Discrete Differential Operators on Polygonal Meshes

FERNANDO DE GOES, Pixar Animation Studios
ANDREW BUTTS, Pixar Animation Studios
MATHIEU DESBRUN, ShanghaiTech/Caltech

Fig. 1. **Geometry processing on polygonal meshes:** Our approach enriches the available numerical toolbox for polygonal meshes by offering discrete differential operators (gradient, Laplacian, covariant derivative, shape operator, etc) acting either on discrete forms [Desbrun et al. 2008] or directional fields [Vaxman et al. 2017]. Our operators allow for the seamless extension of existing geometry processing algorithms to meshes with arbitrary 3D polygons, including non-planar and non-convex faces. Here, an example showing grooming on a quad mesh where fur and feathers are designed interactively by solving scalar and vector Poisson equations. Offset blue curves represent handles to guide the tangent direction and local shape of strands of fur and feathers.

Geometry processing of surface meshes relies heavily on the discretization of differential operators such as gradient, Laplacian, and covariant derivative. While a variety of discrete operators over triangulated meshes have been developed and used for decades, a similar construction over polygonal meshes remains far less explored despite the prevalence of non-simplicial surfaces in geometric design and engineering applications. This paper introduces a principled construction of discrete differential operators on surface meshes formed by (possibly non-flat and non-convex) polygonal faces. Our approach is based on a novel mimetic discretization of the gradient operator that is linear-precise on arbitrary polygons. Equipped with this discrete gradient, we draw upon ideas from the Virtual Element Method in order to derive a series of discrete operators commonly used in graphics that are now valid over polygonal surfaces. We demonstrate the accuracy and robustness of our resulting operators through various numerical examples, before incorporating them into existing geometry processing algorithms.

CCS Concepts: • **Mathematics of computing → Discretization**; • **Computing methodologies → Mesh models**.

Additional Key Words and Phrases: Discrete differential operators, geometry processing, polygonal meshes.

Authors' addresses: Fernando de Goes, Andrew Butts, Pixar Animation Studios, Emeryville, CA, USA; Mathieu Desbrun, California Institute of Technology, Pasadena, CA, USA, on sabbatical at SIST in ShanghaiTech University, Shanghai, China.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
© 2020 Copyright held by the owner/author(s).
0730-0301/2020/7-ART110
https://doi.org/10.1145/3386569.3392389

ACM Reference Format:
Fernando de Goes, Andrew Butts, and Mathieu Desbrun. 2020. Discrete Differential Operators on Polygonal Meshes. *ACM Trans. Graph.* 39, 4, Article 110 (July 2020), 14 pages. https://doi.org/10.1145/3386569.3392389

1 INTRODUCTION

Discrete differential operators play a central role in geometry processing, allowing for the analysis and manipulation of 3D surfaces. The vast majority of discrete differential operators assume a triangulated surface, as its simplicial and piecewise-linear nature renders the derivation and error analysis of operators particularly simple. Yet, modeling and engineering applications often rely on polygonal meshes to better capture geometric features and ease both artistic design and fabrication. Deriving discrete differential operators for polygonal meshes by first triangulating each polygonal face is ill-advised since it introduces unnecessary computational bias due to the dependence of the results on the choice of triangulation. While recent works have offered extensions of the Laplacian operator on polygonal meshes [Alexa and Wardetzky 2011; Sharp et al. 2019; Bunge et al. 2020], there remains a long list of basic discrete differential operators used in geometry processing for which variants that are valid on non-simplicial meshes do not exist.

In this paper, we introduce a systematic construction of discrete differential operators on surface meshes made out of (possibly non-planar and non-convex) polygonal faces. At its core is a new discretization of the gradient operator that is linear-precise for arbitrary polygons. From this gradient, we bootstrap the derivation of various first- and second-order discrete operators, including the covariant derivative and the shape operator, by mimicking key structural

properties of their continuous counterparts. Our approach also offers improvements over existing polygonal Laplacians, and suggests an extension of the Virtual Element Method [Beirão et al. 2013] to handle non-planar polygons. Finally, we show that our discrete operators can be seamlessly integrated into a wide range of existing geometry processing algorithms, from shape editing and suggestive contours to parameterization and vector field design.

1.1 Related Work

Before delving into our contributions, we first review prior work, restricting our exposition to discretization methods instead of surveying the large number of applications in which they are used.

Calculus on triangle meshes. Various discretizations of scalar and vector calculus over triangulated surfaces have been proposed in geometry processing, typically using a coordinate-free representation like in Discrete Exterior Calculus (DEC) [Desbrun et al. 2008; Crane et al. 2013a], or through finite element methods with conforming [Tong et al. 2003] and non-conforming [Polthier and Preuss 2003; Wardetzky 2006] linear basis functions. First- and second-order derivatives of tangent vectors and directional fields have also been proposed [de Goes et al. 2016a; Vaxman et al. 2017], involving a discrete definition of connection [Zhang et al. 2006; Knöppel et al. 2013; Liu et al. 2016], local tensor decompositions [de Goes et al. 2014], or functional maps [Azencot et al. 2015]. Alas, all these methods rely heavily on the fact that the surface mesh is simplicial.

Curvatures and shape operator. The evaluation of curvatures over surface meshes has also attracted a large amount of attention. Early on, the work of Taubin [1995] defined a discrete shape operator as a weighted average of normal curvatures. Other approaches assembled a discrete shape operator from dihedral angles between triangles [Cohen-Steiner and Morvan 2003], triangle and edge normals [Grinspun et al. 2006], or via multi-scale spherical kernels [Yang et al. 2006]. Mixed finite element and finite volume methods were used to approximate curvature values in [Meyer et al. 2003], while Hildebrandt and Polthier [2011] discretized the shape operator by leveraging vector calculus on meshes. Local surface fitting was also considered based on least-squares regression [Rusinkiewicz 2004], osculating jets [Cazals and Pouget 2005], statistical estimation [Kalogerakis et al. 2007], and Minkowski mixed volumes [Sullivan 2008]. Once again, all these approaches focus on triangle meshes.

Polygonal finite element. An important body of work to review in the context of our goals is the mechanics literature, which brims with computational methods using 2D polygonal meshes. This includes the typical use of quadrilateral meshes as well as unstructured grids such as Voronoi-like cell complexes, which are particularly convenient for meshing complex geometries [Mengolini et al. 2019]. Early work applied the finite element framework to convex polygons based on rational basis functions [Wachspress 1975], an approach later revisited and extended to non-convex polygons using generalized barycentric coordinates [Floater 2003; Sukumar and Tabarraei 2004; Gillette et al. 2016]. This formulation also covers the common case of quadrilateral elements interpolated by bilinear basis functions. Solving numerically for basis functions over non-convex polygons via the boundary element method was also proposed [Copeland 2009; Rjasanow and Weißer 2012]. However, these methods require cubature schemes to perform integration against the constructed polygonal basis functions. To bypass the evaluation of non-linear basis functions altogether, virtual node methods employ an intermediate triangulation within each flat polygon that inserts a virtual node at its centroid, and maps triangle values back to polygons via averaging [Dai et al. 2007; Tang et al. 2009]. The Mimetic Finite Difference method (MFD) [Lipnikov et al. 2014] presents instead an extension of finite volume and finite difference techniques to polygons that first discretizes a prime operator (typically, the gradient or the divergence) via a boundary integral, and then derives other operators by mimicking continuous structural properties. While MFD applied to triangulated surfaces is similar to DEC, it handles polygonal meshes as well by observing that discrete inner products must include a stabilization term to ensure they remain positive-definite for non-simplicial faces [Brezzi et al. 2005]. More recently, the Virtual Element Method (VEM) [Beirão et al. 2013] showed that MFD can, in fact, be derived from a finite element point of view where the basis functions inside the polygons are fully determined by boundary basis functions, but do not need to be constructed explicitly, hence the term "virtual element". Importantly, VEM points out that the stabilization of inner products to enforce their positive-definiteness is directly linked to the virtual functional space, thus leading to a principled construction of operators over polygonal cells [Beirão et al. 2017].

Fig. 2. **The perils of triangulating:** We compare the analytical mean curvature of a torus against a numerical approximation computed per vertex using the dot product between the vertex normal and the discrete Laplacian of vertex positions divided by twice the vertex area. The calculated mean curvatures (top) and their residuals to the ground-truth values (bottom) are color-coded per mesh vertex. While a hexagonal mesh of a torus (left) accurately captures the mean curvature of the shape, triangulating the hexagons (center and right) results in far less accurate approximations.

