DOI:10.1111/cgf.14070
Eurographics Symposium on Geometry Processing 2020
Q. Huang and A. Jacobson
(Guest Editors)

Volume 39 (2020), Number 5

A Simple Discretization of the Vector Dirichlet Energy

Oded Stein[1], Max Wardetzky[2], Alec Jacobson[3] and Eitan Grinspun[3,1]

[1]Columbia University, USA
[2]University of Göttingen, Germany
[3]University of Toronto, Canada

Abstract
We present a simple and concise discretization of the covariant derivative vector Dirichlet energy for triangle meshes in 3D using Crouzeix-Raviart finite elements. The discretization is based on linear discontinuous Galerkin elements, and is simple to implement, without compromising on quality: there are two degrees of freedom for each mesh edge, and the sparse Dirichlet energy matrix can be constructed in a single pass over all triangles using a short formula that only depends on the edge lengths, reminiscent of the scalar cotangent Laplacian. Our vector Dirichlet energy discretization can be used in a variety of applications, such as the calculation of Killing fields, parallel transport of vectors, and smooth vector field design. Experiments suggest convergence and suitability for applications similar to other discretizations of the vector Dirichlet energy.

1. Introduction

The covariant derivative ∇ generalizes the gradient of scalar functions to vector fields defined on surfaces. As the gradient does for scalar functions, the covariant derivative measures the infinitesimal change of a vector field in every direction. As with the gradient's scalar Dirichlet energy, $E_{\text{scalar}}(u) := \frac{1}{2}\int_\Omega \|\nabla u\|^2 \,\mathrm{d}x$ for a smooth scalar function u and a smooth surface Ω, the covariant derivative has a corresponding *vector Dirichlet energy*,

$$E(\mathbf{u}) := \frac{1}{2}\int_\Omega \|\nabla \mathbf{u}\|_{\mathrm{F}}^2 \,\mathrm{d}x\,, \tag{1}$$

where $\mathbf{u}$ is a smooth vector-valued function on Ω, and $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm. Much like the scalar Dirichlet energy E_{scalar} does for scalar functions, the vector Dirichlet energy E measures the smoothness of a vector field.

Just as E_{scalar} is useful for scalar data processing, E has many uses in vector field processing. While E_{scalar} has been employed in many geometry processing methods and applications, E has, in comparison, seen less usage in practice. A key reason for the wide adoption of E_{scalar} is the existence of a simple, reliable, and readily available finite element discretization: the cotangent Laplacian. There are a variety of existing discretizations of E, but they can not be implemented using quite as few lines of code as the cotangent Laplacian. The matrix for the cotangent Laplacian can be constructed using only the expression $L_{\text{cotan}ij} = \frac{1}{2}\left(\cot\alpha_{ij} + \cot\beta_{ij}\right)$ for each vertex pair i, j (where α_{ij}, β_{ij} are the two angles opposite the edge ij), as well as summation [MDSB03, (5)]. We advocate for a discretization of E that aspires to the simplicity of the cotangent Laplacian.

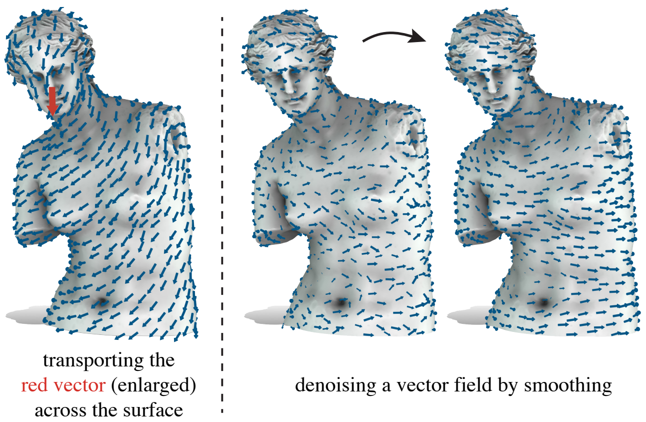


Figure 1: *Parallel transporting a vector across a surface with our Crouzeix-Raviart discretization of vector heat flow* (left)*, and denoising a vector field with our discretization of the vector diffusion equation* (right).

Our discretization is based on Crouzeix-Raviart finite elements. Its most notable features are its simplicity and ease of implementation, *without sacrificing any quality*:

- the real, sparse Dirichlet energy matrix can be built in a single pass over all faces using a simple formula depending only on each (individual) triangle's edge lengths;
- the degrees of freedom directly correspond to the edges of the mesh, making the results easy to understand and visualize;
- no preprocessing or intermediate data structures beyond simple matrices are required;
- applications, performance and convergence are similar to other discretizations of E.

The sparse $2m \times 2m$-matrix L of the discrete energy E for a mesh with m edges

```
1: for all faces $f$, consecutive edges $e_i, e_j \in f$ do
2:     $l_i = \mathrm{len}(e_i),\ l_j = \mathrm{len}(e_j),\ l_k = \mathrm{len}(e_k)$          // $e_k$ 3rd edge
3:     $s = \mathrm{orientation}(e_i) \cdot \mathrm{orientation}(e_j)$          // can be 1 or $-1$
4:     $a = s\,(l_i^2 + l_j^2 - l_k^2)^2 / (4 l_i l_j\, \mathrm{area}(f))$
5:     $b = s\,(l_i^2 + l_j^2 - l_k^2)/(l_i l_j)$
6:     $L(e_i,e_i)$ += $l_i^2 / \mathrm{area}(f)$;  $L(e_i+m, e_i+m)$ += $l_i^2 / \mathrm{area}(f)$
           // $L(a,b) \leftarrow x$ expands to $L(a,b) = x;\ L(b,a) = x$
7:     $L(e_i,e_j) \leftarrow a$;  $L(e_i+m, e_j+m) \leftarrow a$
8:     $L(e_i,e_j+m) \leftarrow b$;  $L(e_j,e_i+m) \leftarrow -b$
```

We demonstrate our discretization for three established applications of the vector Dirichlet energy—smoothing vector fields, creating Killing vector fields, and efficient parallel transport of vectors (see Figure 1). We study the convergence of our method and two previous methods, and we find that our method converges at the same rate as previous methods.

The way we arrive at our discretization parallels the discretization of E_{scalar} by the cotangent Laplacian for applications in scalar geometry processing. Desbrun et al. [DMSB99] adopted the well-established cotangent Laplacian [Mac49], demonstrated its utility, and spurred its adoption in geometry processing. Similarly, we employ a discretization introduced by Stein et al. [SJWG20] as part of a routine to process scalar functions, and demonstrate its utility for vector field processing. Our discretization generalizes popular and extensively studied methods from numerical analysis of finite elements for flat domains in $\mathbb{R}^2$, which simplifies mathematical study of the method.

2. Theoretical background

In this section we provide a brief overview of the covariant derivative and the vector Dirichlet energy. Readers who are already familiar with these topics, or who want to get straight to the applications, can safely skip this section. The discretization and matrix implementation follow in Section 3.

2.1. The covariant derivative

For a vector field $\mathbf{u}$, the covariant derivative $\nabla\mathbf{u}$ generalizes the gradient of a scalar function, ∇f [Lee97, Section 4].

We will be working on a smooth, orientable surface Ω. The covariant derivative is an operator $\nabla : \mathcal{T}(\Omega) \times \mathcal{T}(\Omega) \to \mathcal{T}(\Omega)$, where $\mathcal{T}(\Omega)$ is the space of tangent vector fields on the smooth surface Ω. It is usually written as $\nabla_{\mathbf{v}}\mathbf{u}$ for two vector fields $\mathbf{v}, \mathbf{u}$, and when the subscript $\mathbf{v}$ is omitted, $\nabla\mathbf{u}$ is interpreted as the operator that takes $\mathbf{v}$ as an argument and returns $\nabla_{\mathbf{v}}\mathbf{u}$. In the flat Euclidean space $\mathbb{R}^3$, with a vector field $\mathbf{u}$ written as a column vector $\mathbf{u} = \begin{pmatrix} \mathbf{u}_x & \mathbf{u}_y & \mathbf{u}_z \end{pmatrix}^\intercal$, the covariant derivative is simply componentwise differentiation,

$$\nabla\mathbf{u} = \begin{pmatrix} \partial_x\mathbf{u}_x & \partial_y\mathbf{u}_x & \partial_z\mathbf{u}_x \\ \partial_x\mathbf{u}_y & \partial_y\mathbf{u}_y & \partial_z\mathbf{u}_y \\ \partial_x\mathbf{u}_z & \partial_y\mathbf{u}_z & \partial_z\mathbf{u}_z \end{pmatrix} .$$

The covariant derivative on a smooth surface is this vector gradient of $\mathbb{R}^3$ restricted to the tangent space of the surface, where it becomes dependent on the surface's curvature (while this definition only works for surfaces embedded in$\mathbb{R}^3$, it generalizes to other kinds of surfaces [Lee97, Section 5]).

