representing divergence-free vector fields and did not use this representation for tangent vector field analysis and design. In this paper, we consider general vector fields, demonstrate how this representation can be used for vector field processing, and show a deep connection with the functional map framework [OBCS*12].

1.2. Contributions

Our main observation is that tangent vector fields can be represented in a coordinate-free way as functional operators. While this view is classical in differential geometry [Mor01], it has so far received limited attention in geometry processing. Using this perspective we:

- Show how functional vector fields can be naturally composed with other operators, and thus relate vector fields to other common operators such as maps between shapes and the Laplace-Beltrami operator (Section 2).
- Provide a novel coordinate-free discretization of tangent vector fields (Section 4).
- Describe various applications for vector field processing including Killing vector field design, design of symmetric vector fields and joint vector field design on multiple shapes, which are all easily solvable as linear systems in our framework (Section 5).

2. Vector Fields as Operators

In this section we define the coordinate-free view of vector fields as abstract *derivations* of functions in the continuous setting. This point of view is well-known in differential geometry (see e.g. [Mor01] for an excellent reference). Thus, we only recall the standard definition and its main properties.

2.1. The Covariant Derivative of Functions

We first assume that we are given a compact smooth Riemannian manifold M and a tangent vector field V, which can be thought of as a smooth assignment of a tangent vector $V(p)$ to each point $p \in M$. The vector field defines a one-parameter family of maps, $\Phi_V^t : M \to M$ for $t \in \mathbb{R}$, called the *flow* of V. The flow is formally defined as the unique solution to the differential equation:

$$\frac{d}{dt}\Phi_V^t(p) = V(\Phi_V^t(p)), \qquad \Phi_V^0(p) = p. \tag{1}$$

Then, for a given function $f \in C^\infty(M)$, the *covariant derivative* $D_V(f)$ of f with respect to V is a function g, which intuitively measures the change in f with respect to the flow under V. Formally,

$$g(p) = D_V(f)(p) = \lim_{t\to 0} \frac{f(\Phi_V^t(p)) - f(p)}{t}.$$

A classical result in Riemannian geometry ([Mor01], p. 148) is that the covariant derivative can also be computed as :

$$D_V(f)(p) = g(p) = \langle (\nabla f)(p), V(p) \rangle_p, \tag{2}$$

where $\langle , \rangle_p$ denotes the inner product in the tangent space of p, and ∇f is the gradient of f (see Figure 2).

2.2. The Covariant Derivative as a Functional Operator

We stress that D_V is best viewed as an *operator*, which maps smooth functions on M to smooth functions on M. Moreover,

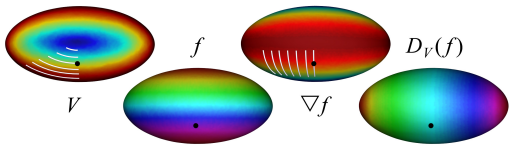


Figure 2: *Given a vector field V (left) and a function f (center left), the inner product of ∇f (center right) with V is the covariant derivative $D_V(f)$ (right). For the marked point, for example, V is orthogonal to ∇f, yielding 0 for $D_V(f)$. Vector fields are visualized by color coding their norm, and showing a few flow lines for a fixed time t.*

one can show that D_V encodes V so that if V_1 and V_2 are vector fields such that $D_{V_1} f = D_{V_2} f$ for any $f \in C^\infty(M)$, then $V_1 = V_2$ (see [Mor01], p.38). Said differently, there is no loss of information when moving from V to D_V.

The covariant derivative (viewed as a functional operator, i.e. an FVF) satisfies the following two key properties:

Linearity:

$$D(\alpha f + \beta g) = \alpha D(f) + \beta D(g), \tag{3}$$

and Leibnitz (product) rule:

$$D(fg) = fD(g) + gD(f). \tag{4}$$

Conversely, a functional operator D corresponds to a vector field, if and only if it satisfies (3) and (4) (see [Spi99] p. 79).

Why are these the necessary properties for operators that represent vector fields? Intuitively, this is because vector fields can be thought of as first order directional derivatives, which have two essential properties. First, that *constant functions are mapped to the zero function*. And second, that $D_V(f)$ depends on f *only to first order*.

One of the advantages of considering vector fields as abstract derivations is that this point of view can be generalized to settings where differential quantities are not well defined. For example, on a discrete surface there is no well defined normal direction at vertices and edges. By working with purely algebraic constructs, such as linear operators, we can define differentiation without requiring the concept of a limit, which is useful when the underlying surface is not continuous and such a limit does not exist. Moreover, as we will see, the operator point of view makes it easier to manipulate vector fields and relate them to other functional operators.

2.3. Properties

While the operator point of view is equivalent to the standard notion of a vector field as a smooth assignment of tangent vectors, certain operations are more natural in this representation. Below we list such operations, which we will use in our applications in Section 5. The proofs of all lemmas are provided in the supplemental material.