Polygonal Laplacians. While polygonal finite element techniques offer stable computations over 2D meshes with non-convex polygons, they cannot handle non-flat polygons typically found in geometry processing applications. In [Alexa and Wardetzky 2011], a geometric approach was described to construct a discrete Laplacian for polygonal surfaces that exploits the gradient of the magnitude of the polygonal vector area [Sullivan 2008] combined with a MFD-based inner product stabilization [Brezzi et al. 2005]. The work of Herholz et al. [2015] later characterized polygonal meshes that admit a discrete Laplacian with only non-negative weights, while Sharp et al. [2019] enriched this polygonal operator with vertex-to-vertex

rotations in order to assemble a discrete vector Laplacian. Recently, Bunge et al. [2020] adapted the virtual node method to non-flat polygons by refining each polygon with a triangle fan, emanating not from the face centroid but from the position that minimizes the sum of squared triangle areas. However, these formulations do not offer other mesh operators compatible with their polygonal Laplacian, severely limiting their applicability to geometry processing.

1.2 Contributions

In this paper, we present a principled construction of discrete differential operators on polygonal surface meshes, including covariant derivative, shape operator and Laplacians. Our work can be interpreted as a generalization of MFD/VEM [Brezzi et al. 2005; Beirão et al. 2013] to arbitrary 3D polygons based on a new discretization of the gradient operator. In the process, we define a polygonal version of DEC that adapts the stabilization term used in VEM [Beirão et al. 2013] to discrete differential forms, thus extending the work of Alexa and Wardetzky [2011] to offer polygonal operators that are both structure-preserving and compatible to our discrete gradient. Discrete derivatives of tangent vectors and directional fields are also introduced, along with a discrete shape operator. Finally, we show that all our discrete operators on polygonal meshes are local, numerically convergent under mesh refinement, and simple to retrofit into archetypal geometry processing algorithms.

2 DEFINITIONS AND NOTATIONS

In this work, we consider an arbitrary discrete surface $\mathcal{M}$ represented by an oriented two-manifold cell complex embedded in 3D. The spatial location of a mesh vertex v is indicated by $\mathbf{x}_v \in \mathbb{R}^3$, while a mesh face f is assumed to be a simple but possibly non-convex and/or non-planar 3D polygon with n_f vertices. We denote by $\mathbf{X}_f$ the matrix of size $n_f \times 3$ where the position of the vertices of f ordered counter-clockwise are stacked row-wise, i.e., $\mathbf{X}_f = \left[\mathbf{x}_{v_1} \dots \mathbf{x}_{v_{n_f}}\right]^{\mathrm{t}}$. We also denote the center of a polygonal face f as $\mathbf{c}_f = \mathbf{X}_f^{\mathrm{t}} \mathbf{1}_f / n_f$, where $\mathbf{1}_f$ is the n_f-sized vector of ones.

For each mesh face f, we make use of two auxiliary matrices $\mathbf{D}_f$ and $\mathbf{A}_f$ of size $n_f \times n_f$. The matrix $\mathbf{D}_f$ computes the difference of any vertex-based values between consecutive vertices along the face, i.e., $\mathbf{D}_f^{i,i+1} = 1$, $\mathbf{D}_f^{i,i} = -1$, and zero otherwise. Conversely, the matrix $\mathbf{A}_f$ computes the average of values between pairs of consecutive vertices of the face, i.e., $\mathbf{A}_f^{i,i+1} = \mathbf{A}_f^{i,i} = 1/2$, and $\mathbf{A}_f^{i,j} = 0$ otherwise. Note that $\mathbf{A}_f^{\mathrm{t}} \mathbf{D}_f + \mathbf{D}_f^{\mathrm{t}} \mathbf{A}_f = 0$ by definition. Using these two matrices, we can express the edge vectors of the polygon f via $\mathbf{E}_f = \mathbf{D}_f \mathbf{X}_f$ and its edge midpoints by $\mathbf{B}_f = \mathbf{A}_f \mathbf{X}_f$. These face attributes are summarized in Table 1 for the reader's convenience.

We discretize the restriction of any scalar function ϕ to a face f by defining a vector $\boldsymbol{\phi}_f$ of size n_f, where the i-th entry of $\boldsymbol{\phi}_f$ indicates the value of ϕ at the i-th vertex of f in counter-clockwise order, i.e., $\boldsymbol{\phi}_f = \left[\phi(v_1) \dots \phi(v_{n_f})\right]^{\mathrm{t}}$. With this notation, a linear function on face f must be of the form:

$$\boldsymbol{\phi}_f = \mathbf{X}_f \boldsymbol{s} + \mathbf{1}_f r \quad \text{for} \quad \boldsymbol{s} \in \mathbb{R}^3,\ r \in \mathbb{R}. \tag{1}$$

Lastly, we indicate by $\mathbf{I}$ the identity matrix, whose dimension will be clear from context, and denote by $[\boldsymbol{p}]$ the map from a 3D vector $\boldsymbol{p}$ to a 3×3 skew-symmetric matrix such that $[\boldsymbol{p}]\boldsymbol{q} = \boldsymbol{p} \times \boldsymbol{q}$ for any $\boldsymbol{q} \in \mathbb{R}^3$.

Table 1. Summary of the main symbols per face f and their definitions.

Symbol	Meaning	Definition
n_f	Number of vertices	$v_1, \dots, v_{n_f} \in f$
$\mathbf{X}_f$	Vertex positions	$\mathbf{X}_f = \left[\mathbf{x}_{v_1} \dots \mathbf{x}_{v_{n_f}}\right]^{\mathrm{t}} \in \mathbb{R}^{n_f \times 3}$
$\mathbf{D}_f$	Difference operator	$\mathbf{D}_f^{i,i+1} = 1,\ \mathbf{D}_f^{i,i} = -1$
$\mathbf{A}_f$	Average operator	$\mathbf{A}_f^{i,i+1} = \mathbf{A}_f^{i,i} = 1/2$
$\mathbf{E}_f$	Edge vectors	$\mathbf{E}_f = \mathbf{D}_f \mathbf{X}_f$
$\mathbf{B}_f$	Edge midpoints	$\mathbf{B}_f = \mathbf{A}_f \mathbf{X}_f$
$\mathbf{c}_f$	Face center	$\mathbf{c}_f = \mathbf{X}_f^{\mathrm{t}} \mathbf{1}_f / n_f$
$\mathbf{a}_f$	Polygonal vector area	$\mathbf{a}_f = 1/2 \sum_{v_i \in f} \mathbf{x}_{v_i} \times \mathbf{x}_{v_{i+1}}$
a_f	Area of polygonal face	$a_f = \lvert \mathbf{a}_f \rvert$
$\mathbf{n}_f$	Normal of polygonal face	$\mathbf{n}_f = \mathbf{a}_f / a_f$
$\mathbf{h}_f$	Vertex heights for polygonal face	$\mathbf{h}_f = (\mathbf{X}_f - \mathbf{1}_f \mathbf{c}_f^{\mathrm{t}}) \mathbf{n}_f$

Unless explicitly stated, the definitions of our discrete differential operators will always be provided through their restriction to a given polygonal face f.

3 GRADIENT OPERATOR

We begin by introducing a new discrete gradient operator over arbitrary 3D polygons. In order to extend the traditional finite element discretization from triangulated to polygonal meshes, we wish to construct for each polygonal face f a matrix $\mathbf{G}_f$ of size $3 \times n_f$ such that applying this matrix to the vector $\boldsymbol{\phi}_f$ (which encodes a scalar function ϕ restricted to f) returns a 3D vector $\mathbf{G}_f \boldsymbol{\phi}_f$ offering an approximation of the gradient $\nabla\phi$. A usual means to define a discrete gradient is to evaluate its integral over each mesh face. In the case of simplicial elements or flat polygonal faces where the surface normal is constant locally, this integral can be expressed as a boundary integral through Stokes' theorem:

$$\int_f \nabla\phi(\mathbf{x})\, d\mathbf{x} = \oint_{\partial f} \phi(\mathbf{x})\, \mathbf{t}(\mathbf{x}) \times \mathbf{n}(\mathbf{x})\, d\mathbf{x}, \tag{2}$$

where $\mathbf{n}(\mathbf{x})$ is the unit surface normal vector and $\mathbf{t}(\mathbf{x})$ is the unit tangent vector at boundary point $\mathbf{x}$. Consequently, the discrete gradient can be evaluated purely based on $\boldsymbol{\phi}_f$, the face edge vectors and the constant face normal. However, Eq. (2) no longer holds for *non-planar* polygons as the normal field $\mathbf{n}(\mathbf{x})$ now varies based on the choice of a particular surface interpolating the polygonal boundary. Therefore, there exists no canonical expression for this weak form of the gradient defined over arbitrary 3D polygons.