Operators like $\nabla\mathbf{u}$, which consume a vector field and return a vector field, have an associated inner product $:$ that generalizes the scalar product $\cdot$. The operation $:$ for X, Y represented as matrices can be written as $X : Y := \mathrm{tr}\,(X^\intercal Y)$, where $X^\intercal$ is the transpose of an operator X that takes the surface's metric into account [Lee97]. $X^\intercal Y$ is sometimes also referred to as composing the adjoint of X with Y.

The covariant derivative ∇ has an *adjoint* operator ∇^*, similar to how the gradient of a scalar function is adjoint to the divergence of a vector field. The adjoint covariant derivative is defined by integration by parts over a surface Ω,

$$\int_\Omega \nabla\mathbf{u} : X \,\mathrm{d}x = \int_\Omega \mathbf{u} \cdot \nabla^* X \,\mathrm{d}x + \int_{\partial\Omega} \mathbf{u} \cdot X(\mathbf{n}) \,\mathrm{d}\Gamma \qquad (2)$$
$$\mathbf{n} \text{ unit boundary normal at } \partial\Omega ,$$

where X is an operator that takes a vector field and returns a vector field, and $\mathbf{u}$ is a vector field. This adjoint operator is used to define a Laplace operator for vector fields, the *Bochner Laplacian* (also referred to as the connection Laplacian) [Pet06, pp. 209],

$$\Delta_B \mathbf{u} = \nabla^* \nabla \mathbf{u} .$$

In $\mathbb{R}^n$, the Bochner Laplacian is (up to sign) the usual vector field Laplacian defined by applying the scalar Laplacian in each coordinate. It appears, for example, in the Navier-Stokes equations for fluid mechanics [Fan19].

2.2. The vector Dirichlet energy

The Bochner Laplacian Δ_B defines a vector Laplace equation,

$$\Delta_B \mathbf{u} = 0 , \qquad (3)$$

which shares many properties with the scalar Laplace equation $\Delta u = 0$, such as smoothness of the associated flow [BGV96, Chapter 2]. In general, the right-hand side can be any function, but we will assume that it is zero for simplicity.

The Bochner Laplacian has an associated energy, constructed using integration by parts. Let $\mathbf{u}, \mathbf{v}$ be smooth vector fields on the surface Ω. Then we can write (3), by multiplying with a test function $\mathbf{v}$ and integrating, as

$$\int_\Omega \Delta_B \mathbf{u} \cdot \mathbf{v} \,\mathrm{d}x = \int_\Omega \nabla\mathbf{u} : \nabla\mathbf{v} \,\mathrm{d}x - \int_{\partial\Omega} \nabla_{\mathbf{n}}\mathbf{u} \cdot \mathbf{v} \,\mathrm{d}\Gamma \qquad (4)$$
$$\mathbf{n} \text{ unit boundary normal at } \partial\Omega ,$$

where we used (2). By calculus of variation [Eva92, Section 8], the solution $\mathbf{u}$ of (4) is the minimizer of the *vector Dirichlet energy*,

$$\boxed{E(\mathbf{u}) := \frac{1}{2} \int_\Omega \|\nabla\mathbf{u}\|_{\mathrm{F}}^2 \,\mathrm{d}x ,} \qquad (5)$$

where $\|X\|_{\mathrm{F}}^2 := X : X$ is the Frobenius norm. The boundary term in (4) implies that minimizers will fulfill the Neumann boundary condition, $\nabla_{\mathbf{n}}\mathbf{u} = 0$ at $\partial\Omega$, if no explicit boundary condition is applied

(Neumann conditions are the natural boundary conditions [GF63, I.6]).

The vector Dirichlet energy (5) quantifies the smoothness of a vector field, just like the scalar Dirichlet energy $E_{\text{scalar}}(f) = \frac{1}{2}\int_\Omega \|\nabla f\|^2 \mathrm{d}x$ does for scalar functions. This makes the vector Dirichlet energy and its associated Laplace equation useful for applications requiring a notion of vector field smoothness (see Section 4).

3. Discretization

3.1. Scalar Crouzeix-Raviart finite elements

Our discretization of the vector Dirichlet energy is based on Crouzeix-Raviart finite elements, a simple discontinuous finite element. In this section, we introduce the scalar Crouzeix-Raviart basis functions. More detail can be found in the book of Braess [Bra07], Chapter III.

Scalar Crouzeix-Raviart elements are used for applications in computer graphics such as simulation [BWH*06, WBH*07, EB08] and geometry processing [HP04, HTWB11].

Consider a nondegenerate, manifold, and oriented triangle mesh with vertices $v \in V$, edges $e \in E$ and faces $f \in F$. There is one Crouzeix-Raviart basis function per edge. The basis function b_e for edge e is

- 0 outside the two triangles neighboring e;
- constant 1 at the edge e, and -1 at the vertices opposite e;
- 0 at the edge midpoints not on e;
- linear within the two triangles neighboring e.

Figure 2 features an illustration of Crouzeix-Raviart basis functions. Linear combinations of the basis functions $\sum_e u_e b_e$ are continuous only at edge midpoints, and in general discontinuous everywhere else. This makes the finite element *nonconforming*. When formulating the Poisson equation, we look for solutions in the space of all continuous, piecewise differentiable functions (to be more precise, we look for solutions in the Sobolev space H^1 of functions that have one L^2-integrable weak derivative). Since the Crouzeix-Raviart functions are not continuous, they are not themselves contained in the solution space. This is in contrast to, for example, Lagrangian piecewise linear per-vertex basis functions (the so-called hat functions). They are continuous and piecewise differentiable, and thus contained in the solution space—they are *conforming*.

The finite element operators are constructed as one would with conforming elements, with one subtlety: differential operators are applied to the basis functions within triangles only, and we ignore the jumps caused by discontinuities. These jumps are nonzero, but they can be ignored without harming convergence for certain classes of problems, such as the Poisson equation [Bra07, III]. The Laplace matrix, which discretizes $E_{\text{scalar}}(f) = \frac{1}{2}\int_\Omega \|\nabla f\|^2 \,\mathrm{d}x$, is given by

$$L_{\text{scalar}_{e_i e_j}} = \int_\Omega \nabla b_{e_i} \cdot \nabla b_{e_j} \,\mathrm{d}x = -\cot\theta_{e_i e_j} \qquad i \neq j\,,$$

for edges e_i, e_j sharing a vertex, where $\theta_{e_i e_j}$ denotes the angle between the edges e_i, e_j, and the diagonal terms are given by the fact that rows must sum to zero [WBH*07, p. 504]. The mass matrix, which discretizes $\frac{1}{2}\int_\Omega |f|^2 \,\mathrm{d}x$, and contains only diagonal terms (without having to use a lumped matrix), is given by

$$M_{\text{scalar}_{e_i e_i}} = \int_\Omega b_{e_i}^2 \,\mathrm{d}x = \frac{1}{3} A_{e_i}\,,$$

where A_{e_i} is the area of the two faces incident on e_i.

As with every nonconforming finite element, one has to be careful for which problem the method is employed. As the functions themselves are not contained in a discrete subspace of the solution space, in general, they might not converge, or their limit might not solve the smooth PDE. One needs to make sure that the finite elements are actually amenable to the particular problem. For the Poisson equation, the Crouzeix-Raviart finite element converges on the order of h^2, where h is the maximum edge length, given certain triangle regularity conditions [Bra07, III.1.5].

3.2. Vector Crouzeix-Raviart finite elements

Using the scalar Crouzeix-Raviart element from Section 3.1, we now construct a *vector* Crouzeix-Raviart element. This element appears in the work of Stein et al. [SJWG20], where it is used to

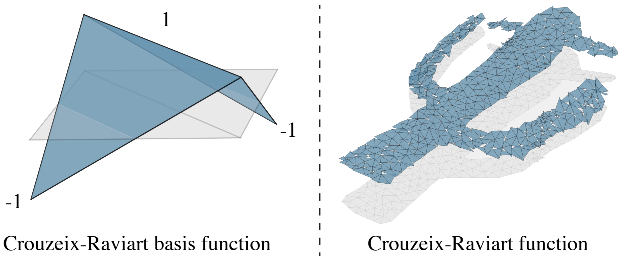


Figure 2: *A Crouzeix-Raviart basis function is* 1 *at its associated edge, and* -1 *on the opposing vertices* (left). *In general, Crouzeix-Raviart functions are discontinuous, except at edge midpoints* (right).