Operator composition. By using the operator point of view

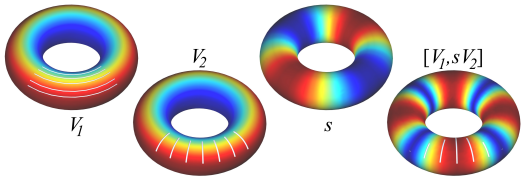


Figure 3: *Two orthogonal vector fields on the torus V_1, V_2, whose Lie derivative is 0. Modifying the norm of V_2 using a function s yields a lie derivative which is parallel to V_2.*

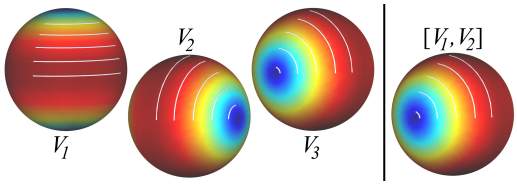


Figure 4: *Using commutativity with L, we compute the KVFs on the sphere (V_1, V_2, V_3). Alternatively, we compute $V_4 = [V_1, V_2]$, note the similarity of V_3 and V_4.*

of vector fields, it becomes easy to define their *composition* both with other vector fields and other more general functional operators. Unfortunately, given two vector fields D_{V_1} and D_{V_2}, the operator $D_{V_1} \circ D_{V_2}$ does not necessarily correspond to a vector field. However, one can modify this operator to obtain a fundamental notion:

Lie derivative of a vector field. Given two vector fields V_1 and V_2, the Lie derivative (or Lie bracket) of V_2 with respect to V_1 is a vector field V_3 defined as:

$$\mathcal{L}_{V_1}(V_2) = [V_1, V_2] = D_{V_3} = D_{V_1} \circ D_{V_2} - D_{V_2} \circ D_{V_1}. \quad (5)$$

It is easy to see that D_{V_3} thus defined is both linear and satisfies the product rule. Hence, D_{V_3} corresponds to a unique vector field V_3. Intuitively, the Lie derivative captures the commutativity of the flows of V_1 and V_2. In particular, the Lie derivative is zero if and only if the flows defined by V_1 and V_2 commute (see [Spi99], p.157):

$$\Phi_{V_2}^{-s} \circ \Phi_{V_1}^{-t} \circ \Phi_{V_2}^{s} \circ \Phi_{V_1}^{t} = Id \quad \forall s,t \in \mathbb{R} \quad (6)$$

Figure 3 illustrates the computation of the Lie derivative on a torus. We consider two vector fields V_1 and V_2 whose flows commute. The average norm of $[V_1, V_2]$ computed using the discrete operators we describe in Section 4 is on the order of 1e-8, close to 0 as expected. In general, if $[V_1, V_2] = 0$, it can be shown that for any scalar function $s : M \to \mathbb{R}$, $[V_1, sV_2]$ must be parallel to V_2. In Figure 3, we show a scaling function s, and the computed vector field $V_3 = [V_1, sV_2]$, which is parallel to V_2, as expected.

Composition with other operators. Of course, it is possible to consider the composition of the FVF operator D_V with other functional operators. Interestingly, the commutativity of D_V with a differential operator D is closely related to the commutativity of its flow with D.

Lemma 2.1 Let T_F^t, $t \in \mathbb{R}$ be the functional operator representations of the flow diffeomorphisms $\Phi_V^t : M \to M$ of V, defined by $T_F^t(f) = f \circ \Phi_V^t$ for any function $f \in C^\infty(M)$. If D is a linear partial differential operator then $D_V \circ D = D \circ D_V$ if and only if for any $t \in \mathbb{R}$, $T_F^t \circ D = D \circ T_F^t$.

For example, on a Riemannian manifold, we can consider composition with the Laplace-Beltrami operator L. The commutativity of D_V with L is then closely related to the metric distortion imposed by the flow of V. In particular, recall that a vector field is called a Killing vector field (KVF) if Φ_V^t is an isometry for all t (see [Pet97], Chapter 7). Then:

Lemma 2.2 A vector field V is a Killing vector field if and only if $D_V \circ L = L \circ D_V$.

From this lemma, it is easy to see that KVFs form a group under the Lie derivative. Indeed, the following lemma, which follows directly from the definition of the Lie derivative, is useful in general:

Lemma 2.3 Given two vector fields D_{V_1} and D_{V_2} that both commute with some operator D, the Lie derivative $\mathcal{L}_{V_1}(V_2)$ will also commute with D.

Figure 4 demonstrates these properties on the sphere. On the left, we show V_1, V_2, V_3, the three KVFs of the sphere, computed using Lemma 2.2 by constructing a linear system (as explained in Section 5). On the right, we show $V_4 = [V_1, V_2]$, which was computed as the Lie bracket of the first two KVFs. Note the similarity between V_3 and V_4. We will use these results for designing approximate KVFs in Section 5.

Composition with mappings. In many settings we are also interested in the relation between vector fields on multiple surfaces related by mappings. In particular, given a vector field V_1 on surface M and a *diffeomorphism* $T : M \to N$, one can define the vector field V_2 on surface N via the push forward: $V_2(q) = dT(V_1(T^{-1}(q)))$. Note that in the discrete case, it is often difficult to compute the differential dT of a map T between shapes with different discretizations. At the same time, one can equivalently define the vector field V_2 using the operator approach, without relying on dT, by using the functional representation of the map T.

As mentioned in [OBCS*12], the functional representation T_F of a map T is a linear operator on the space of functions, taking functions on N to functions on M defined by $T_F(g) = g \circ T$ for any function $g \in C^\infty(N)$. This means that the functional vector field D_{V_2}, and thus V_2 itself can be obtained by simple composition of three linear functional operators without the need to estimate the differential dT, using:

Lemma 2.4 $D_{V_2} = (T_F)^{-1} \circ D_{V_1} \circ T_F$.