To bypass the need for an interpolant, we propose instead to evaluate a weak form of the cogradient operator $\nabla\phi^{\perp} = [\mathbf{n}(\mathbf{x})]\nabla\phi$, i.e., the gradient $\nabla\phi$ locally rotated by $\pi/2$ around the surface normal $\mathbf{n}(\mathbf{x})$. Applying Stokes' theorem, we obtain

$$\int_f \nabla\phi^{\perp}(\mathbf{x})\, d\mathbf{x} = \oint_{\partial f} \phi(\mathbf{x})\, \mathbf{t}(\mathbf{x})\, d\mathbf{x}, \tag{3}$$

which now involves only the boundary tangent, independent of how the surface interpolates the polygonal boundary. Since this tangent field is well-defined along the polygonal boundary, we can exactly

evaluate Eq. (3) for linear functions ϕ as a sum over each polygonal edge of the average of the values of ϕ along the edge multiplied by the edge vector, that is, the integrated cogradient of ϕ is $\mathbf{E}_f^{\mathrm{t}}\mathbf{A}_f\boldsymbol{\phi}_f$.

We can also use the weak form of the cogradient to relate linear functions to the surface normals $\mathbf{n}(\mathbf{x})$ integrated over the polygon f. Since the cogradient of a linear function $\phi(\mathbf{x}) = \boldsymbol{s}^{\mathrm{t}}\mathbf{x} + r$ should return the vector $[\mathbf{n}(\mathbf{x})]\boldsymbol{s}$, we can expand the left-hand side of Eq. (3) into

$$\int_f \nabla\phi^{\perp}(\mathbf{x})\,d\mathbf{x} = \left(\int_f [\mathbf{n}(\mathbf{x})]\,d\mathbf{x}\right)\boldsymbol{s}. \tag{4}$$

Combining this result with the boundary integral of Eq. (3) applied to linear functions (whose discrete form is given in Eq. (1)), we have

$$\left(\int_f [\mathbf{n}(\mathbf{x})]\,d\mathbf{x}\right)\boldsymbol{s} = \mathbf{E}_f^{\mathrm{t}}\mathbf{A}_f\left(\mathbf{X}_f\boldsymbol{s} + \mathbf{1}_f r\right) = \mathbf{E}_f^{\mathrm{t}}\mathbf{B}_f\boldsymbol{s}, \tag{5}$$

where the term $\mathbf{E}_f^{\mathrm{t}}\mathbf{A}_f\mathbf{1}_f$ cancels out due to the edge vectors forming a closed path. We thus conclude that $\mathbf{E}_f^{\mathrm{t}}\mathbf{B}_f$ encodes the integrated normals over 3D polygons, agnostic to any surface interpolant.

Remarkably, the matrix $\mathbf{E}_f^{\mathrm{t}}\mathbf{B}_f$ also reveals a simple geometric construction of a normal vector constant over any 3D polygon. As shown by Lemma 1 of [Alexa and Wardetzky 2011], the matrix $\mathbf{E}_f^{\mathrm{t}}\mathbf{B}_f$ is equivalent to the skew-symmetric matrix $[\mathbf{a}_f]$ associated with the vector area $\mathbf{a}_f$ of the polygonal face f [Sullivan 2008], where

$$\mathbf{a}_f = \frac{1}{2}\oint_{\partial f} \mathbf{x}\times\mathbf{t}(\mathbf{x})\,d\mathbf{x} = \frac{1}{2}\sum_{v_i\in f} \mathbf{x}_{v_i}\times\mathbf{x}_{v_{i+1}}. \tag{6}$$

As detailed in Alexa and Wardetzky [2011], the magnitude of the polygonal vector area $\mathbf{a}_f$ corresponds to the largest area over all orthogonal projections of the polygon f onto 3D planes, while the the direction of $\mathbf{a}_f$ represents the normal of the projection plane maximizing the area. We thus define the area of a polygonal face f as $a_f = |\mathbf{a}_f|$, and assign $\mathbf{n}_f = \mathbf{a}_f/a_f$ as its constant normal vector.

Equipped with this polygonal vector area, we can now mimic the relation $\nabla\phi^{\perp} = [\mathbf{n}(\mathbf{x})]\nabla\phi$ between gradient and cogradient in the smooth setting and finally deduce our discrete gradient operator from the weak form of the cogradient, resulting in

$$\boxed{\mathbf{G}_f = -\frac{1}{a_f}[\mathbf{n}_f]\,\mathbf{E}_f^{\mathrm{t}}\mathbf{A}_f.} \tag{7}$$

We can further isolate the contribution of each vertex in f from our gradient matrix $\mathbf{G}_f$ to better understand the geometric construction inherent to our operator. By expanding $\mathbf{G}_f\boldsymbol{\phi}_f = \sum_{v\in f}\mathbf{g}_f^v\,\phi(\mathbf{x}_v)$, we denote the gradient vector associated with the vertex v in f as

$$\mathbf{g}_f^v = \frac{1}{2a_f}\mathbf{n}_f\times\left(\mathbf{x}_{v_{i-1}} - \mathbf{x}_{v_{i+1}}\right). \tag{8}$$

As expected, our gradient vector $\mathbf{g}_f^v$ for a triangular face matches the gradient of barycentric coordinates as commonly found in piecewise linear finite element methods. Importantly, our construction of the discrete gradient is linear-precise as shown in the following lemma.

Lemma 1. *For any linear function restricted to a 3D polygonal face f (Eq. (1)), the discrete gradient in Eq. (7) is linear-precise, i.e.,*

$$\mathbf{G}_f\left(\mathbf{X}_f\boldsymbol{s} + \mathbf{1}_f r\right) = \left(\mathbf{I} - \mathbf{n}_f\mathbf{n}_f^{\mathrm{t}}\right)\boldsymbol{s}. \tag{9}$$

Fig. 3. **Convergence plots:** In the top row, L_2 error plots in log-log scale for a variety of polygonal meshes indicate at least linear convergence for our discrete gradient operator as a function of the mesh resolution, computed by evaluating the function $\sin(4\pi x)\cos(4\pi y)$ at mesh vertices and comparing its face gradient to the ground-truth value. In the bottom row, the Frobenius norm of the difference between our discrete covariance derivative and its ground-truth values exhibits a similar behavior, computed using the analytical tangent vector $\sqrt{1-z^2}\,[-y, x, 0]$ sampled at vertices of polygonal meshes approximating a unit sphere.

Proof: For a linear function as in Eq. (1), the face gradient is

$$\mathbf{G}_f\left(\mathbf{X}_f\mathbf{s} + \mathbf{1}_f r\right) = -\frac{1}{a_f}[\mathbf{n}_f]\left(\mathbf{E}_f^{\mathrm{t}}\mathbf{B}_f\mathbf{s} + r\,\mathbf{E}_f^{\mathrm{t}}\mathbf{A}_f\mathbf{1}_f\right) = -[\mathbf{n}_f]^2\,\mathbf{s},$$

where the first term $\mathbf{E}_f^{\mathrm{t}}\mathbf{B}_f$ is replaced by $[\mathbf{a}_f] = a_f[\mathbf{n}_f]$. Noting that $-[\mathbf{n}]^2 = \mathbf{I} - \mathbf{n}\mathbf{n}^{\mathrm{t}}$ for any unit vector $\mathbf{n}$ completes the proof. □

With this discrete gradient, we can now derive discrete operators acting on either discrete forms or directional fields defined on $\mathcal{M}$, as we detail in the next two sections.

4 OPERATORS ON DISCRETE FORMS

Discrete exterior calculus (DEC) [Desbrun et al. 2008] provides a coordinate-free discretization of calculus on discrete manifolds which has been shown versatile for geometry processing [Crane et al. 2013a]. In this numerical framework, differential operators act solely on discrete forms, i.e., values assigned to mesh elements that encode either functions evaluated at vertices, tangent vectors through their circulation along oriented edges, or densities integrated over faces.

Next, we exploit our discrete gradient to derive a discrete exterior calculus valid on arbitrary polygonal meshes. In the process, we point out differences between our results and the similar endeavor of Alexa and Wardetzky [2011], and discuss how our approach can be seen as an extension of the Virtual Element Method [Beirão et al. 2013] to discrete forms over non-flat polygons.