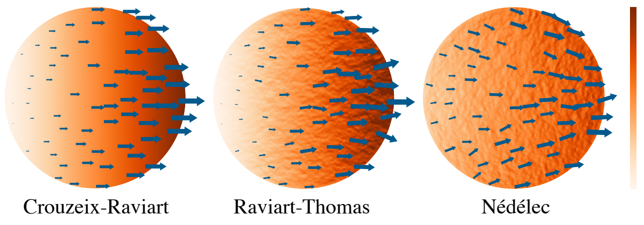


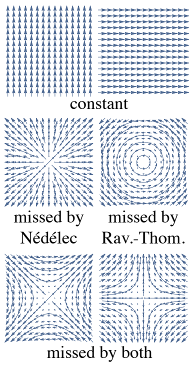
Figure 3: *Trying to recover the linear planar vector field* $(x,0)^\mathsf{T}$ *on the unit disk by fixing its values on the boundary and then minimizing the Crouzeix-Raviart vector Dirichlet energy* (left), *as well as vector Dirichlet energies constructed by integrating the pointwise covariant derivatives of Raviart-Thomas* (center) *and Nédélec* (right) *basis functions. Only the Crouzeix-Raviart version succeeds.*

compute the one-form Dirichlet energy as part of a larger routine to minimize the Hessian energy of a scalar function (but not for any vector field processing). We use their approach to construct a discrete function space for tangent vector fields and discretize the vector Dirichlet energy, and then minimize this energy directly for vector field applications. Our matrices are almost the same as those of Stein et al. [SJWG20], but differ by a scale factor for each basis function, as our degrees of freedom correspond to unit vectors (which is only a difference of convention, and does not have any practical implications beyond multiplying matrix entries by the respective edge lengths).

The Crouzeix-Raviart finite element is used to discretize a variety of problems in the numerical analysis of flat $\mathbb{R}^n$. For an overview, we refer the reader to the survey by Brenner [Bre15].

The linear Raviart-Thomas element (different from our discretization), which is related to the Crouzeix-Raviart element, is another popular vector fields discretization [Gat14]. However, it can only be safely used to compute divergence-like operations. The same holds true of the Nédélec linear triangle element for curl-like operations [Hip99] (also known as Whitney element), which is featured in Discrete Exterior Calculus [Hir03], and can be used to discretize the Hodge Laplacian (but *not* the Bochner Laplacian Δ_B).

Both Raviart-Thomas functions and Nédélec functions consist of one degree of freedom per edge, and every face supports three basis functions: the two constant functions, plus (for Raviart-Thomas) functions with zero curl and constant divergence, or (for Nédélec) functions with zero divergence and constant curl. Since on a flat triangle the space of linear vector fields is six-dimensional, these approaches can not sample all linear vector fields on each triangle. Our approach, on the other hand, will use six degrees of freedom supported in each triangle, two per edge, and feature *exactly* the six linear functions on each triangle. Because they do not span all linear vector fields, both Raviart-Thomas and Nédélec elements are ill-suited for discretizing the vector Dirichlet energy using simple per-face integrals. Neither encompasses the full space of linear vector fields—Raviart-Thomas can not represent functions with curl within the triangle, and Nédélec can not represent functions with divergence. Even combining both spaces would not suffice, as one would still miss saddle-like linear vector fields (see inset, which shows a basis of piecewise linear vector fields). These saddle-like vector fields have non-zero vector Dirichlet energy E and thus matter for our purposes.



This can also be seen by a short degree of freedom argument: since both Nédélec and Raviart-Thomas have three degrees of freedom per triangle, and both contain all constant functions (two per triangle), both combined can not cover more than four linearly independent functions per triangle. There are, however, six linearly independent linear functions per triangle. These are exactly the functions represented by the six degrees of freedom of the vector Crouzeix-Raviart finite element per triangle.

In order to illustrate the need to capture all degrees of freedom of linear vector fields, we compared our approach to using the discontinuous Raviart-Thomas and as Nédélec basis functions. We employed Raviart-Thomas and as Nédélec basis functions to construct a discrete vector Dirichlet energy by computing the integrand from (5) for each basis function, ignoring discontinuities (just as one would for Crouzeix-Raviart discontinuous Galerkin), fix the boundary of a planar domain to the boundary values of a Bochner-harmonic function **u**, and then trying to globally recover **u** by minimizing the vector Dirichlet energy. For both Raviart-Thomas as well as Nédélec, the matrix was not invertible, so we regularized it with 10^{-14}Id. by minimizing the vector Dirichlet energy. In Figure 3, this approach for a linear function **u** using Raviart-Thomas and Nédélec basis functions fails, while the Crouzeix-Raviart approach recovers the function exactly (up to numerical error).

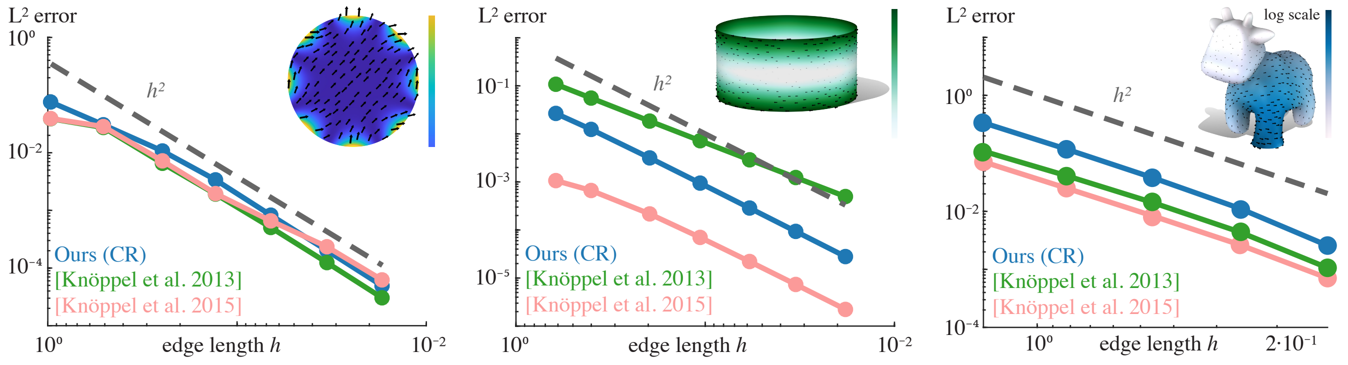
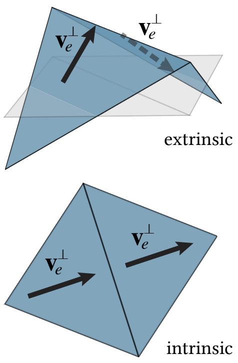


Figure 4: *L^2 convergence plots of our method as well as the methods of Knöppel et al. [KCPS13, KCPS15] (the method of Knöppel et al. [KCPS15] is also used by Sharp et al. [SSC19]) for the Dirichlet boundary value problem. The flat* (left) *and cylindrical* (middle) *examples show convergence to the exact solution, the cow* (right) *shows convergence to the respective highest-resolution numerical solution. The figures in the corners are colored by vector magnitude.*

To extend the Crouzeix-Raviart element to vectors, we multiply the scalar basis functions b_e with appropriate vectors [SJWG20]. At the midpoint of each edge e, we represent all tangent vectors as a linear combination of the following two vectors,

- $\mathbf{v}_e^{\|}$, the unit vector parallel to e that points in the same direction as the oriented edge e;
- $\mathbf{v}_e^{\perp}$, the unit vector normal to e that correspoints to $\mathbf{v}_e^{\|}$ rotated by $\pi/2$ in the tangent space.



At first glance, $\mathbf{v}_e^{\perp}$ seems to be ambiguously defined, since there is no unique normal at the midpoint of e to rotate around. However, *intrinsically*, the tangent space of the mesh (V,E,F), viewed as a polyhedron, is well-defined away from vertices [War06]. A pair of triangles on its own is intrinsically flat. This means that if, at each triangle adjacent to e, we rotate $\mathbf{v}_e^{\|}$ around the respective triangle normal, we get two different extrinsic representations of $\mathbf{v}_e^{\perp}$ in each triangle, each corresponding to the same *intrinsic* tangent vector (see inset). $\mathbf{v}_e^{\|}$ is well-defined at the midpoint of e in both adjacent triangles, as the edge e is contained in both triangles.

The vectors $\mathbf{v}_e^{\|}$ and $\mathbf{v}_e^{\perp}$, defined at the midpoint of e, can be easily extended along each of the two faces adjacent to e by transporting them along the flat triangles. This allows us to define two vector basis-functions per edge:

$$\mathbf{w}_e^{\|} = b_e \mathbf{v}_e^{\|}$$
$$\mathbf{w}_e^{\perp} = b_e \mathbf{v}_e^{\perp} \ .$$

The $\mathbf{w}_e^{\|}, \mathbf{w}_e^{\perp}$ $\forall e \in E$ form the basis of our discrete vector space.