Figure 5 illustrates vector field transportation using this approach (vector fields are visualized using [PZ11]). Given V_1 on M, and a map $T : M \to N$, we generate V_2 on N using Lemma 2.4. V_3 is computed using the differential of the map,

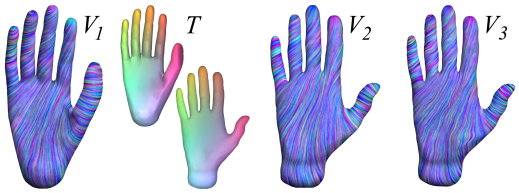


Figure 5: *Given a vector field V_1 on M and a map $T : M \to N$, we generate a vector field V_2 on N using Lemma 2.4. Compare with directly transporting V_1 using the differential of the map, yielding V_3. Note the ringing artifacts in V_3.*

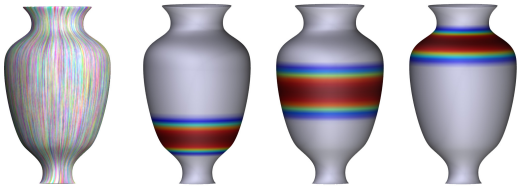


Figure 6: *Applying the flow of a vector field (left) to a function (center left) using Lemma 2.5. (center right, right) The function after the flow, for two sample t values.*

given by the affine map between corresponding triangles. Note that V_3 tends to be noisy, due to the locality of the transport procedure, as opposed to the smoother V_2. Furthermore, this approach is applicable to shapes with different connectivity, where computing dT is challenging. In Section 5 we use a similar approach to *design* vector fields which are consistent with the map $T : M \to N$.

Vector field flow. The FVF D_V representing a vector field is also closely related to the functional representation of the flow Φ_V^t. In particular:

Lemma 2.5 Let $T^t = \Phi_V^t$ be the self-map associated with the flow of V at time t. Then if T_F^t is the functional representation of T^t, for any real analytic function f (see [DFN92], p. 210):

$$T_F^t f = \exp(t\, D_V) f = \sum_{k=0}^{\infty} \frac{(tD_V)^k f}{k!}.$$

This lemma is particularly useful since it allows to avoid the potentially costly solution of the system of equations (1) and directly estimate the functional representation of the map Φ_V^t through operator exponentiation. Note that D_V is a moderately sized matrix when represented in a basis, and therefore its exponent can be computed efficiently. Figure 6 shows an example of function flow using this method.

Covariant derivative of a tangent vector field. Some PDEs can be described using the *covariant derivative* [Mor01] of a vector field V_1 with respect to another vector field V_2, denoted $\nabla_{V_2} V_1$. For planar vector fields, for example, $\nabla_{V_2} V_1 = J(V_1)V_2$, where $J(V_1)$ is the Jacobian matrix of V_1.

On a surface, however, this representation requires a basis for the tangent space at every point, and a suitable *connection* that allows to transport a vector $V(p)$ to a neighboring point q, which makes $\nabla_{V_2} V_1$ elusive to compute in a coordinate-free way. Fortunately, there is an intimate connection between the Lie and covariant derivatives of vector fields, through the Koszul formula, ([Pet97], p. 25):

$$\begin{aligned} 2g(\nabla_{V_1} V_2, Z) = {} & D_{V_1}(g(V_2,Z)) - g(V_1,[V_2,Z]) \\ & + D_{V_2}(g(V_1,Z)) - g(V_2,[V_1,Z]) \\ & - D_Z(g(V_1,V_2)) + g(Z,[V_1,V_2]). \end{aligned} \tag{7}$$

Here, Z is an arbitrary vector field, $g(\cdot,\cdot) = \langle \cdot,\cdot \rangle_p$ is the inner product in the tangent space of p, and $[\cdot,\cdot]$ is the Lie bracket (Eq. 5). Hence, given an operator representation of D_{V_1} and D_{V_2}, we can use Equation (7) to compute $\nabla_{V_1} V_2$. We leave further investigation of this direction, and possible applications for future work.

3. Representation in a Basis

The properties mentioned above suggest that representing and analyzing tangent vector fields through their functional representation can enable a number of applications which are challenging using standard methods. Our goal, therefore, will be to represent this operator such that it can be easily analyzed and manipulated in practice.

3.1. Basis for the Function Space

As mentioned in Section 2.2, an FVF is a linear operator acting on smooth functions defined on the manifold. In practice, we will assume that the functional space of interest can be endowed with a (possibly infinite) basis, so that any function can be represented as a linear combination of some basis functions $\{\phi_i\}$. Then, for any given function $f = \sum_i a_i \phi_i$, we have that $g = D_V(f) = D_V(\sum_i a_i \phi_i) = \sum_i a_i D_V(\phi_i)$. Since $D_V(\phi_i)$ is also a function, it can be represented in the basis as $D_V(\phi_i) = \sum_j D_{ij}\phi_j$. Therefore, $g = \sum_j (\sum_i D_{ij} a_i)\phi_j = \sum_j b_j \phi_j$. In other words, if one thinks of the coefficients a_i, b_i as vectors $\mathbf{a}, \mathbf{b}$ and $D = (D_{ij})$ as a matrix, then the transformation between the basis representations of f and $g = D_V(f)$ is given by: $\mathbf{b} = D\mathbf{a}$.

When the basis functions ϕ_i are orthonormal with respect to the standard functional inner product on M, i.e. $\int_M \phi_i \phi_j d\mu = 1$ if $i = j$ and 0 otherwise, then the $(i,j)^{\text{th}}$ element D_{ij} of the FVF corresponding to V is given by:

$$D_{ij} = \int_M \phi_i D_V(\phi_j) d\mu(p) = \int_M \phi_i(p) \left\langle V(p), \nabla \phi_j \right\rangle_p d\mu(p), \tag{8}$$

where $\langle , \rangle_p$ denotes the inner product in the tangent space of the point p, and $d\mu(p)$ represents the volume measure at p.