4.1 Exterior Derivatives

The derivatives of discrete forms are computed by discretizing the exterior derivative operators. Similar to the simplicial case [Desbrun et al. 2008], we construct two discrete exterior derivatives over polygonal meshes using the transpose of the signed adjacency matrices of $\mathcal{M}$. The discrete exterior derivative for 0-forms is expressed by a sparse matrix $\boldsymbol{d}_0$ that maps the differences of vertex values to a 1-form assigned to their corresponding oriented edges, while the discrete exterior derivative for 1-forms is another sparse matrix $\boldsymbol{d}_1$ that sums the oriented edge circulations into a 2-form at their respective faces. Even in our polygonal case, these discrete exterior derivatives trivially verify $\boldsymbol{d}_1\boldsymbol{d}_0 = 0$, thus ensuring that they produce a de-Rham sequence. For notational convenience, we express these derivatives through restriction to a single polygonal face f as we will systematically derive our other operators per face. To this end, we reshape the values of a given discrete k-form ($k = 0, 1$) incident to face f into a n_f-sized vector with entries ordered counter-clockwise, thus defining a local discrete k-form per face f. The discrete operator corresponding to $\boldsymbol{d}_0$ restricted to the face f reduces to the matrix $\mathbf{D}_f$ defined in §2, which computes the difference of values between consecutive vertices. Similarly, the restriction of $\boldsymbol{d}_1$ to face f simplifies to $\mathbf{1}_f^{\mathsf{t}}$ as it returns the total circulation of a 1-form summed over the polygonal edges.

4.2 Sharp and Flat Operators

The sharp $\sharp$ and flat $\flat$ operators, nicknamed musical isomorphisms, convert 1-forms to tangent vector fields and vice-versa [Abraham et al. 1988]. Although these two operators are rarely used in DEC applications since tangent vectors are manipulated solely through their 1-form representation, they are key to establishing the equivalence between exterior calculus and vector calculus. Therefore, we provide expressions for their discrete counterparts on arbitrary polygonal meshes so as to leverage our gradient operator in the discretization of differential operators on discrete forms.

The *discrete flat* operator should map a single vector on a polygonal face f into a local discrete 1-form encoding the vector circulation along the edges of f. We can thus construct this operation through a matrix $\mathbf{V}_f$ of size $n_f \times 3$ defined as

$$\boxed{\mathbf{V}_f = \mathbf{E}_f \left(\mathbf{I} - \mathbf{n}_f \mathbf{n}_f^{\mathsf{t}}\right).} \tag{10}$$

Applying the discrete flat operator $\mathbf{V}_f$ to an input 3D vector first extracts its tangential part, and then computes its circulation along each edge of f oriented counter-clockwise. By construction, assuming that the polygonal face is not degenerate, the rank of $\mathbf{V}_f$ is 2 and its kernel only contains 3D vectors along $\mathbf{n}_f$. Note that this operator is only suited for local computations per face. If a piecewise-constant vector field over a mesh needs to be mapped into a discrete 1-form, one should average on each non-boundary edge the result of the discrete flat operators computed on its two adjacent faces.

The *discrete sharp* operator should, conversely, aggregate the values of a discrete 1-form into a tangent vector per face of the polygonal mesh. We construct the discrete sharp operator restricted to a face f as a matrix $\mathbf{U}_f$ of size $3 \times n_f$ expressed as

$$\boxed{\mathbf{U}_f = \frac{1}{a_f} [\mathbf{n}_f] \left(\mathbf{B}_f^{\mathsf{t}} - \mathbf{c}_f \mathbf{1}_f^{\mathsf{t}}\right).} \tag{11}$$

Note that, here again, the rank of $\mathbf{U}_f$ is 2 for non-degenerate polygons, and its kernel is of size $n_f - 2$. Importantly, this sharp operator links our discrete gradient to the discrete exterior derivative of 0-forms, reproducing the continuous definition $\nabla\phi = (\mathbf{d}\phi)^{\#}$ for any scalar function ϕ.

Lemma 2. *The operator $\mathbf{U}_f$ satisfies $\mathbf{G}_f = \mathbf{U}_f \mathbf{D}_f$ on each face f.*

Proof: $\left(\mathbf{B}_f^{\mathsf{t}} - \mathbf{c}_f \mathbf{1}_f^{\mathsf{t}}\right)\mathbf{D}_f = \mathbf{X}_f^{\mathsf{t}} \mathbf{A}_f^{\mathsf{t}} \mathbf{D}_f = -\mathbf{E}_f^{\mathsf{t}} \mathbf{A}_f$ due to $\mathbf{D}_f^{\mathsf{t}} \mathbf{A}_f + \mathbf{A}_f^{\mathsf{t}} \mathbf{D}_f = 0$. Thus, we deduce $\mathbf{U}_f \mathbf{D}_f = -(1/a_f)\,[\mathbf{n}_f] \mathbf{E}_f^{\mathsf{t}} \mathbf{A}_f = \mathbf{G}_f$ by Eq. (7). □

Geometrically, each column of $\mathbf{U}_f$ represents a 3D vector associated with the barycentric dual mesh that connects the face center $\mathbf{c}_f$ to edge midpoints rotated by $\pi/2$ around the face normal and divided by the polygonal area. Instead of $\mathbf{c}_f$ in Eq. (11), any other face center could have been used without altering the result of Lemma 2. However, we chose $\mathbf{c}_f$ as an unbiased center which further reproduces, in the special case of triangular faces, the evaluation of 1-form Whitney basis functions at the face centroid $\mathbf{c}_f$.

Lemma 3. *In the simplicial case, applying the sharp operator $\mathbf{U}_f$ to a local discrete 1-form is equivalent to the tangent vector resulting from the area-average of the corresponding sharpened discrete 1-form reconstructed via Raviart-Thomas edge elements.*

Proof: Whitney 1-form elements [Desbrun et al. 2008] sharpened into a vector field correspond to Raviart-Thomas edge basis functions [Raviart and Thomas 1977]. From the oriented edge values of a simplicial mesh, these edge basis functions produce a piecewise linear vector field inside each mesh face. As noted in [Auchmann and Kurz 2006], using barycentric dual edges linking the barycenter of a triangle to the edge midpoints corresponds to a pointwise evaluation of this Raviart-Thomas vector field reconstruction at the face barycenter. Since the value of a function at the barycenter is a quadrature known to be exact for linear functions, our sharp operator is thus equal to the integral of the piecewise reconstructed vector field on the triangle face divided by its area. □

4.3 Projection Operator

The musical operators are both isomorphisms in the continuous setting in the sense that they only convert quantities (by raising or lowering the indices in their tensorial notation) without any loss of information [Abraham et al. 1988], and thus they should be the exact inverse of each other. Unfortunately, our discretization of the flat and sharp operators only preserves this structure partially, just like in the simplicial case [Hirani 2003]. For instance, a vector $\boldsymbol{s} \in \mathbb{R}^3$ tangent to a polygonal face f (that is, satisfying $\boldsymbol{s}^{\mathsf{t}} \mathbf{n}_f = 0$) can be flattened to a local discrete 1-form, and a further sharpening of this 1-form returns the same vector $\boldsymbol{s}$, i.e., $\mathbf{U}_f \mathbf{V}_f \boldsymbol{s} = \boldsymbol{s}$ as we prove next.

Fig. 4. **Laplacian accuracy:** On various polygonal discretizations of a unit sphere, we show the L_2 error in log-log scale of a Poisson solve using our discrete Laplacian operator with the right-hand side set to the spherical harmonic function $Y_2^4(x, y, z) = (x^2 - y^2)(7z^2 - 1)$ sampled at mesh vertices, versus this same function scaled by the inverse of its associated eigenvalue. As expected, errors decrease as the mesh resolution increases, measured using the inverse of the mean edge length. While the choice of stabilization parameter λ affects the overall accuracy of our approach, the same convergence trend is witnessed for all λ's, with $\lambda = 1$ being invariably best. Compared to [Alexa and Wardetzky 2011] and [Bunge et al. 2020] (labeled AW11 and BHKB20, respectively), our method exhibits similar accuracy with a small but noticeable improvement on most meshes.

Lemma 4. *Given an arbitrary 3D polygon f, the discrete musical operators satisfy* $\mathbf{U}_f \mathbf{V}_f = \mathbf{I} - \mathbf{n}_f \mathbf{n}_f^{\mathsf{t}}$.

Proof: Combining Eqs. (10) and (11), and using $[\mathbf{a}_f] = \mathbf{E}_f^{\mathsf{t}} \mathbf{B}_f$, yields

$$\mathbf{U}_f \mathbf{V}_f = (1/a_f)\,[\mathbf{n}_f][\mathbf{a}_f]^{\mathsf{t}} (\mathbf{I} - \mathbf{n}_f \mathbf{n}_f^{\mathsf{t}}) = \mathbf{I} - \mathbf{n}_f \mathbf{n}_f^{\mathsf{t}}. \qquad \square$$

However, a similar property does not hold if one applies sharp and flat in the reverse order. Indeed, our sharp operator is rank-deficient as it reduces n_f values from a discrete 1-form on face f down to only a vector tangent to f. Taking a cue from the functional space projection used in VEM [Beirão et al. 2013] and adapting it to the case of discrete 1-forms, we introduce a *projection* operator $\mathbf{P}_f$ for each face f defined via a scale-invariant matrix of size $n_f \times n_f$:

$$\boxed{\mathbf{P}_f = \mathbf{I} - \mathbf{V}_f \mathbf{U}_f.} \tag{12}$$

This operator acts on discrete 1-forms by quantifying the failure of the discrete sharp and flat operators to be the inverse of each other. In essence, it takes a discrete 1-form and removes its component that would result in a 3D tangent vector once sharpened. Consequently, the operator $\mathbf{P}_f$ annihilates all discrete 1-forms in the image of $\mathbf{V}_f$.