We now use them to compute a discretization of the vector Dirichlet operator (5) by plugging each pair of basis functions into the definition of the energy and integrating over triangles. The resulting sparse symmetric matrix L has two degrees of freedom (DOFs) for each edge $e \in E$, which we will denote by the indices $e^{\|}$ and $e^{\perp}$. For a DOF vector u, $\frac{1}{2}\mathrm{u}^{\mathsf{T}} L \mathrm{u}$ will discretize $E(\mathbf{u})$. The matrix L is constructed using the following formula, which is looped over all faces $f \in F$, and all pairs of consecutive edges $e_i, e_j \in f$,

$$\boxed{\begin{aligned} L_{e_i^{\|},e_i^{\|}},\, L_{e_i^{\perp},e_i^{\perp}} &\mathrel{+}= \frac{|e_i|^2}{|f|} \\ L_{e_i^{\|},e_j^{\|}},\, L_{e_i^{\perp},e_j^{\perp}} &= 2\, s_{ij} \cot\theta_{ij} \cos\theta_{ij} \\ L_{e_i^{\|},e_j^{\perp}},\, -L_{e_i^{\perp},e_j^{\|}} &= 2\, s_{ij} \cos\theta_{ij} \ , \end{aligned}} \qquad (6)$$

where $|e_i|$ is the length of e_i, $|f|$ is the area of f, and θ_{ij} is the angle between e_i, e_j. s_{ij} is 1 if the local orientations of e_i, e_j within f both agree or disagree with the global orientations of e_i, e_j, and -1 if one of them disagrees (i.e., whether a halfedge in f has the same orientation as an arbitrarily chosen global orientation for every edge). We use the notation $\mathrel{+}=$ to highlight that diagonal terms are visited twice when looping over all faces and edges, and both entries must be accumulated. The off-diagonal entries must be added in two places each, i.e., $L_{\alpha,\beta} = L_{\beta,\alpha}$.

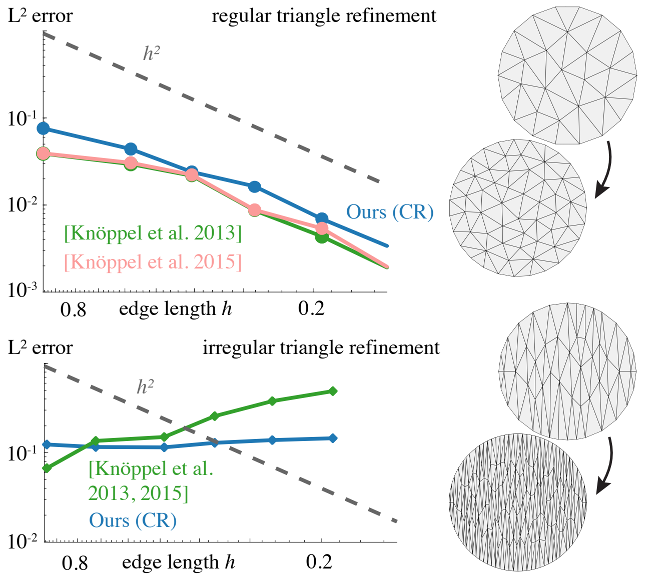


Figure 5: *Our method, like the methods of Knöppel et al. [KCPS13, KCPS15], converges to the exact solution under regular triangle refinement. If the refinement is* irregular *(for example, because the refinement adds many more vertices in one dimension than the other), no convergence is observed.*

By the same approach, the discretization of $\frac{1}{2}\int_\Omega \|\mathbf{u}\|_{\mathrm{F}}^2 \,\mathrm{d}x$, the diagonal mass matrix, is given by

$$\boxed{M_{e^{\|},e^{\|}},\, M_{e^{\perp},e^{\perp}} = \frac{A_e}{3} \ ,} \qquad (7)$$

where A_e is the sum of the areas of the two triangles adjacent to e.

The formulas for the matrices in (6) and (7) are very concise. They involve no preprocessing, and no complicated mathematical operations—simply a loop over all faces, and short formulas involving basic trigonometry with edge lengths. This is reminiscent of the simple construction of the scalar cotangent Laplacian [PP93], which has become ubiquitous in geometry processing due to its simple construction and good performance.

3.2.1. Other vector Dirichlet energies in literature

Many discretizations of the vector Dirichlet energy for non-planar triangle meshes exist, and in this section we aim to give a short overview over alternatives to our discretization. The main difference between most of these methods and our discretization is the simplicity of our matrix assembly: the matrix expressions are very simple, and the operators for the entire mesh can be constructed in one parallelizable loop over all faces using basic operations.

Knöppel et al. [KCPS13] use a finite element method with degrees of freedom on vertices. At each vertex, the vertex neighbor-

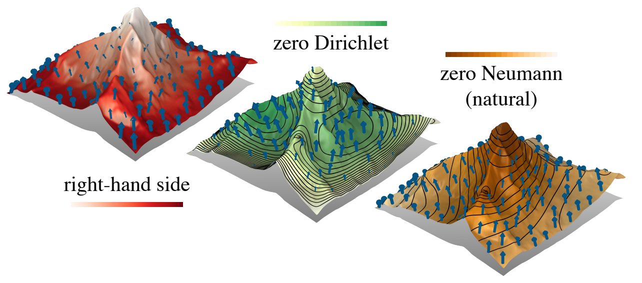


Figure 6: *Solving the Bochner-Poisson equation with nonzero right-hand side* (left)*. Fixing all boundary degrees of freedom to zero gives a solution with zero Dirichlet boundary conditions* (center)*, where all isolines are parallel to the boundary. Fixing no degrees of freedom gives the natural zero Neumann boundary conditions* (right)*, where isolines are perpendicular to the boundary.*

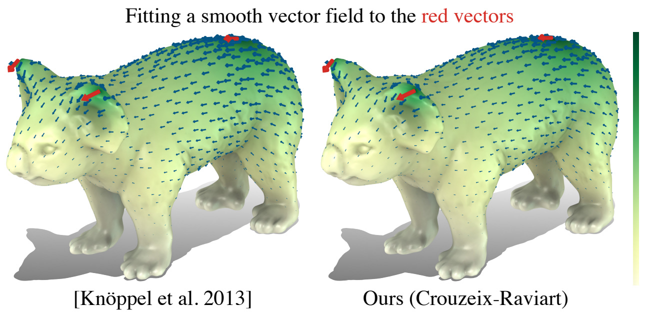


Figure 7: *Minimizing the vector Dirichlet energy subject to constraints is a useful tool for designing smooth vector fields. Our Crouzeix-Raviart discretization* (right) *produces outputs similar to the discretization of Knöppel et al. [KCPS13]* (left)*. The figure is in log scale.*

hood is locally flattened, and its tangent space then parametrized using $\mathbb{C}$. The local flattening of the vertex neighborhood requires a preprocessing step and introduces curvature at every point in the flattened triangle that needs to be accounted for when constructing finite element operators. The method of Knöppel et al. [KCPS13] is generalized by Liu et al. [LTGD16], who introduce discrete connections to improve results for certain applications. The preprocessing step for constructing globally optimal discrete operators involves solving an optimization problem. Knöppel et al. [KCPS15] and Sharp et al. [SSC19] use a finite-difference-like method that evolves the approach of Knöppel et al. [KCPS13]: the same vertex flattening preprocessing is performed, but then a finite-difference-like approach is used to construct discrete operators. Custers and Vaxman [CV18] present a subdivision scheme for per-face tangent vectors using a data structure of scalar quantities on halfedges, and apply it to vector design and optimal transport. Their basis functions are constructed using mesh subdivision. A different vertex-based approach is presented by Jakob et al. [JTPSH15] and extended by Huang et al. [HJ16].

The methods mentioned so far place their degrees of freedom (DOFs) on vertices or faces (halfedges). The fact that our method has its DOFs on edges, is neither an advantage nor a drawback compared to vertices or faces. Some applications desire DOFs on edges, as vertex DOFs can lead to locking [EB08]. For other applications, DOFs on vertices or faces might be more appropriate. For yet others, there is no preference.

Some methods do not discretize vector fields on meshes directly, and work on a proxy instead. In the work of Azencot et al. [AOCBC15], all computations are performed in the spectral domain, circumventing the need for discretizing vector fields. This approach is extended by Azencot et al. [ACBCO17]. Corman and Ovsjanikov [CO19] leverage a functional approach to compute the covariant derivative of vector fields.

3.3. Experimental evaluation

Convergence experiments for the Dirichlet boundary value problem of $\Delta_{\mathcal{B}}\mathbf{u} = 0$ solved with our Crouzeix-Raviart discretization of the vector Dirichlet energy can be found in Figure 4. We observe convergence on the order of h^2, similar to the methods of Knöppel et al. [KCPS13] and Knöppel et al. [KCPS15], Sharp et al. [SSC19].

Figure 5 shows that a certain degree of triangle regularity is required for convergence. Triangle regular refinement means that the maximum ratio between circumcircle and incircle for all triangles remains bounded. If this ratio is not appropriately bounded (*irregular* refinement) the method diverges. This regularity condition occurs in other finite element methods [Bra07].

The boundary behavior of our Crouzeix-Raviart discretization is explored in Figure 6 where the Bochner-Poisson equation $\Delta_{\mathcal{B}}\mathbf{u} = \mathbf{f}$ is solved with different boundary conditions. If all degrees of freedom (parallel and perpendicular) corresponding to boundary edges are fixed to a given value before minimizing the vector Dirichlet energy, the solution $\mathbf{u}$ is subject to *Dirichlet boundary conditions*, $\mathbf{u}|_{\partial\Omega} = \mathbf{g}$. If no degrees of freedom are explicitly fixed, *natural boundary conditions* apply, the *zero Neumann boundary conditions* $\nabla_{\mathbf{n}}\mathbf{u}|_{\partial\Omega} = 0$ (see Section 2.2). This mirrors the standard way to enforce Dirichlet and Neumann boundary conditions for the cotangent Laplacian [Bra07].