The Laplace-Beltrami basis. A useful basis for the space of smooth functions on a compact manifold, which we will often use in practice, is the basis given by the eigenfunctions of the Laplace-Beltrami operator (note that on a compact manifold the space $L^2(M)$ is strictly larger than the space of

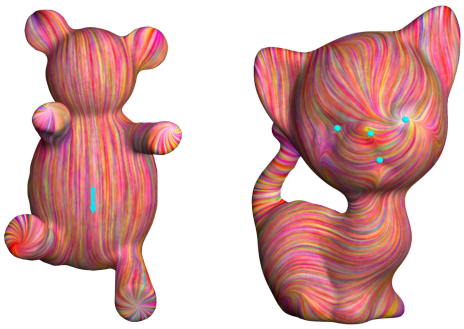


Figure 7: *Prescribing directional constraints (left) or singularities (right).*

smooth functions). Since each eigenfunction of the Laplace-Beltrami operator is smooth, Equation (8) is well defined. One advantage of this basis is that the basis functions are ordered and can be attributed a notion of *scale*, given by the corresponding eigenvalue. This has been exploited in a number of scenarios including the work on functional maps [OBCS*12] where a mapping between two shapes is compactly encoded using a sub-matrix of a possibly infinite functional map matrix. This choice of basis yields a compact representation of the FVF operator as an $N_f \times N_f$ matrix, where N_f is the number of basis functions we use.

3.2. Parameterization with Basis Operators

As mentioned in Section 2.2, the space of linear functional operators is strictly larger than the space of vector fields. Therefore, in order to work with this representation in practice, it is useful to have a *parametrization* of the space of FVFs, which is easy to manipulate.

One possible such parameterization, is to consider a basis for the space of tangent vector fields ψ_i, and to represent an operator D_V as a linear combination of the functional vector field operators D_{ψ_i}. In our work, we consider the eigenfunctions of the 1-form Laplace-de Rham operator to generate a basis for the 1-forms on a surface, and use these as a basis for the tangent vectors, by duality [Mor01].

Given such basis operators D_{ψ_i}, the FVF operator D_V that represents a vector field $V = \sum_i a_i \psi_i$ is given by: $D_V = \sum_i a_i D_{\psi_i}$. Note, that this basis is also ordered, so that smoother vector fields can be represented using fewer basis operators. In practice, we truncate the basis, and limit the number of basis operators to a fixed value N_D.

With this parameterization, it is straightforward to use the properties we mentioned in Section 2 to design a vector field that has various desirable characteristics, simply by solving a linear system for the coefficients a_i. Figure 7 shows a vector field designed by posing a small number of directional constraints (one direction for the teddy (left) and 4 zero valued vectors for the kitten (right)), and solving for the coefficients as explained in Section 5.

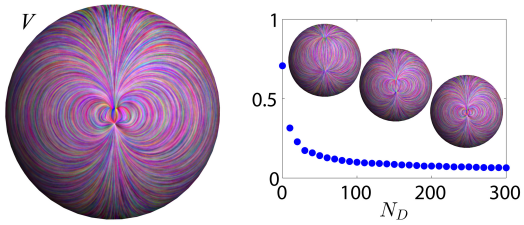


Figure 8: *Given a vector field (left), we reconstruct it with growing accuracy by increasing the number of basis operators N_D (right). Note that the index 2 singularity is accurately reconstructed given enough basis operators.*

Figure 8 demonstrates the effect of using a varying number of basis operators. Given a direction field (left), we project it on a growing number of basis operators and show the reconstruction error as a function of N_D (right). We additionally show the reconstructed vector field, for a few choices of N_D. Note, that although the direction field is smooth, due to the jump from unit length norm to zero norm at the singular point, it is difficult to reconstruct this vector field exactly. However, using a growing number of basis operators we can approximate better this discontinuity in scale.

4. Discretization

So far we have described the properties of tangent vector fields as functional operators in the continuous case. In this section we will focus on the discretization of these concepts to surfaces which are represented as triangle meshes. We propose a finite-element based discretization, and discuss its consistency and experimental convergence properties.

4.1. Representation

We will first address the following problem: given a triangle mesh $M = (X, F, N)$, where X are the vertices, F the faces and N the normals to the faces, and a piecewise constant tangent vector field $V = \{v_r \in \mathbb{R}^3 | r \in F, v_r \perp N_r\}$, how do we represent the functional vector field D_V?

The answer is in fact straightforward, when we consider the representation of D_V in the functional basis given by the standard hat functions. On a triangle mesh we can represent functions in a piecewise linear basis, namely $f(p) = \sum_{j=1}^{|X|} b_j \gamma_j(p)$, where γ_j are the standard hat functions (valued 1 at vertex i and 0 at all other vertices), and $b_j \in \mathbb{R}$ are the coefficients. Now, given the function $f(p) = \sum_j b_j \gamma_j(p)$, and a piecewise constant vector field V, we wish to compute $g = D_V(f)$. We set $g(p) = \sum_j a_j \gamma_j(p)$, and solve (2) in the weak sense, as is standard in Finite Element Analysis (see [AFW06] for a complete discussion of this approach):

$$\int_M \gamma_i g d\mu = \int_M \gamma_i D_V(f) d\mu, \quad \forall i.$$

Plugging in the expressions for f, g and D_V we get $\forall i$:

$$\sum_j a_j \int_M \gamma_i \gamma_j d\mu = \sum_j b_j \int_M \gamma_i \langle \nabla \gamma_j, V \rangle d\mu. \qquad (9)$$

The integrands in (9) vanish everywhere, except on the set of triangles $R_{ij} \subset F$, for which both γ_i and γ_j are non-zero. For $i = j$, these are the triangles neighboring the vertex i. For $i \neq j$, we have that (i, j) must be an edge, and R_{ij} contains only the two triangles which share that edge.