Lemma 5. *For any polygonal face f, one has* $\mathbf{P}_f \mathbf{V}_f = 0$.

Proof: Expanding Eq. (12) and using Lemma 4, one finds

$$\mathbf{P}_f \mathbf{V}_f = \mathbf{V}_f - \mathbf{V}_f (\mathbf{U}_f \mathbf{V}_f) = \mathbf{V}_f - \mathbf{V}_f (\mathbf{I} - \mathbf{n}_f \mathbf{n}_f^{\mathsf{t}}) = 0. \qquad \square$$

We can use Lemma 5 to show that $\mathbf{P}_f$ is an idempotent matrix, thus confirming that $\mathbf{P}_f$ defines a projection operator.

Lemma 6. *For any polygonal face f, one has* $\mathbf{P}_f^2 = \mathbf{P}_f$.

Proof: Trivially by Lemma 5, $\mathbf{P}_f \mathbf{P}_f = \mathbf{P}_f - (\mathbf{P}_f \mathbf{V}_f) \mathbf{U}_f = \mathbf{P}_f$. $\square$

From Lemma 6, we deduce that $\operatorname{Im} \mathbf{P}_f = \{\boldsymbol{w} \in \mathbb{R}^{n_f} \mid \mathbf{P}_f \boldsymbol{w} = \boldsymbol{w}\}$, which implies that $\operatorname{Ker} \mathbf{P}_f \oplus \operatorname{Im} \mathbf{P}_f = \mathbb{R}^{n_f}$. We can also relate the image of the projection matrix $\mathbf{P}_f$ to our sharp operator $\mathbf{U}_f$.

Lemma 7. *For any polygonal face f, one has* $\operatorname{Im} \mathbf{P}_f = \operatorname{Ker} \mathbf{U}_f$.

Proof: We must prove that, for any discrete 1-form $\boldsymbol{w} \neq 0$ local to the face f, one has $\mathbf{U}_f \boldsymbol{w} = 0 \Leftrightarrow \mathbf{P}_f \boldsymbol{w} = \boldsymbol{w}$. If $\mathbf{U}_f \boldsymbol{w} = 0$, then $\mathbf{P}_f \boldsymbol{w} = \boldsymbol{w} - \mathbf{V}_f (\mathbf{U}_f \boldsymbol{w}) = \boldsymbol{w}$. Conversely, $\mathbf{P}_f \boldsymbol{w} = \boldsymbol{w}$ implies $\mathbf{V}_f \mathbf{U}_f \boldsymbol{w} = 0$ by Eq. (12). Since $\mathbf{U}_f \boldsymbol{w}$ generates a tangent vector but the kernel of $\mathbf{V}_f$ contains only normal vectors, there exists no 1-form $\boldsymbol{w} \neq 0$ such that both $\mathbf{U}_f \boldsymbol{w} \neq 0$ and $\mathbf{V}_f \mathbf{U}_f \boldsymbol{w} = 0$, thus completing the proof. $\square$

Using these results, we conclude that the kernel of the projection operator $\mathbf{P}_f$ is indeed equal to the subspace spanned by $\mathbf{V}_f$.

Lemma 8. *For any polygonal face f, one has* $\operatorname{Ker} \mathbf{P}_f = \operatorname{Im} \mathbf{V}_f$.

Proof: From Lemma 7, we deduce that $\dim(\operatorname{Im} \mathbf{P}_f) = n_f - 2$. Since $\operatorname{Ker} \mathbf{P}_f \oplus \operatorname{Im} \mathbf{P}_f = \mathbb{R}^{n_f}$, we can conclude that $\dim(\operatorname{Ker} \mathbf{P}_f) = 2$. Moreover, Lemma 5 implies that $\operatorname{Im} \mathbf{V}_f \subset \operatorname{Ker} \mathbf{P}_f$, but since these spaces have the same rank, we have $\operatorname{Ker} \mathbf{P}_f = \operatorname{Im} \mathbf{V}_f$. $\square$

The projection operator as well as its properties will be essential to enforce positive-definiteness of the inner product of 1-forms on arbitrary polygonal faces as we discuss next.

4.4 Inner Products

Defining inner products of discrete forms requires care as one must ensure that these operators are symmetric, positive-definite, scale-aware, and exact for constant fields [Auchmann and Kurz 2006]. For 0- and 2-forms, the corresponding discrete inner products are simply encoded using local areas associated to vertices and faces, respectively. Restricted to a single polygonal face f, the inner product for discrete 0-forms is set to the $n_f \times n_f$ diagonal matrix $(a_f / n_f)\,\mathbf{I}$ where we evenly distribute the polygonal area a_f to its vertices, while the inner product for discrete 2-forms is simply the inverse of the polygonal area $1/a_f$.

To form the inner product for discrete 1-forms, we can emulate the continuous setting and compute the dot product between the tangent vectors resulting from our discrete sharp operator, thus suggesting a matrix per face f of size $n_f \times n_f$ equal to $a_f\, \mathbf{U}_f^{\mathsf{t}} \mathbf{U}_f$. However, as discussed in §4.3, the sharp operator is rank-deficient, making the discrete inner product only positive semi-definite. Leveraging Lemma 7, we can now use the projection operator $\mathbf{P}_f$ as a stabilization term that penalizes the kernel of $\mathbf{U}_f$. We thus define

the inner product matrix for local discrete 1-forms as

$$\boxed{\mathbf{M}_f = a_f\,\mathbf{U}_f^{\mathsf{t}}\mathbf{U}_f + \lambda\,\mathbf{P}_f^{\mathsf{t}}\mathbf{P}_f,} \tag{13}$$

where the parameter $\lambda > 0$ weights the contribution of the projection term. We now show that our construction of the discrete inner product satisfies important structural properties.

Lemma 9. *The matrix $\mathbf{M}_f$ is symmetric, positive-definite, scale-invariant, and constant-precise.*

Proof: The matrix $\mathbf{M}_f$ is trivially symmetric and scale-invariant. The squared norm of a discrete 1-form $\boldsymbol{w}$ can be expressed as

$$\boldsymbol{w}^{\mathsf{t}}\mathbf{M}_f\boldsymbol{w} = a_f\|\mathbf{U}_f\boldsymbol{w}\|^2 + \lambda\|\mathbf{P}_f\boldsymbol{w}\|^2.$$

Using Lemma 7, we deduce that only trivial discrete 1-forms have zero norm, thus confirming that $\mathbf{M}_f$ is positive-definite. Finally, given a constant vector $\boldsymbol{s}$ tangent to face f, we can use Lemmas 4 and 5 to show that the squared norm of the corresponding 1-form local to f is $(\mathbf{V}_f\boldsymbol{s})^{\mathsf{t}}\mathbf{M}_f(\mathbf{V}_f\boldsymbol{s}) = a_f\|\boldsymbol{s}\|^2$, therefore, our discrete inner product of 1-forms is constant-precise. □

We point out that our construction of the discrete inner product is an extension of the VEM stabilization proposed in [Beirão et al. 2013]. Indeed, VEM also uses a projection operator as a stabilization to the inner product matrix, but its projection operator measures how a scalar function fails to be captured by their discrete functional space locally. Our work adapts the notion of projection to discrete 1-forms instead of functions by evaluating the lack of isomorphism between the local sharp and flat operators. Additionally, while VEM only considered polygons in the plane, our stabilization is valid for arbitrary 3D polygons, including non-flat faces commonly found in surface meshes. The case of near-degenerate polygons is also simple to handle: as the magnitude of the vector area goes to zero, our sharp and gradient operators collapse to zero, while the associated projection operator simplifies to the identity matrix. We thus set these operators accordingly when the face area is below a small threshold to remove possible roundoff errors. Finally, we notice that our inner product matrix computed on a single triangle reduces to a stabilized (and thus, non-singular) version of the geometric Hodge operator proposed by Auchmann and Kurz [2006].