Further evaluation of the discretization, such as analysis of its performance and spectrum, can be found in Appendix A.1.

4. Applications

We use our discretization for three popular applications in geometry processing: the design of smooth vector fields, the construction of Killing fields, and efficient parallel transport of vectors. In all of these applications we take an established method and use our Crouzeix-Raviart discretization to discretize it. We achieve comparable results to previous methods.

Unless otherwise noted, figures in this section are colored by vector magnitude.

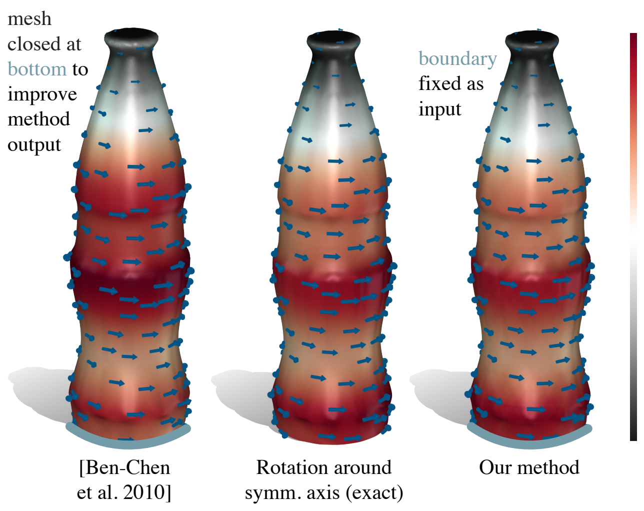


Figure 8: *A Killing field generated with our Crouzeix-Raviart discretization with regularization* $\alpha = 10^{-4}$ (right) *is similar to the vector field generated by rigidly rotating the shape around the z-axis* (center) *(ground truth), and the Killing field generated with the method of Ben-Chen et al. [BCBSG10]* (left).

4.1. Smooth vector field design

A simple application of the vector Dirichlet energy is finding the smoothest vector field given certain constraints. This is an important basic operation of vector field design, and forms a building block for many such applications [KCPS13, DVPSH14, DVPSH15, KCPS15, AOCBC15, dGDT16, LTGD16, VCD*16, BSEH18, LZC*18].

For this application we solve the following quadratic optimization problem using the Crouzeix-Raviart vector Dirichlet energy L,

$$\operatorname*{argmin}_{\mathbf{u}} \frac{1}{2}\mathbf{u}^{\intercal} L \mathbf{u}, \qquad \mathbf{u}(x_1) = \mathbf{u}_1, \ldots, \mathbf{u}(x_k) = \mathbf{u}_k \ ,$$

for given fixed points $x_1, \ldots, x_k$, and given values $\mathbf{u}_1, \ldots, \mathbf{u}_k$. The result is a smooth vector field that satisfies the given constraints.

Figure 7 shows an application of this energy minimization to design a smooth vector field on a surface. The result is similar to the vector field produced by minimizing the same smooth vector Dirichlet energy with the discretization of Knöppel et al. [KCPS13].

We can also use the vector Dirichlet energy to denoise data by smoothing it. This is achieved by performing one step of the diffusion equation, $\frac{\partial \mathbf{u}}{\partial t} = -\Delta_B \mathbf{u}$. This is a popular approach for smoothing scalar data on meshes and images [DMSB99, YKWT94]. Using our matrices L and M, the implicit Euler implementation is $(M + tL)\,\mathbf{u}_t = M\mathbf{u}_0$ for a noisy input function $\mathbf{u}_0$. Examples of denoising can be seen in Figures 1 and 9.

4.2. Killing fields

A vector field $\mathbf{u}$ on the surface Ω is called a Killing field if the geometric flow generated by following the vector field is an isometry,

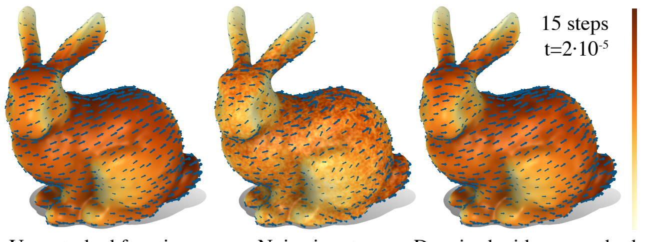


Figure 9: *Solving the vector diffusion equation, we can denoise noisy vector fields* (center) *to get a smooth result* (right) *which matches the unperturbed function before adding noise* (left).

i.e., it does not change the geometry of the shape. Such flows can be useful to find intrinsic symmetries of shapes [BCBSG10], for visualization [GMDW09], to compute deformations [SBCBG11], for surface reconstruction [SBCI17], and as a part of a larger tangent vector field processing routine [ABCCO13, AOCBC15].

The vector field $\mathbf{u}$ is a Killing field if and only if its covariant derivative $\nabla\mathbf{u}$ is a skew-symmetric tensor [Pet06, pp. 188], $(\nabla\mathbf{u})^{\intercal} = -\nabla\mathbf{u}$. This motivates the definition of a Killing energy measuring how skew-symmetric the covariant derivative of $\mathbf{u}$ is,

$$E_{\text{Killing}}(\mathbf{u}) = \frac{1}{2}\int_{\Omega} \left\|\nabla\mathbf{u} + (\nabla\mathbf{u})^{\intercal}\right\|_{\text{F}}^2 \,\mathrm{d}x \,. \tag{8}$$

We can minimize the Killing energy E_{Killing} to find fields that are as-Killing-as-possible, subject to certain constraints. The discretization of $\|\nabla\mathbf{u}\|_{\text{F}}^2$ and $\left\|(\nabla\mathbf{u})^{\intercal}\right\|_{\text{F}}^2$ is given by our usual Crouzeix-Raviart vector Dirichlet energy discretization. We also add a small amount of additional smoothing, $\alpha E(\mathbf{u})$, to the optimization as a regularizer to make it more robust—otherwise there are spurious minimizers. The mixed term containing both $\nabla\mathbf{u}$ and $(\nabla\mathbf{u})^{\intercal}$, $\int_{\Omega} \nabla\mathbf{u} : (\nabla\mathbf{u})^{\intercal} \,\mathrm{d}x$, can be straightforwardly implemented following the vector Crouzeix-Raviart approach. Its entries are, looping over all faces $f \in F$ and all pairs of consecutive edges $e_i, e_j \in f$,

$$S_{e_i^{\perp}, e_i^{\perp}} \mathrel{+}= \frac{|e_i|^2}{|f|}, \qquad S_{e_i^{\parallel}, e_j^{\parallel}} = -2 s_{ij} \sin\theta_{ij} \ ,$$
$$S_{e_i^{\perp}, e_j^{\perp}} = 2 s_{ij} \cot\theta_{ij} \cos\theta_{ij} \ , \qquad S_{e_i^{\parallel}, e_j^{\perp}}, -S_{e_i^{\perp}, e_j^{\parallel}} = 2 s_{ij} \cos\theta_{ij} \ ,$$

where $|e_i|$ is the length of edge e_i, $|f|$ is the area of face f, and θ_{ij} is the angle between e_i, e_j. s_{ij} is 1 if the local orientations of e_i, e_j within f both agree or disagree with the global orientations of e_i, e_j, and -1 if one of them disagrees. The notation $\mathrel{+}=$ signifies that diagonal terms are visited twice when looping over all faces and edges, and both entries must be accumulated. The off-diagonal entries must be added in two places each, i.e., $S_{\alpha,\beta} = S_{\beta,\alpha}$.

An example of an application of the minimization of E_{Killing} can be seen in Figure 8, where we use it to find an approximate Killing field on a surface. The result is compared to the rotation around the object's symmetry axis (an almost exact Killing field), and a discrete Killing field produced with the method of Ben-Chen et al. [BCBSG10].