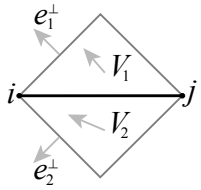
This leads to $\sum_j a_j B_{ij} = \sum_j b_j S_{ij}$, where:

$$B_{ij} = \sum_{t_r \in R_{ij}} \int_{t_r} \gamma_i \gamma_j d\mu, \quad S_{ij} = \sum_{t_r \in R_{ij}} \int_{t_r} \gamma_i \langle \nabla \gamma_j, V \rangle d\mu.$$

Computing the elements B_{ij} yields the standard mass matrix used in the solution of Laplacian systems, whereas S_{ij} is given by (see the inset figure for the notations):

$$S_{ij} = \frac{1}{6}\left(\left\langle V_1, e_1^{\perp}\right\rangle + \left\langle V_2, e_2^{\perp}\right\rangle\right)$$

$$S_{ii} = -\sum_{j \in N(i)} S_{ij}.$$



Here, r_1 and r_2 are the two faces that share the edge (i, j), V_1 is the value of V on the face r_1, $e_1^{\perp}$ is the rotation by $\pi/2$ of the edge opposite to the vertex j in the face r_1 (similarly for V_2 and $e_2^{\perp}$), and $N(i)$ are the neighboring vertices of vertex i. The derivation is given in the supplemental material.

We further replace B with a diagonal lumped mass matrix W of the Voronoi areas w_i of the vertices [Bot10], and get:

$$\mathbf{a} = \hat{D}_V \mathbf{b}, \quad \hat{D}_V = W^{-1} S. \tag{10}$$

Note, that the size of $\hat{D}_V$ is $|X| \times |X|$, but it is sparse, as only the diagonal and entries of adjacent vertices are non-zero.

It is sometimes useful to decompose $\hat{D}_V$ as a product of two operators: $\hat{D}_V = P_{|X|\times|F|}(\hat{D}_V^F)_{|F|\times|X|}$, where P is independent of V and depends only on the mesh. We take:

$$(P)_{ir} = \frac{1}{3w_i}\mathcal{A}_r, \quad (\hat{D}_V^F)_{ri} = \langle \nabla \gamma_i, V \rangle_r, \tag{11}$$

where $\mathcal{A}_r$ is the area of the triangle t_r. In fact, the operator $\hat{D}_V^F$ is simply the smooth operator D_V per triangle, where V is fixed. Therefore, it preserves most of the properties of its smooth counterpart. However, to get an operator which commutes with other operators, we need to apply P, averaging values from faces to vertices. This introduces a discretization error into our formulation, due to the discontinuity of the vector field near the vertices.

Alternatively, we can use the first N_f eigenvectors $\hat{\phi}_i$ of the discrete Laplace-Beltrami operator as the basis for the function space, and then D_V will be represented using an $N_f \times N_f$ matrix, which we will denote by $\hat{D}_V^{LB}$. We compute $\hat{D}_V^{LB}$ by using a change of basis:

$$\hat{D}_V^{LB} = B^{+} \hat{D}_V B, \tag{12}$$

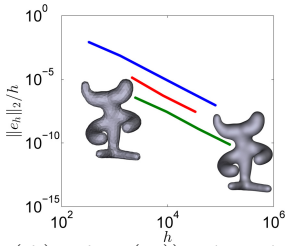
where B is a matrix whose columns are $\hat{\phi}_i$ and B^{+} is its pseudo-inverse. This representation introduces some additional error, due to the truncation of the basis, and there exists a trade-off between the complexity of the representation (in terms of N_f) and the amount of detail the functions we work with can represent.

4.2. Properties

It is interesting to investigate which properties of D_V are preserved from the smooth case, and which are not but converge under refinement of the mesh.

Constant functions. We have that $D_V(c) = 0$, for any constant function c. It is easy to see this property is preserved in the discrete case, since the rows of $\hat{D}_V$ sum to zero, hence the constant functions are in its kernel.

Product rule. The continuous D_V fulfills two defining properties: linearity (Equation (3)) and the Leibnitz product rule (Equation (4)). Since $\hat{D}_V$ is a matrix, linearity is clearly satisfied. However, as we work in a limited subspace of functions, the product rule is no longer valid: given two PL functions f, g, their pointwise product fg is no longer PL, and therefore we cannot apply $\hat{D}_V$ to it. However, we can show empirically that when applying increasingly finer discretizations of D_V to increasingly finer discretizations of continuous functions f, g, the product rule error decreases.



Let f_h, g_h, be the two random smooth piecewise linear functions defined on a mesh with h vertices, and take V to be a smooth tangential vector field. Now, for every h, compute the error $e_h = D_V(f_h g_h) - (g_h D_V(f_h) + f_h D_V(g_h))$, where the multiplication is done vertex-wise. The inset figure shows the graph of $\|e_h\|_2/h$ as a function of h, in *loglog* scale, for a few choices of models. Note that the graph is linear, implying exponential convergence under refinement.

Uniqueness. The correspondence between a vector field V and its FVF operator D_V is one-to-one and onto in the continuous case, implying that given an operator D_V we can uniquely reconstruct the vector field V. This property, unfortunately, may not hold in the discrete case. We do, however have the following weaker result:

Lemma 4.1 Let $M = (X, F, N)$ and let V_1, V_2 be two piecewise constant vector fields on M. Then: $\hat{D}_{V_1}^F = \hat{D}_{V_2}^F$ if and only if $V_1 = V_2$.

In practice, given an operator $\hat{D}_V$ we reconstruct the corresponding vector field V by projecting on the operator basis, as described in Section 3.2.

Metric invariance. The continuous functional vector field operator D_V commutes with the pushforward under a map.