4.5 Divergence, Curl, and Laplace-Beltrami Operator

Equipped with discrete exterior derivatives and inner products, we can now assemble discrete versions of standard calculus operators. For instance, the divergence of a discrete 1-form within a face f is computed via $\mathbf{D}_f^{\mathsf{t}}\mathbf{M}_f$, while the curl is simply $\mathbf{1}_f^{\mathsf{t}}$. We can also construct the discrete Laplace-Beltrami operator per face f as a matrix $\mathbf{L}_f$ of size $n_f \times n_f$ of the form:

$$\boxed{\mathbf{L}_f = \mathbf{D}_f^{\mathsf{t}}\mathbf{M}_f\mathbf{D}_f.} \tag{14}$$

Based on Lemma 9, it is straightforward to show that our operator is symmetric, scale-invariant, positive semi-definite, and linear-precise in planar domains. Moreover, we can use this matrix to compute, per face, the Dirichlet energy of a discrete function $\boldsymbol{\phi}_f$ via

$$E_D(\boldsymbol{\phi}_f) = a_f\|\mathbf{G}_f\boldsymbol{\phi}_f\|^2 + \lambda\|\mathbf{P}_f\mathbf{D}_f\boldsymbol{\phi}_f\|^2 = \boldsymbol{\phi}_f^{\mathsf{t}}\,\mathbf{L}_f\,\boldsymbol{\phi}_f. \tag{15}$$

As expected, our Laplacian operator also reproduces the well-known cotan-Laplacian in the case of simplicial meshes.

4.6 Discussion

Our extension of the DEC machinery to polygonal meshes bears many similarities with the work of Alexa and Wardetzky [2011]. In particular, our stabilization operator $\lambda\mathbf{P}_f^{\mathsf{t}}\mathbf{P}_f$ corresponds to a choice of "admissible" matrix used in their work to make the inner product of 1-forms positive-definite (see their Section 3.3). This can be verified by noticing that our flat operator $\mathbf{V}_f$ uses the edge vectors of a 3D polygon projected to the plane. As a result, the singular vectors computed by Alexa and Wardetzky [2011] are orthogonal to the image of $\mathbf{V}_f$, which also implies that these vectors are orthogonal to the kernel of $\mathbf{P}_f$ due to Lemma 8. However, our construction of the stabilization term in $\mathbf{M}_f$ requires no SVD computations since we know the closed-form expression for the projection matrix from Eq. (12). We can even generalize our stabilization term by replacing $\lambda\mathbf{P}_f^{\mathsf{t}}\mathbf{P}_f$ by $\mathbf{P}_f^{\mathsf{t}}Z_f\mathbf{P}_f$ in Eq. (13), where Z_f is any symmetric matrix of size $n_f \times n_f$ so that $\boldsymbol{w}^{\mathsf{t}}Z_f\boldsymbol{w} > 0$ for any vector $\boldsymbol{w}$ in the kernel of $\mathbf{U}_f$, thus reproducing Theorem 1 from [Alexa and Wardetzky 2011].

In addition to the stabilization term, another difference between our approach and the work of Alexa and Wardetzky [2011] is that they compute the inner product of discrete 1-forms using edge midpoints $\mathbf{B}_f$ centered at zero, while we employ the discrete sharp operator $\mathbf{U}_f$. Consequently, their inner product matrix includes an extra term of the form $(1/a_f)(\mathbf{D}_f\mathbf{h}_f)(\mathbf{D}_f\mathbf{h}_f)^{\mathsf{t}}$, where $\mathbf{h}_f$ denotes the n_f-sized vector with the height of every vertex v incident to face f relative to the face center $\mathbf{c}_f$ in counter-clockwise order, i.e., $\mathbf{h}_f = (\mathbf{X}_f - \mathbf{1}_f\mathbf{c}_f^{\mathsf{t}})\mathbf{n}_f$. While this term is zero for planar polygons, it is in the image of $\mathbf{P}_f$ for non-flat polygons and, therefore, it is already accounted for by the stabilization term in $\mathbf{M}_f$.

Lemma 10. *For any non-planar face f, one has $\mathbf{P}_f\mathbf{D}_f\mathbf{h}_f = \mathbf{D}_f\mathbf{h}_f$.*

Proof: We first show that $\mathbf{D}_f\mathbf{h}_f \in \operatorname{Ker}\mathbf{U}_f$:

$$\mathbf{U}_f\mathbf{D}_f\mathbf{h}_f = (1/a_f)[\mathbf{n}_f]\mathbf{B}^{\mathsf{t}}\mathbf{E}_f\mathbf{n}_f = (\mathbf{I} - \mathbf{n}_f\mathbf{n}_f^{\mathsf{t}})\mathbf{n}_f = 0,$$

where we use the fact that $\mathbf{D}_f\mathbf{h}_f = \mathbf{E}_f\mathbf{n}_f$ and $\mathbf{B}_f^{\mathsf{t}}\mathbf{E}_f = -[\mathbf{a}_f]$. Using Lemma 7 completes the proof. □

Finally, it is worth pointing out that our operators given in Eqs. (10-14) can be trivially extended to subdivision surfaces. By adopting the so-called Subdivision Exterior Calculus (SEC) [de Goes et al. 2016b] framework, we can exploit the refinability of subdivision basis functions and convert computations from the polygonal to the subdivision setting seamlessly.

5 OPERATORS ON DIRECTIONAL FIELDS

While DEC encodes vector fields as 1-forms, processing tangent vectors and, more generally, directional fields sampled at vertices of discrete surfaces has been increasingly common in geometry processing. For this purpose, we now formulate discrete differential operators of vector fields on polygonal meshes. We assume that a normal vector $\mathbf{n}_v$ is assigned to each mesh vertex v using, e.g., the normalized sum of the polygonal vector areas $\mathbf{a}_f$ incident to v. We then discretize tangent vectors as a 2D vector $\boldsymbol{u}_v$ per vertex v expressed in an arbitrary, but fixed local frame $\mathbf{T}_v \in \mathbb{R}^{3\times 2}$ orthogonal to $\mathbf{n}_v$, i.e., $\mathbf{T}_v^{\mathsf{t}}\mathbf{T}_v = \mathbf{I}$ and $\mathbf{n}_v^{\mathsf{t}}\mathbf{T}_v = 0$. Similarly, we select an arbitrary 3×2 matrix $\mathbf{T}_f$ for each polygonal face f defining an orthonormal tangent frame perpendicular to $\mathbf{n}_f$. Akin to the way we assembled

Fig. 5. **Robustness to polygon types:** From user-specified strokes indicating constrained directions (left), we compute a tangent direction field by minimizing the vector Dirichlet energy under these constraints on different polygonal meshes. The resulting tangent fields (right) demonstrate the robustness of our discrete operators to mesh discretization. For better illustration, insets show an interpolation of the vertex-based directions generated via vertex-to-face parallel-transport and generalized barycentric coordinates.

the face vector $\boldsymbol{\phi}_f$ for a scalar function ϕ, we gather the 2D vectors of all vertices adjacent to f and concatenate them into a column vector $\boldsymbol{u}_f$ of size $2n_f$, that is, $\boldsymbol{u}_f = \left[\boldsymbol{u}_{v_1}^{\mathrm{t}} \dots \boldsymbol{u}_{v_{n_f}}^{\mathrm{t}}\right]^{\mathrm{t}}$.

5.1 Discrete Levi-Civita Connection

Since tangent vectors are expressed in their own tangent frames, one first needs to parallel-transport them to their shared faces before evaluating differential operators. To this end, we compute a 3×3 rotation matrix $\boldsymbol{Q}_f^v$ for each vertex v incident to a polygonal face f that aligns the vertex normal $\mathbf{n}_v$ to the face normal $\mathbf{n}_f$ with the smallest rotation. The discrete connection from v to f is then defined as a 2×2 rotation matrix of the form

$$\boxed{\boldsymbol{R}_f^v = \mathbf{T}_f^{\mathrm{t}} \boldsymbol{Q}_f^v \mathbf{T}_v.} \tag{16}$$

These 2D rotations correspond to a discrete equivalent to the Levi-Civita connection induced by the metric of the surface inherited from the 3D Euclidean space. Other connections can be discretized as well, if necessary, similar to [Crane et al. 2010; Liu et al. 2016], but we stick to the usual Levi-Civita connection in our derivations.

5.2 First-order Derivatives

We now leverage our gradient operator from §3 to discretize the covariant derivative of vector fields per face f. Mimicking the continuous realm, we first make use of the discrete connection to parallel-transport the vectors $\boldsymbol{u}_f$ from the vertices incident to f to the local coordinate system on f, and then compute the discrete gradient of each coordinate of these transported vectors. The resulting discrete covariant derivative $\mathbf{G}_f^{\nabla}$ produces a linear operator that maps vertex-based vectors restricted to f to a 2×2 matrix expressed relative to the local tangent space on f, which we write out as

$$\mathbf{G}_f^{\nabla} \boldsymbol{u}_f = \sum_{v \in f} \left(\mathbf{T}_f^{\mathrm{t}} \mathbf{g}_f^v\right) \left(\boldsymbol{R}_f^v \boldsymbol{u}_v\right)^{\mathrm{t}}. \tag{17}$$

Note that each gradient vector $\mathbf{g}_f^v$ is tangent to f (Eq. (8)), hence multiplying it by $\mathbf{T}_f^{\mathrm{t}}$ simply results in a local change of coordinates.