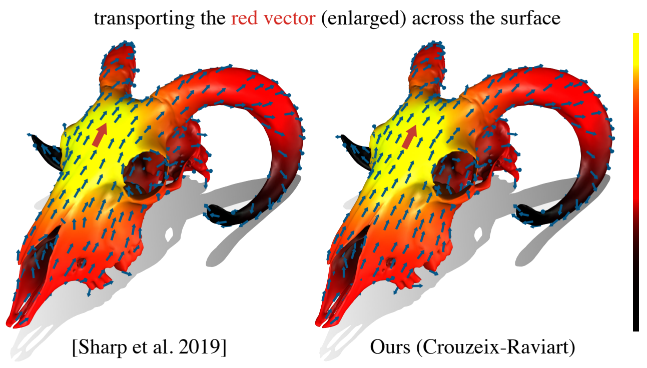


Figure 10: *Parallel transport of vectors using the heat method discretized with the Crouzeix-Raviart vector Dirichlet energy* (right) *yields very similar results to the discretization of Sharp et al. [SSC19]* (left). *The color on the mesh is the (scalar) heat distance from the prescribed vector location. The parameter t in the vector heat method is set to correspond to the mean edge length.*

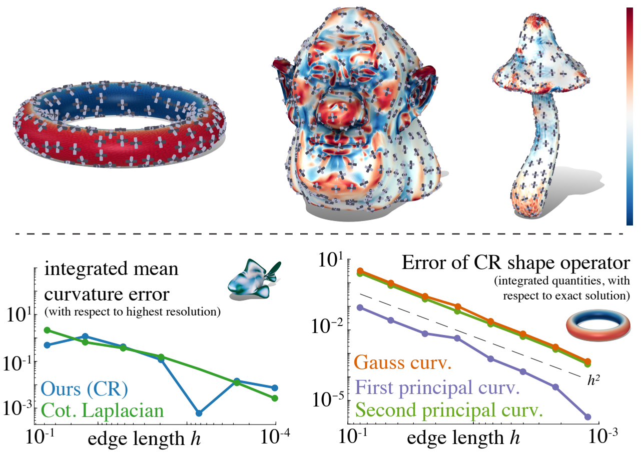


Figure 11: *Top: computing principal curvature on a variety of shapes colored by Gaussian curvature (log scale, also computed using the shape operator). Bottom: preliminary convergence experiments suggest similar behavior as other methods (mean curvature compared to cotangent Laplacian [DMSB99]).*

4.3. Efficient parallel transport

Sharp et al. [SSC19] introduce the vector heat method, an efficient way to compute the parallel transport of vectors based on the heat method for geodesics [CWW13]. They employ the Bochner Laplacian of vector fields, $\Delta_B \mathbf{u}$, to solve a vector-valued heat equation, which is then used in combination with a scalar heat equation to propagate vectors over the entire surface efficiently.

We discretize the vector heat equation using our Crouzeix-Raviart vector Dirichlet energy L and the associated Crouzeix-Raviart vector mass matrix M for the vector parts, and the scalar Crouzeix-Raviart Dirichlet energy L_{scal} and mass matrix M_{scal} for the scalar parts. The discretized vector heat equation becomes $(M+tL)\,\mathbf{v}_t = M\mathbf{v}_0$, and the discretized scalar heat equation becomes $(M_{\text{scal}}+tL_{\text{scal}})\,u_t = M_{\text{scal}}u_0$. It is easy to naively combine scalar functions and vector fields discretized with Crouzeix-Raviart: as there are two vector degrees of freedom (DOFs) for each scalar DOF, whenever a vector needs to be multiplied with a scalar (which happens in step IV of Algorithm 1 by Sharp et al. [SSC19]), both vector DOFs corresponding to the edge e are multiplied with the scalar DOF corresponding to e.

In Figure 10, we use this algorithm to efficiently compute the parallel transport of a vector across a complicated shape. The Crouzeix-Raviart discretization produces a similar result to the discretization used by Sharp et al. [SSC19]. Another example is shown in Figure 1.

5. Discussion

A limitation of this work is the lack of a convergence proof of the Crouzeix-Raviart discretization of the vector Dirichlet energy, even though we do present numerical evidence. Without such a proof, we cannot be completely sure that our approach correctly discretizes the Dirichlet energy. However, as far as we know, no convergence proofs exist for any of the discretizations of the vector Dirichlet energy for non-planar surfaces mentioned in Section 3.2.1.

Such a convergence proof is an interesting direction for future work. A convergence proof exists for flat domains $U \subseteq \mathbb{R}^2$ [Bra07, Theorem II.1.5], forming a natural starting point for a proof of convergence for our discretization. Additionally, it would be interesting to explore which features of the smooth vector Dirichlet energy carry over to our Crouzeix-Raviart discretization. Which no-free-lunch properties of Laplacian discretizations [WMKG07] hold for the vector Crouzeix-Raviart discretization? Does it admit a version of Rippa's theorem [Rip90]?

Crouzeix-Raviart finite elements can be unstable for some applications (see, for example, the discussion by Quaglino [Qua12] for the scalar case). While our discretization of the vector Dirichlet energy appears to be robust under appropriate regularity conditions, the discretization of the Killing energy of Section 4.2 requires normalization with a small amount of the vector Dirichlet energy to have a good minimizer—otherwise spurious minimizers can occur.

6. Further uses of the Crouzeix-Raviart covariant derivative

The Crouzeix-Raviart discretization of the vector Dirichlet energy is attractive because of its simplicity, which does not come at the cost of quality. The Crouzeix-Raviart approach itself, however, can also be used to discretize other differential geometric operators. While not our primary focus, and not necessarily as simple to implement as our discretization of the vector Dirichlet energy, it is intriguing that the same approach can be used to discretize a whole family of operators, and we would be remiss not to mention this.

As an example, consider the Hessian of a function u, $\nabla\nabla u$. It can be used, among other things, to construct the shape operator S of a surface [Pet06, p. 96]. The shape operator encodes all extrinsic curvature information of a shape: the largest and smallest local curvatures (the principal curvatures) and their directions. S can be expressed in terms of the coordinate functions of the embedding of the surface in $\mathbb{R}^3$, x, y, z.

$$S(\mathbf{v},\mathbf{w}) = -\mathbf{N} \cdot \begin{pmatrix} (\nabla\nabla x)(\mathbf{v},\mathbf{w}) \\ (\nabla\nabla y)(\mathbf{v},\mathbf{w}) \\ (\nabla\nabla z)(\mathbf{v},\mathbf{w}) \end{pmatrix} , \tag{9}$$

where $\mathbf{N}$ is the surface normal, and $\mathbf{v}, \mathbf{w}$ are tangent vectors. The gradient can be discretized using standard piecewise linear hat functions, and the covariant derivative can be discretized using vector Crouzeix-Raviart finite elements (see Appendix A.2).

A few preliminary results for this discretization of S can be seen in Figure 11, and suggest that integral operators derived from the shape operator can be discretized in a convergent way. It would be interesting future work to try to apply this discretization to some of the applications of the shape operator, such as the computation and optimization of geometric energies like the Willmore energy [CPS13], thin shell bending energies [GSH*04], and others, and compare the discretization to the many existing discrete shape operators [GGRZ06, GSH*04, Wei12].

A natural question is whether the vector Crouzeix-Raviart approach is suitable for *all* differential geometric operations involving vectors. This is not the case. Non-conforming (discontinuous) finite elements like Crouzeix-Raviart should always be used carefully. Just because they correctly discretize one differential operator of a certain order, does not mean that they can be used for other such operators. A simple example, for which our Crouzeix-Raviart discretization of vector fields cannot be used, is the energy $\frac{1}{2}\int_\Omega \|\operatorname{curl}\mathbf{u}\|^2 \,dx$. By definition, the curl of basis functions normal to the edge is always zero, and thus these basis functions do not contribute to the energy—any minimizer of the discretized energy can have an arbitrary normal component at any edge. For such operators, other discretization methods are more appropriate [Hip99].

Beyond the shape operator, interesting future work would be the direct application of the weak covariant derivative to vector fields directly (which does not occur in this work, as we only compute the covariant derivative in the context of energies outside of this section) [ABCCO13, dGLB*14, HH16, YCL*19], as well as discretizing higher-order vector Dirichlet energies used in data processing and learning [SW12, LHZJ13, LYHY14, Wu17] or in physics [MTW73], where the covariant derivative appears in the Einstein field equations of general relativity.

Acknowledgements

Meshes and images used in this work include Venus of Milo [smk16], cactus [Jac13], Spot the cow [Cra20], Mountain [las15], koala [Yah13], Stanford bunny [Sta14], soda bottle [dal16], goat head bone [str18], Brucewick and mushroom [Hol20].

We thank Anne Fleming, Henrique Maia, Honglin Chen, Josh Holinaty, John Kanji, and Abhishek Madan for proofreading.

This work is supported by the National Science Foundation (NSF Award IDs CCF-17-17268, IIS-1717178). This work is partially supported by the Canada Research Chairs Program and the Fields Centre for Quantitative Analysis and Modeling. This work is partially supported by the DFG project 282535003: Geometric curvature functionals: energy landscape and discrete methods.