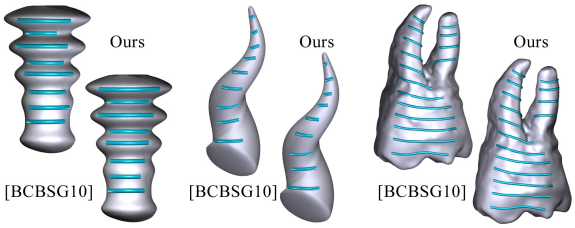


Figure 9: *Geodesic distances between pairs of starting points are measured before and after the flow. Comparing the normalized average error for the models shown yields (left to right): 0.2,0.96, 2.47 for our method, and 0.23,1.15, 4.5 for [BCBSG10] (units are average edge length).*

Specifically, given a bijective diffeomorphism $T : M \to N$, a vector field V_1 on M and a function $f : M \to \mathbb{R}$, we have that $D_{V_1}(f)(p) = D_{V_2}(f \circ T^{-1})(T(p))$, where $V_2 = dT(V_1(p))$, and dT is the differential of T. As a consequence, D_V does not depend on the embedding of the shape M.

As we do not have the uniqueness property, the discrete metric invariance property is also limited to the $\hat{D}_V^F$ operator:

Lemma 4.2 Let $M_1 = (X_1, F, N_1)$ and $M_2 = (X_2, F, N_2)$ be two triangle meshes with the same connectivity but different metric (i.e. different embedding). Additionally, let V_1 be a piecewise constant vector field on M_1, then $\hat{D}_{V_1}^F = \hat{D}_{V_2}^F$.

Here $(V_2)_r = A(V_1)_r$, where A is the linear transformation that takes the triangle r in M_1 to the corresponding triangle in M_2. Note that $\hat{D}_{V_i}$ is computed using the embedding X_i.

Integration by parts. For a closed surface, we have that $\int_M f(\nabla \cdot V) = \int_M \langle \nabla f, V \rangle = \int_M D_V(f)$, for all $f : M \to \mathbb{R}$. This holds exactly in the discrete case, when using the standard vertex-based discrete divergence, defined as in [PP03]:

Lemma 4.3 Let $M = (X, F, N)$, V a piecewise constant vector field on M, $f = \sum_i f_i \gamma_i$ a PL function on M, and w_i the Voronoi area weights, then:

$$\sum_{i=1}^{|X|} w_i (\hat{D}_V f)_i = \sum_{i=1}^{|X|} w_i f_i (\mathrm{div}(V))_i.$$

5. Applications

In this section, we describe how our representation can be used to compute vector fields which have various desirable properties. While some of the suggested applications have been attempted before (e.g. designing vector fields using direction and singularity constraints [FSDH07, CDS10], computing Killing vector fields [BCBSG10] and symmetric vector fields [PLPZ12], among others), our framework is unique in that it allows to combine any such constraints into a single optimization problem. In addition, we provide a proof-of-concept for more advanced tools, such as jointly designing vector fields on two or more surfaces.

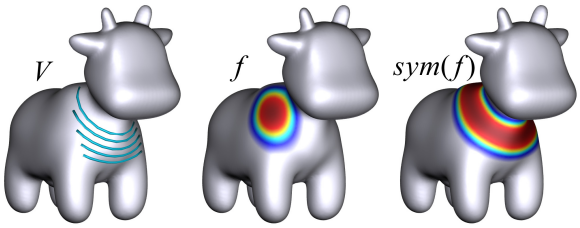


Figure 10: *An AKVF V (left), an indicator function f (center), and its symmetrization computed by projecting f on the kernel of D_V (right).*

5.1. Implementation Details

Given a mesh M, scalars N_f, N_D and a set of desired properties for a vector field, we propose the following algorithm:

1. Compute the first N_f eigenfunctions of the LB operator $\hat{\phi}_i$, using the area normalized cotangent scheme [Bot10].
2. Compute the first N_D 1-form eigenfunctions of the Laplace-de Rham operator, and convert those to piecewise constant vector fields $\hat{\psi}_i$. We used the definitions from [FSDH07] for both operations.
3. Convert $\hat{\psi}_i$ to $\hat{D}_{\hat{\psi}_i}^{LB}$ using Equation (12).
4. Optimize simultaneously for the vector field $V = \sum_i a_i \hat{\psi}_i$ and its functional representation $D_V = \sum_i a_i \hat{D}_{\hat{\psi}_i}^{LB}$, by solving a linear system for a_i. The joint formulation allows us to stack constraints which are best represented using the operator (e.g. commutativity constraints) together with constraints which require the vector field (e.g. prescribed directions at specified locations). This yields a linear system $\mathbf{Wa} = \mathbf{c}$, which we solve in the least squares sense.
5. Output the computed vector field $V = \sum_i a_i \hat{\psi}_i$.

Throughout our experiments we used meshes in the range of 5k-200k vertices, with N_f and N_D between 50 and 300, depending on the experiment. The computational time was dominated by the eigen-decompositions and took a few minutes on a standard laptop.

Figures 3, 4, 5 and 7 from the previous sections were generated using this framework. In addition, we describe a few examples of potential applications of our framework, related to the properties discussed in Section 2.

5.2. Approximate Killing Vector Fields

Lemma 2.2 provides a linear constraint on the FVF operator, which guarantees that a given vector field is a KVF. We can use this result, and optimize for the best KVF on a given surface, by optimizing for a set of coefficients $\mathbf{a}$ such that the resulting operator D_V will commute with the Laplace-Beltrami operator, i.e. $||D_V \circ L - L \circ D_V|| = 0$. Here we get a homogeneous system $\mathbf{Wa} = 0$, hence the AKVF is the singular vector corresponding to the lowest singular value.