Other first-order derivatives of vector fields are easily derived using the geometric decomposition of the covariant derivative matrix provided in [Liu et al. 2016]. For instance, the trace of the resulting matrix $\mathbf{G}_f^{\nabla} \boldsymbol{u}_f$ on face f corresponds to the *divergence* of $\boldsymbol{u}_f$, while the skew-symmetric part of $\mathbf{G}_f^{\nabla} \boldsymbol{u}_f$ measures the *curl* of $\boldsymbol{u}_f$. Similarly, the *Killing operator* of $\boldsymbol{u}_f$ can be computed as the symmetric part of $\mathbf{G}_f^{\nabla} \boldsymbol{u}_f$, while the *directional derivative* of $\boldsymbol{u}_f$ along a tangent vector $\boldsymbol{s}$ is the matrix-vector product $\boldsymbol{s}^{\mathrm{t}}(\mathbf{G}_f^{\nabla} \boldsymbol{u}_f)$. Additional derivatives, such as the holomorphic derivative or the complex conjugate (Cauchy-Riemann) derivative, are also expressible as combinations of these aforementioned terms.

To further highlight the link between the discrete covariant derivative and the gradient operator applied to vector coordinates, we form the matrix $\boldsymbol{u}_f^{\nabla}$ of size $n_f \times 2$ containing the vertex-based vectors $\boldsymbol{u}_f$ incident to the face f parallel-transported to f via $\boldsymbol{R}_f^v$, i.e., $\boldsymbol{u}_f^{\nabla} = \left[\boldsymbol{R}_f^{v_1} \boldsymbol{u}_{v_1} \dots \boldsymbol{R}_f^{v_{n_f}} \boldsymbol{u}_{v_{n_f}}\right]^{\mathrm{t}}$. The discrete covariant derivative on face f from Eq. (17) can now be more compactly rewritten as

$$\boxed{\mathbf{G}_f^{\nabla} \boldsymbol{u}_f = \mathbf{T}_f^{\mathrm{t}} \mathbf{G}_f \boldsymbol{u}_f^{\nabla}.} \tag{18}$$

This construction also reveals that our discrete covariant derivative inherits both the linear precision of the gradient operator as well as its rank-deficiency. To penalize the null-space of $\mathbf{G}_f^{\nabla}$ on the polygonal face f, we extend the projection operator $\mathbf{P}_f$ from §4.3 to act on vector fields instead of discrete 1-forms. More concretely, we define the *covariant projection* operator $\mathbf{P}_f^{\nabla}$ per face f as a linear mapping that parallel-transports the vertex-based vectors to the tangent space of face f and returns a matrix of size $n_f \times 2$ with the 1-form projection of the exterior derivative of the vector coordinates in f, which we write out as

$$\boxed{\mathbf{P}_f^{\nabla} \boldsymbol{u}_f = \mathbf{P}_f \mathbf{D}_f \boldsymbol{u}_f^{\nabla}.} \tag{19}$$

5.3 Vector Laplacian

Using the discrete covariant derivative and the connection projection, we can now extend the Dirichlet energy defined in Eq. (15) from scalar functions to vector fields and construct a vector Laplacian matrix $\mathbf{L}_f^{\nabla}$ of size $2n_f \times 2n_f$ restricted to face f:

$$E_D(\boldsymbol{u}_f) = a_f \|\mathbf{G}_f^{\nabla} \boldsymbol{u}_f\|_{\mathrm{F}}^2 + \lambda \|\mathbf{P}_f^{\nabla} \boldsymbol{u}_f\|_{\mathrm{F}}^2 \boldsymbol{u}_f^{\mathrm{t}} \mathbf{L}_f^{\nabla} \boldsymbol{u}_f, \tag{20}$$

where the first term computes the squared Frobenius norm of the our discrete covariant derivative, while the second term accounts

for the stabilization for each coordinate of the vertex-based vectors transported to the face f. As in the scalar case, the vector Laplacian matrix is symmetric, scale-invariant, positive semi-definite, and linear-precise in planar meshes.

5.4 Discussion

Once a vertex-to-face connection is established, our discrete gradient operator allows for simple and intuitive expressions of the covariant derivative and all the other typical vector processing operators. We note that Sharp et al. [2019] also proposed a discrete vector Laplacian for polygonal meshes by multiplying coefficients of the scalar Laplacian matrix from [Alexa and Wardetzky 2011] with rotations encoding a vertex-to-vertex connection. Our approach offers a similar assembly of the vector Laplacian matrix per polygonal face, but using vertex-to-face rotations instead. Importantly, since our discretization is compatible with a discrete covariant derivative, we can easily penalize various combinations of the first-order derivatives as detailed in §5.2 and construct alternative quadratic energies for vector fields, e.g., the discrete Hodge-Laplace operator or the squared Frobenius norm of the Killing operator [Solomon et al. 2011]. Finally, we point out that our discrete operators are easy to extend to n-vector fields [Vaxman et al. 2017]. As described in [Knöppel et al. 2013], processing a n-vector field is achieved by converting the input field to representative vectors per vertex, replacing the 2D rotations of the discrete connection by their n-th power, and after computations, reverting the resulting vertex-based representative vectors back to a n-vector field.

6 ADDITIONAL OPERATORS

So far we have covered a series of discrete differential operators acting on discrete forms and tangent vectors. Building upon these previous results, we discuss next a few additional discrete operators that are particularly relevant to geometry processing.

Fig. 6. **Avoiding spurious deformations:** Given a polygonal mesh with face-based Jacobian matrices set to $\boldsymbol{J}_f=\mathbf{I}$, the minimization of Eq. (21) reproduces the input rest shape (left) exactly as expected. In contrast, ignoring the rest points from the stabilization term (i.e., using only $\|\mathbf{P}_f\mathbf{D}_f\mathbf{Y}_f\|^2$) penalizes the input vertex heights $\mathbf{h}_f$, thus resulting in more planar polygons (center). Disabling the stabilization term altogether (i.e., $\lambda=0$) introduces spurious deformations (right). Pseudo-colors encoding the signed heights of vertices per face are used for visual comparison.

Fig. 7. **Parameterization of polygonal meshes:** Be it with the spectral approach of [Mullen et al. 2008] using a discrete conformal energy (left: car model with quads), the Symmetric Dirichlet energy [Smith and Schaefer 2015] (center: spot model with quads), or the As-Rigid-As-Possible energy [Liu et al. 2008] (right: buddha model with hex tiles), our discrete deformation gradient allows the parameterization of arbitrary polygonal meshes for a wide variety of non-linear forms of distortion.

6.1 Deformation Gradient and Distortion Energies

Be it for deformation or parameterization, computing the deformation gradient from a rest shape of a discrete surface $\mathcal{M}$ to a different pose is a common task in geometry processing. In the case of polygonal meshes, our discrete gradient operator from Eq. (7) can be directly used to derive a deformation gradient. After forming the gradient matrix $\mathbf{G}_f$ of a polygonal face f using its rest shape embedding $\mathbf{X}_f$, we compute the deformation gradient on a target shape embedding $\mathbf{Y}_f$ as a 3×3 matrix of the form $\mathbf{G}_f\mathbf{Y}_f$, which simply applies the gradient to each coordinate separately.

The total distortion caused by the deformation gradient is often quantified as a single scalar value through the definition of an application-specific distortion energy $E(\cdot)$. For instance, Poisson editing techniques [Yu et al. 2004; Sumner and Popoviç 2004] construct an embedding that minimizes the sum of all squared Frobenius norms between the deformation gradient $\mathbf{G}_f\mathbf{Y}_f$ and a prescribed 3×3 Jacobian matrix $\mathbf{J}_f$ per face f. For polygonal meshes where faces may be non-planar, special care must be taken to handle the various heights of a face as well as to avoid spurious deformations modes (see Figure 6). This can be achieved by extending the Poisson distortion energy with a stabilization term, yielding

$$E_{3\mathrm{D}}(\mathbf{Y}_f)=a_f\|\mathbf{G}_f(\mathbf{Y}_f-\mathbf{X}_f\mathbf{J}_f)\|_{\mathrm{F}}^2+\lambda\|\mathbf{P}_f\mathbf{D}_f(\mathbf{Y}_f-\mathbf{X}_f\mathbf{J}_f)\|_{\mathrm{F}}^2, \quad (21)$$

where the first term measures the linear in-plane deformation, while the second term penalizes normal and non-affine variations of the polygon. The energy derivative w.r.t. the unknown points $\mathbf{Y}_f$ is then simply expressed per face f as $\mathbf{L}_f\mathbf{Y}_f=\mathbf{L}_f\mathbf{X}_f\mathbf{J}_f$, where the stabilization is already included in the polygonal Laplacian matrix.