References

[ABCCO13] Azencot O., Ben-Chen M., Chazal F., Ovsjanikov M.: An operator approach to tangent vector field processing. In *Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing* (Goslar, DEU, 2013), SGP '13, Eurographics Association, pp. 73–82. 7, 9

[ACBCO17] Azencot O., Corman E., Ben-Chen M., Ovsjanikov M.: Consistent functional cross field design for mesh quadrangulation. *ACM Trans. Graph. 36*, 4 (July 2017). 6

[AOCBC15] Azencot O., Ovsjanikov M., Chazal F., Ben-Chen M.: Discrete derivatives of vector fields on surfaces – an operator approach. *ACM Trans. Graph. 34*, 3 (May 2015). 6, 7

[BCBSG10] Ben-Chen M., Butscher A., Solomon J., Guibas L.: On discrete killing vector fields and patterns on surfaces. *Comput. Graph. Forum 29*, 5 (2010), 1701–1711. 7

[BGV96] Berline N., Getzler E., Vergne M.: Heat kernels and dirac operators. 2

[Bra07] Braess D.: *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*, 3 ed. Cambridge University Press, 2007. 3, 6, 8

[Bre15] Brenner S. C.: Forty years of the crouzeix-raviart element. *Numerical Methods for Partial Differential Equations 31*, 2 (2015), 367–396. 4

[BSEH18] Brandt C., Scandolo L., Eisemann E., Hildebrandt K.: Modeling n-symmetry vector fields using higher-order energies. *ACM Trans. Graph. 37*, 2 (Mar. 2018). 7

[BWH*06] Bergou M., Wardetzky M., Harmon D., Zorin D., Grinspun E.: A quadratic bending model for inextensible surfaces. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing* (Goslar, 2006), SGP '06, Eurographics Association, pp. 227–230. 3

[CO19] Corman E., Ovsjanikov M.: Functional characterization of deformation fields. *ACM Trans. Graph. 38*, 1 (Jan. 2019). 6

[CPS13] Crane K., Pinkall U., Schröder P.: Robust fairing via conformal curvature flow. *ACM Trans. Graph. 32*, 4 (2013). 9

[Cra20] Crane K.: Keenan's 3d model repository, 2020. `https://www.cs.cmu.edu/~kmcrane/Projects/ModelRepository/`. 9

[CV18] Custers B., Vaxman A.: Subdivision directional fields, 2018. `arXiv:1810.06884`. 6

[CWW13] Crane K., Weischedel C., Wardetzky M.: Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph. 32*, 5 (Oct. 2013). 8

[dal16] daliberator: Coca-cola bottle, Mar. 2016. `https://www.thingiverse.com/thing:1419319`. 9

[dGDT16] de Goes F., Desbrun M., Tong Y.: Vector field processing on triangle meshes. In *ACM SIGGRAPH 2016 Courses* (New York, NY, USA, 2016), SIGGRAPH '16, Association for Computing Machinery. 7

[dGLB*14] de Goes F., Liu B., Budninskiy M., Tong Y., Desbrun M.: Discrete 2-tensor fields on triangulations. In *Proceedings of the Symposium on Geometry Processing* (Goslar, 2014), SGP '14, Eurographics Association, pp. 13–24. 9

[DMSB99] Desbrun M., Meyer M., Schröder P., Barr A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., pp. 317–324. 2, 7, 8

[DVPSH14] DIAMANTI O., VAXMAN A., PANOZZO D., SORKINE-HORNUNG O.: Designing n-polyvector fields with complex polynomials. *Comput. Graph. Forum 33*, 5 (Aug. 2014), 1–11. 7

[DVPSH15] DIAMANTI O., VAXMAN A., PANOZZO D., SORKINE-HORNUNG O.: Integrable polyvector fields. *ACM Trans. Graph. 34*, 4 (July 2015). 7

[EB08] ENGLISH E., BRIDSON R.: Animating developable surfaces using nonconforming elements. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 1–5. 3, 6

[Eva92] EVANS L. C.: *Partial Differential Equations*. The American Mathematical Society, 1992. 2

[Fan19] FANG C.: *An Introduction to Fluid Mechanics*. Springer International Publishing, 2019. 2

[Gat14] GATICA G. N.: *A Simple Introduction to the Mixed Finite Element Method*. Springer Cham, 2014. 4

[GF63] GELFAND I. M., FORMIN S. V.: *Calculus of Variations*. Prentice-Hall Inc., 1963. 3

[GGRZ06] GRINSPUN E., GINGOLD Y., REISMAN J., ZORIN D.: Computing discrete shape operators on general meshes. *Comput. Graph. Forum 25*, 3 (2006), 547–556. 9

[GMDW09] GRAVE F., MÜLLER T., DACHSBACHER C., WUNNER G.: The gödel engine - an interactive approach to visualization in general relativity. In *Proceedings of the 11th Eurographics / IEEE - VGTC Conference on Visualization* (Chichester, 2009), EuroVis'09, The Eurographs Association & John Wiley & Sons, Ltd., pp. 807–814. 7

[GSH*04] GINGOLD Y., SECORD A., HAN J. Y., GRINSPUN E., ZORIN D.: A discrete model for inelastic deformation of thin shells. *preprint* (2004). 9

[HH16] HILL D. J., HENDERSON R. D.: Efficient fluid simulation on the surface of a sphere. *ACM Trans. Graph. 35*, 2 (Apr. 2016). 9

[Hip99] HIPTMAIR R.: Canonical construction of finite elements. *Math. Comp. 68* (1999), 1325–1346. 4, 9

[Hir03] HIRANI A. N.: *Discrete Exterior Calculus*. PhD thesis, California Institute of Technology, 2003. 4

[HJ16] HUANG Z., JU T.: Extrinsically smooth direction fields. *Computers & Graphics 58* (2016), 109–117. Shape Modeling International 2016. 6

[Hol20] HOLINATY J.: Meshes, 2020. provided by https://github.com/odedstein/meshes/tree/master/objects/brucewick and https://github.com/odedstein/meshes/tree/master/objects/mushroom. 9

[HP04] HILDEBRANDT K., POLTHIER K.: Anisotropic filtering of non-linear surface features. *Comput. Graph. Forum 23*, 3 (2004), 391–400. 3

[HTWB11] HUANG J., TONG Y., WEI H., BAO H.: Boundary aligned smooth 3d cross-frame field. *ACM Trans. Graph. 30*, 6 (Dec. 2011), 1–8. 3

[Jac13] JACOBSON A.: *Algorithms and Interfaces for Real-Time Deformation of 2D and 3D Shapes*. PhD thesis, ETH Zürich, 2013. 9

[JTPSH15] JAKOB W., TARINI M., PANOZZO D., SORKINE-HORNUNG O.: Instant field-aligned meshes. *ACM Trans. Graph. 34*, 6 (Oct. 2015). 6

[KCPS13] KNÖPPEL F., CRANE K., PINKALL U., SCHRÖDER P.: Globally optimal direction fields. *ACM Trans. Graph. 32*, 4 (July 2013). 4, 5, 6, 7

[KCPS15] KNÖPPEL F., CRANE K., PINKALL U., SCHRÖDER P.: Stripe patterns on surfaces. *ACM Trans. Graph. 34*, 4 (July 2015). 4, 5, 6, 7

[las15] LASTLOGINNAME: Mountain, Aug. 2015. https://www.thingiverse.com/thing:991578. 9

[Lee97] LEE J. M.: *Riemannian Manifolds*. Springer-Verlag New York Inc., 1997. 2

[LHZJ13] LIN B., HE X., ZHANG C., JI M.: Parallel vector field embedding. *J. Mach. Learn. Res. 14*, 1 (Jan. 2013), 2945–2977. 9

[LTGD16] LIU B., TONG Y., GOES F. D., DESBRUN M.: Discrete connection and covariant derivative for vector field analysis and design. *ACM Trans. Graph. 35*, 3 (Mar. 2016). 6, 7

[LYHY14] LIN B., YANG J., HE X., YE J.: Geodesic distance function learning via heat flow on vector fields. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32* (2014), ICML'14, JMLR.org, pp. II–145–II–153. 9

[LZC*18] LIU H., ZHANG P., CHIEN E., SOLOMON J., BOMMES D.: Singularity-constrained octahedral fields for hexahedral meshing. *ACM Trans. Graph. 37*, 4 (July 2018). 7

[Mac49] MACNEAL R.: *The Solution of Partial Differential Equations by Means of Electrical Networks*. PhD thesis, The California Institute of Technology, 1949. 2

[MDSB03] MEYER M., DESBRUN M., SCHRÖDER P., BARR A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III. Mathematics and Visualization*, Hege H., Polthier K., (Eds.). Springer Berlin Heidelberg, 2003. 1

[MTW73] MISNER C. W., THORNE K. S., WHEELER J. A.: Gravitation. 9

[Pet06] PETERSEN P.: *Riemannian Geometry*. Springer Science + Business Media, LLC, 2006. 2, 7, 8

[PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experiment. Math. 2*, 1 (1993), 15–36. 5

[Qua12] QUAGLINO A.: *Membrane Locking in Discrete Shell Theories*. PhD thesis, University of Göttingen, May 2012. 8

[Rip90] RIPPA S.: Minimal roughness property of the delaunay triangulation. *Computer Aided Geometric Design 7*, 6 (1990), 489–497. 8

[SBCBG11] SOLOMON J., BEN-CHEN M., BUTSCHER A., GUIBAS L.: As-killing-as-possible vector fields for planar deformation. *Comput. Graph. Forum 30*, 5 (2011), 1543–1552. 7

[SBCI17] SLAVCHEVA M., BAUST M., CREMERS D., ILIC S.: Killingfusion: Non-rigid 3d reconstruction without correspondences. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017). 7

[SJWG20] STEIN O., JACOBSON A., WARDETZKY M., GRINSPUN E.: A smoothness energy without boundary distortion for curved surfaces. *ACM Trans. Graph. 39*, 3 (Mar. 2020). 2, 3, 4, 5