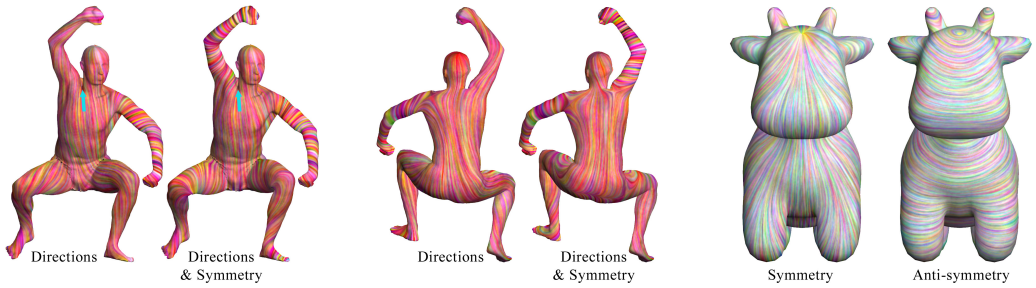


Figure 11: *On the human model (left and center) we show design results with and without symmetry constraints - note the difference on the right hand. On the spot model (right) we show symmetric and anti-symmetric vector fields.*

Figure 9 shows a comparison of the resulting vector fields with the results of the state-of-the-art algorithm [BCBSG10]. The comparison is done using the same meshes, where on each mesh we pick a few vertices and show the flow lines for a fixed time t starting from these vertices. Note, that we achieve similar results, but in our framework we can easily combine the KVF constraint with other constraints such as commutativity with a symmetry operator.

Interestingly, the spectral decomposition of the functional vector field operator is meaningful and potentially useful in applications. Specifically, functions are in the kernel of D_V if and only if they are fixed points of the flow Φ_V^t for all t (since $D_V f = 0$ if and only if $\exp(tD_V)f = f, \forall t$). Therefore, the kernel of an AKVF operator spans the linear subspace of symmetric functions under the corresponding symmetry. This implies, that given an arbitrary function f, we can symmetrize it by projecting it onto the kernel of such an operator. Figure 10 shows an example of an AKVF V, an indicator function f and its symmetrization $sym(f)$.

5.3. Composition with Mappings

Given a self-map S, we design a symmetric vector field by posing a constraint of the form $||D_V \circ S - S \circ D_V|| = 0$. Figure 11 (left and center) shows an example of a vector field designed with directional constraints and one designed with both directional and symmetry commutativity constraints. Note the difference on the hand of the model, as the symmetric constraints enforce similar behavior on both hands. Additionally, we can define an anti-symmetric vector field, by requiring $V(S(p)) = -V(p)$, where S is the symmetry map. To enforce this requirement, we use the constraint $||D_V \circ S + S \circ D_V|| = 0$. Figure 11 (right) shows an example of symmetric and anti-symmetric vector fields.

Given a collection of shapes, a desirable goal when designing vector fields is to have different constraints on each shape, yet generate compatible vector fields across the collection. In Figure 12 (right) we achieve this goal using the map composition property. We are given two shapes M_1 and M_2 and a functional map T_F between the corresponding function spaces. In addition, on each shape we are given a set of directional constraints c_1, c_2. We wish to generate vector fields V_i on the shapes M_i, such that V_i commute with T_F, and fulfill the constraints. A natural approach would be to transfer the constraints and solve separately for each mesh. However, as shown in Figure 12 (left), there is a large difference between the resulting fields - e.g in the locations of the singularities. Figure 12 (right), shows the result when solving jointly for both shapes. Note that the singularities on the back of the shape are consistent between the models. For evaluation, we transport V_1 to M_2 and measure the angle difference between the resulting vector field and V_2. Figure 12 (center) shows the resulting histogram, emphasizing that our joint design method preserves the directions better.

6. Discussion

Tangent vector fields on surfaces are used in a myriad of applications in computer graphics and geometry processing. We propose to represent them as functional operators, thus enabling applications which were not easily attainable using standard representations. We have provided a discretization of the operator, and demonstrated it is consistent and experimentally convergent under refinement. Finally, we described some high level vector field design applications, such as Killing, symmetric and joint vector field design.

We believe the proposed representation opens the door for many additional applications. Specifically, the covariant derivative of one vector field with respect to another could potentially be useful for computing the Gaussian curvature, and for posing smoothness constraints for vector field design. Further applications include finding pairs of vector fields with zero Lie derivative for surface parameterization.

In general, we feel that we only uncovered the tip of the iceberg of possible applications and extensions of this framework. In an even broader context, considering both the operator representation of maps between surfaces, and the operator representation of vector fields, seems to imply that a lot is to gain by abstracting common notions in geometry processing, and viewing them more generally as operators. It remains to be seen whether this approach is applicable to additional concepts as well.

Acknowledgements We thank Keenan Crane, AIM@Shape and SCAPE for the models. The authors acknowledge ISF grant 699/12, ISF equipment grant, Marie Curie CIG 303511, ANR project GIGA (ANR-09-BLAN-0331-01), CNRS chaire d'excellence, and Marie Curie CIG 334283.

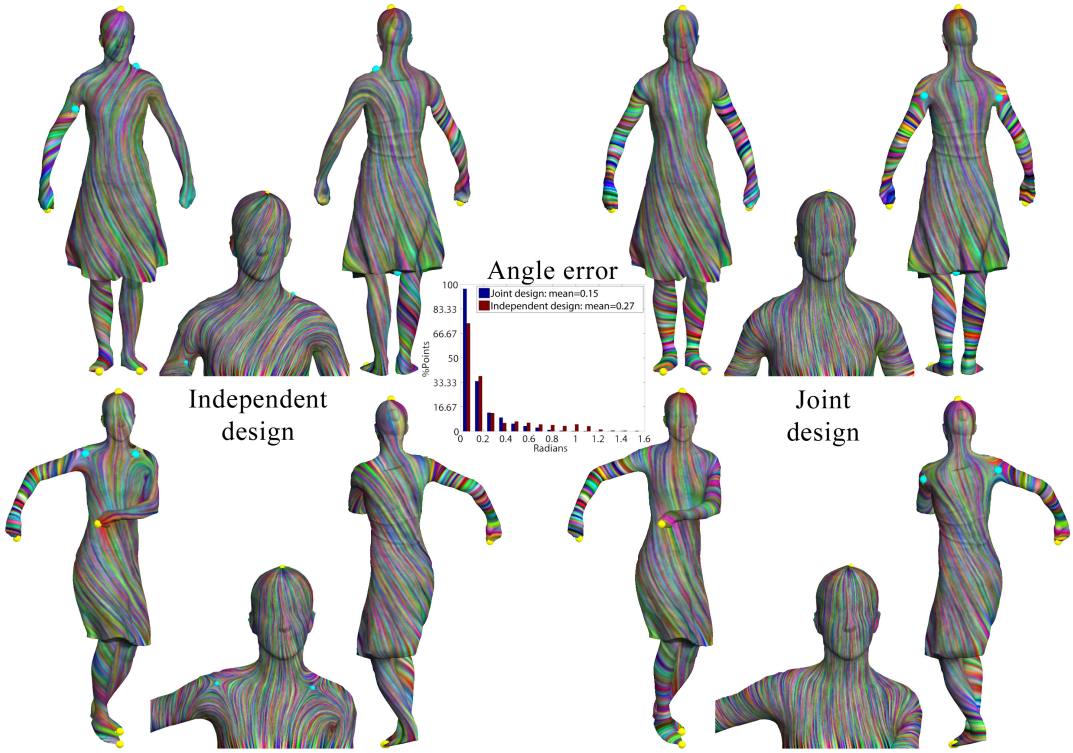


Figure 12: *(left) Independent design on two shapes which are in correspondence does not yield a consistent vector field, even if compatible constraints are used. (right) Solving jointly using our framework yields consistent vector fields (note the corresponding locations of the singularities on the back of the shape). See the text for details.*

References

[AFW06] ARNOLD D. N., FALK R. S., WINTHER R.: Finite element exterior calculus, homological techniques, and applications. *Acta numerica 15*, 1 (2006), 1–155. 6

[BCBSG10] BEN-CHEN M., BUTSCHER A., SOLOMON J., GUIBAS L.: On discrete killing vector fields and patterns on surfaces. In *CGF* (2010), vol. 29, pp. 1701–1711. 8, 9

[Bot10] BOTSCH M.: *Polygon mesh processing*. A K Peters, Natick, Mass, 2010. 7, 8

[CDS10] CRANE K., DESBRUN M., SCHRÖDER P.: Trivial connections on discrete surfaces. In *CGF* (2010), vol. 29, pp. 1525–1533. 2, 8

[DFN92] DUBROVIN B., FOMENKO A., NOVIKOV S.: Modern geometry-methods and applications, part i: The geometry of surfaces, transformation groups, and fields, vol. 93. *Graduate texts in mathematics* (1992). 5

[FSDH07] FISHER M., SCHRÖDER P., DESBRUN M., HOPPE H.: Design of tangent vector fields. *ACM Transactions on Graphics (TOG) 26*, 3 (2007), 56. 2, 8

[Hir03] HIRANI A. N.: *Discrete exterior calculus*. PhD thesis, California Institute of Technology, 2003. 2

[MMP*11] MULLEN P., MCKENZIE A., PAVLOV D., DURANT L., TONG Y., KANSO E., MARSDEN J., DESBRUN M.: Discrete lie advection of differential forms. *Foundations of Computational Mathematics 11*, 2 (2011), 131–149. 2

[Mor01] MORITA S.: *Geometry of differential forms*. American Mathematical Society, Providence, R.I, 2001. 2, 3, 5, 6

[OBCS*12] OVSJANIKOV M., BEN-CHEN M., SOLOMON J., BUTSCHER A., GUIBAS L.: Functional maps: a flexible representation of maps between shapes. *ACM Trans. Graph. 31*, 4 (July 2012), 30:1–30:11. 2, 4, 6

[Pet97] PETERSEN P.: *Riemannian geometry*. Graduate texts in mathematics. Springer, 1997. 2, 4, 5

[PLPZ12] PANOZZO D., LIPMAN Y., PUPPO E., ZORIN D.: Fields on symmetric surfaces. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 111. 8

[PMT*11] PAVLOV D., MULLEN P., TONG Y., KANSO E., MARSDEN J., DESBRUN M.: Structure-preserving discretization of incompressible fluids. *Physica D: Nonlinear Phenomena 240*, 6 (2011), 443 – 458. 2

[PP03] POLTHIER K., PREUSS E.: Identifying vector field singularities using a discrete hodge decomposition. *Visualization and Mathematics 3* (2003), 113–134. 1, 2, 8

[PZ07] PALACIOS J., ZHANG E.: Rotational symmetry field design on surfaces. In *ACM Transactions on Graphics (TOG)* (2007), vol. 26, ACM, p. 55. 2

[PZ11] PALACIOS J., ZHANG E.: Interactive visualization of rotational symmetry fields on surfaces. *Visualization and Computer Graphics, IEEE Transactions on 17*, 7 (2011), 947–955. 4

[RVAL09] RAY N., VALLET B., ALONSO L., LEVY B.: Geometry-aware direction field processing. *ACM Transactions on Graphics (TOG) 29*, 1 (2009), 1. 2

[Spi99] SPIVAK M.: *A comprehensive introduction to differential geometry. Vol. I*, third ed. Publish or Perish Inc., 1999. 3, 4

[TLHD03] TONG Y., LOMBEYDA S., HIRANI A. N., DESBRUN M.: Discrete multiscale vector field decomposition. In *ACM Transactions on Graphics (TOG)* (2003), vol. 22, pp. 445–452. 2