[smk16] SMKMUSEUM: Venus de milo (aphrodite of milos), Nov. 2016. https://www.thingiverse.com/thing:1892710. 9

[SSC19] SHARP N., SOLIMAN Y., CRANE K.: The vector heat method. *ACM Trans. Graph. 38*, 3 (June 2019). 4, 6, 8

[Sta14] STANFORD GRAPHICS: The stanford 3d scanning repository, 2014. http://graphics.stanford.edu/data/3Dscanrep/. 9

[str18] STRONGHERO: goat head bone, July 2018. https://www.thingiverse.com/thing:2995807. 9

[SW12] SINGER A., WU H.-T.: Vector diffusion maps and the connection laplacian. *Communications on Pure and Applied Mathematics 65*, 8 (2012), 1067–1144. 9

[VCD*16] VAXMAN A., CAMPEN M., DIAMANTI O., PANOZZO D., BOMMES D., HILDEBRANDT K., BEN-CHEN M.: Directional field synthesis, design, and processing. *Comput. Graph. Forum 35*, 2 (2016). 7

[War06] WARDETZKY M.: *Discrete Differential Operators on Polyhedral Surfaces – Convergence and Approximation*. PhD thesis, FU Berlin, November 2006. 5

[WBH*07] WARDETZKY M., BERGOU M., HARMON D., ZORIN D., GRINSPUN E.: Discrete quadratic curvature energies. *Computer Aided Geometric Design 24*, 8 (2007), 499–518. Discrete Differential Geometry. 3

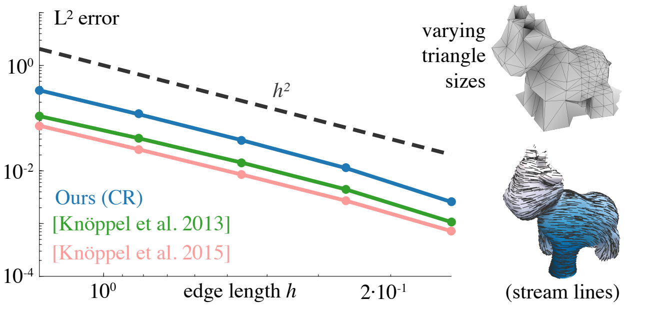


Figure 12: *Repeating the experiment from Figure 4* (right)*, with regular refinement, but on a mesh with varying triangle sizes. The same convergence behavior is still observed.*

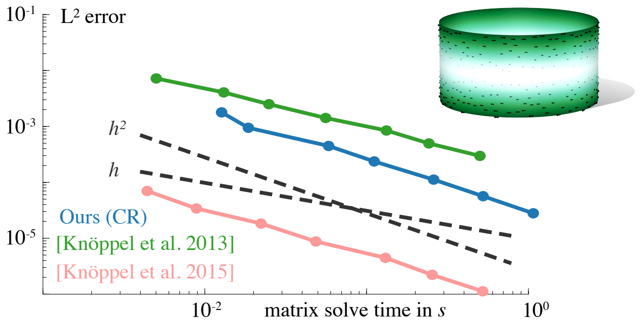


Figure 13: *Repeating experiment from Figure 4* (center)*, plotting the error with respect to matrix solve time on a 2017 13" Macbook Pro. All methods convergence at similar rates (have similar performance).*

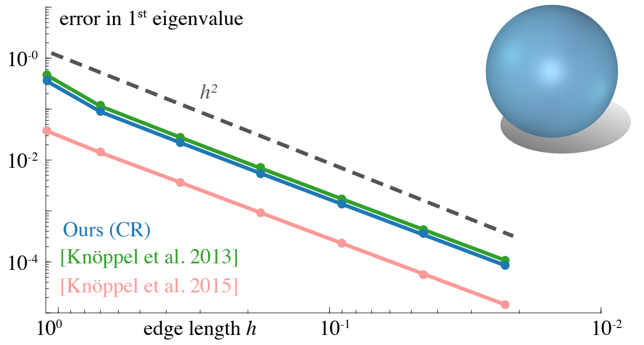


Figure 14: *Convergence plot of the first eigenvalue of the vector Dirichlet energy on the sphere, the exact solution is* 1*. Convergence is observed for all tested methods on the same order.*

[Wei12] WEISCHEDEL C.: *A Discrete Geometric View on Shear-Deformable Shell Models*. PhD thesis, University of Göttingen, 2012. 9

[WMKG07] WARDETZKY M., MATHUR S., KÄLBERER F., GRINSPUN E.: Discrete laplace operators: No free lunch. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing* (Goslar, 2007), SGP '07, Eurographics Association, pp. 33–37. 8

[Wu17] WU H.-T.: Embedding riemannian manifolds by the heat kernel of the connection laplacian. *Advances in Mathematics 304* (2017), 1055–1079. 9

[Yah13] YAHOOJAPAN: Koala, Nov. 2013. https://www.thingiverse.com/thing:182225. 9

[YCL*19] YANG B., CORSE W., LU J., WOLPER J., JIANG C.-F.: Real-time fluid simulation on the surface of a sphere. *Proc. ACM Comput. Graph. Interact. Tech. 2*, 1 (June 2019). 9

[YKWT94] YU-LI YOU, KAVEH M., WEN-YUAN XU, TANNENBAUM A.: Analysis and design of anisotropic diffusion for image processing. In *Proceedings of 1st International Conference on Image Processing* (Nov 1994), vol. 2, pp. 497–501. 7

Appendix

A.1. Further experiments

In this appendix, we present a few more evaluations of our Crouzeix-Raviart discretization of the vector Dirichlet energy. Figure 12 shows that convergence is not impacted by varying triangle sizes across a mesh, as long as refinement is still regular. In Figure 13, the error of the method with respect to runtime is evaluated, and compared with previous methods. Asymptotically, all methods perform similarly. Figure 14 analyzes the spectrum of the vector Dirichlet energy. By comparing to a known exact solution, we see that the first eigenvalue of our discretization converges on the same order as other methods.

A.2. Shape operator

To discretize the shape operator with Crouzeix-Raviart finite elements, (9) is discretized in two parts: the gradients $\nabla x, \nabla y, \nabla z$ are computed using the discrete gradient operator G that maps from the space of piecewise linear hat functions with degrees of freedom on vertices to tangent Crouzeix-Raviart vectors. Looping over all faces $f \in F$ and all triples of consecutive edges $e_i, e_j, e_k \in f$, the discrete gradient of a hat function u is

$$\begin{aligned} G(u) &= M^{-1} D(u) \\ D(u)_{e_i^{\parallel}} &\mathrel{+}= \frac{s_i |f|}{3 e_i}(u_j - u_i) \\ D(u)_{e_i^{\perp}} &\mathrel{+}= -\frac{s_i}{12|e_i|}\left(|e_i|^2(u_j+u_i) + (|e_k| - |e_j|)(u_j - u_i)\right) \\ &\quad + \frac{s_i e_i}{6} u_k \,, \end{aligned}$$

where $|e_i|$ is the length of edge e_i, u_i is the function u evaluated at the tail of e_i (with respect to the orientation of e_i within the face), and $|f|$ is the area of face f. s_i is 1 if the local orientation of e_i within f agrees with the global orientation of e_i, and -1 if it disagrees. The notation $\mathrel{+}=$ signifies that a term is visited twice when looping over all faces and edges, and both entries must be accumulated. The gradient $G(u)$ is computed for all three coordinate functions.

The discrete per-face shape operator S is computed using the discrete gradient of the coordinate functions, $G(x), G(y), G(z)$. Looping over all faces $f \in F$ and all edges $e_i \in f$, the discrete shape operator is

$$S_f \mathrel{+}= \frac{s_i|e_i|}{|f|}\left(\mathbf{N}_f \cdot \begin{pmatrix} G(x)_{e_i^{\|}} \\ G(y)_{e_i^{\|}} \\ G(z)_{e_i^{\|}} \end{pmatrix}\right)\mathbf{v}_i^{\perp}(\mathbf{v}_i^{\|})^{\mathsf{T}} + \frac{s_i|e_i|}{|f|}\left(\mathbf{N}_f \cdot \begin{pmatrix} G(x)_{e_i^{\perp}} \\ G(y)_{e_i^{\perp}} \\ G(z)_{e_i^{\perp}} \end{pmatrix}\right)\mathbf{v}_i^{\perp}(\mathbf{v}_i^{\perp})^{\mathsf{T}} ,$$

where $|e_i|$ is the length of edge e_i, $\mathbf{v}_i^{\|}$ is the normalized edge vector belonging to e_i (with respect to the orientation of e_i within the face), $\mathbf{v}_i^{\perp}$ is $\mathbf{v}_i^{\|}$ rotated by $\frac{\pi}{2}$, $\mathbf{N}_f$ is the normal vector of the face f, and $|f|$ is the area of face f. s_i is 1 if the local orientation of e_i within f agrees with the global orientation of e_i, and -1 if it disagrees. The notation $+=$ signifies that a term is visited twice when looping over all faces and edges, and both entries must be accumulated. The term involving $\mathbf{v}_i^{\|}$ is zero away from the boundary.