For mesh parameterization, one aims instead to find a 2D embedding $\mathbf{Y}$ that minimizes a given distortion energy so as to flatten the original 3D embedding $\mathbf{X}$ with the least amount of face distortion. Here again, we can employ our discrete operators to adapt existing parametrization methods to polygonal meshes with the addition of

Fig. 8. **Shape operator:** With vertex normals set to the normalized sum of polygonal vector areas, our face-based shape operator approximates ground-truth curvature values om various tessellations of a torus. We show the L_2 error plots in log-log scale for the Gaussian (left) and mean curvature (center) values computed using the determinant and the trace of our discrete shape operator, respectively. While we symmetrize our shape operator to better reproduce its continuous properties, the Frobenius norm of the skew-symmetric part of the gradient of the normals is shown to converge to zero as expected (right). We also note that quadrangulated surfaces used in this test produced symmetric shape operators due to radial symmetry of the mesh, thus explaining the near-zero residual represented by the orange line in the rightmost plot.

a stabilization term, resulting in a general energy of the form

$$E_{2\mathrm{D}}(\mathbf{Y}_f) = a_f \Psi(\mathbf{G}_f \mathbf{Y}_f) + \lambda \|\mathbf{P}_f \mathbf{D}_f \mathbf{Y}_f\|_{\mathrm{F}}^2, \tag{22}$$

where $\Psi(\cdot)$ encodes any of the isotropic distortion energies typically used in parametrization, including As-Rigid-As-Possible [Liu et al. 2008] and Symmetric Dirichlet [Smith and Schaefer 2015]. In particular, our formulation trivially accommodates efficient optimization methods such as the projected Newton solver with analytic eigensystems described in Smith et al. [2019].

6.2 Shape Operator

We can also use our results to discretize the shape operator on polygonal meshes. In the smooth setting, the shape operator is equal to the gradient of the surface normal field, that is, $\mathbf{S} = \nabla \mathbf{n}$. Following this definition and using $\mathbf{N}_f$ to denote the matrix of size $n_f \times 3$ storing the vertex normals $\mathbf{n}_v$ incident at f row-wise, we construct the discrete shape operator as a 2×2 matrix $\mathbf{S}_f$ per face f that computes the gradient of each coordinate of $\mathbf{N}_f$, and then takes the symmetrized tangential part of the result, producing

$$\boxed{\mathbf{S}_f = \tfrac{1}{2}\, \mathbf{T}_f^{\mathrm{t}} \big(\mathbf{G}_f \mathbf{N}_f + \mathbf{N}_f^{\mathrm{t}} \mathbf{G}_f^{\mathrm{t}}\big) \mathbf{T}_f.} \tag{23}$$

Compared to existing methods on simplicial meshes [Rusinkiewicz 2004; Hildebrandt and Polthier 2011], our approach is directly valid on triangle and polygonal surfaces with no alterations.

6.3 Adjoint Operators

The discrete differential operators we derived so far offer mappings from values at vertices to faces. Through duality, we can also construct adjoint operators that act, conversely, from faces to vertices. We define an adjoint gradient operator per vertex v as a matrix that takes a scalar value per incident face and returns a vector orthogonal to the vertex normal $\mathbf{n}_v$. With the vertex area $a_v = \sum_{f \ni v} a_f / n_f$, we compute the vertex-based gradient operator via

$$\tilde{\mathbf{G}}_v\, \boldsymbol{\phi}_v = \frac{1}{a_v} \sum_{f \ni v} a_f \big(\boldsymbol{Q}_f^v\big)^{\mathrm{t}} \mathbf{g}_f^v\, \phi(\mathbf{c}_f), \tag{24}$$

where the rotation $\big(\boldsymbol{Q}_f^v\big)^{\mathrm{t}}$ is used to parallel-transport the 3D vector $\mathbf{g}_f^v$ from face to vertex. Likewise, an adjoint covariant derivative can be defined for face-based vector fields through

$$\tilde{\mathbf{G}}_v^{\nabla}\, \boldsymbol{u}_v = \frac{1}{a_v} \sum_{f \ni v} a_f \big((\boldsymbol{R}_f^v)^{\mathrm{t}}\, \mathbf{T}_f^{\mathrm{t}}\, \mathbf{g}_f^v\big)\, \big((\boldsymbol{R}_f^v)^{\mathrm{t}}\, \boldsymbol{u}_f\big)^{\mathrm{t}}. \tag{25}$$

With these expressions, we can also formulate other vertex-based discrete operators through derivations similar to our previous sections. For instance, an adjoint shape operator per vertex v can be constructed by first stacking the normals $\mathbf{n}_f$ of the faces adjacent to v row-wise into a matrix $\mathbf{N}_v$, and then computing

$$\tilde{\mathbf{S}}_v = \tfrac{1}{2}\, \mathbf{T}_v^{\mathrm{t}} \big(\tilde{\mathbf{G}}_v\, \mathbf{N}_v + \mathbf{N}_v^{\mathrm{t}}\, \tilde{\mathbf{G}}_v^{\mathrm{t}}\big) \mathbf{T}_v. \tag{26}$$

7 RESULTS AND APPLICATIONS

In this section, we present a series of numerical tests and geometry processing examples using our discrete differential operators that demonstrate the accuracy and versatility of our discretization.

7.1 Numerical Tests

In order to provide insight on the accuracy of our operators, we compare our numerical results against ground-truth analytical expressions and analyze the convergence of residual errors on polygonal meshes with increasing resolution. Figure 3(top) shows the error plot for our discrete gradient operator computed on various tessellations of a square domain as a function of the mesh resolution measured as the inverse of the mean edge length. In this example, we evaluate the scalar function $\sin(4\pi x)\cos(4\pi y)$ on mesh vertices and compare the L_2 norm of the residual between our numerical gradient vector and the analytical gradient at the barycenter of each polygonal face. These error plots include a triangle mesh refined through Loop subdivision, a quad mesh refined via Catmull-Clark subdivision, a Voronoi diagram for which finer discretizations are obtained by adding more vertices and running a few CVT relaxation steps [Du et al. 1999], and a mesh with concave cells produced from a quad mesh through edge splitting and vertex sliding to create zigzagged edges. The accuracy of our discrete covariant derivative

is demonstrated in Figure 3(bottom) for different polygonal meshes of a unit sphere, with tessellations mirroring our previous setup. For this test, we assign the tangent vector $\sqrt{1-z^2}\,[-y, x, 0]$ to each mesh vertex, compute the covariant derivative per face, and measure the Frobenius norm of the residual with the analytical covariant derivative. Both of these plots show at least linear convergence for all the different tessellations. We also note that the L_∞ errors of these numerical tests exhibit a similar behavior.

In Figure 8(left and middle), we display the L_2 errors of the Gaussian and mean curvatures computed via the determinant and trace of our face-based shape operators, compared with the analytical values on a torus. In this experiment, we discretized the torus with polygonal meshes formed by triangles, hexagons, regular and sheared quadrilaterals. We also plot the magnitude of the skew-symmetric part of our gradient applied to the vertex normals in Figure 8(right), confirming that the discrete gradient of the normal field becomes symmetric under refinement. Additionally, Figure 2 compares the accuracy of the mean curvature estimation on a torus-shaped polygonal mesh with the accuracy obtained after the polygons are triangulated, exemplifying the numerical bias induced by triangulating polygons. In this case, we approximate the mean curvature at mesh vertices by first multiplying the discrete Laplacian matrix with the vertex positions, which returns an integrated mean curvature vector per vertex, and then computing the dot product of these vectors with the vertex normals, divided by twice the vertex area. On the coarsest resolution, the L_∞ residual of the vertex-based mean curvatures compared to the ground-truth values is of 0.2 on the polygonal torus versus 2.3 and 6.1 on the triangulated meshes.

To analyze the influence of the stabilization parameter λ on the accuracy of our discrete Laplacian operator, we consider approximations of a scalar Poisson equation computed on various discretizations of a unit sphere. In our tests, we set the weak form of the right-hand side of the Poisson equation to a spherical harmonic function sampled at mesh vertices and multiplied by their vertex areas. As spherical harmonics are by definition eigenvectors of the Laplacian, we compare the numerical solution of the discrete Poisson equation to the input values divided by its exact eigenvalue. Figure 4 includes a representative example of our tests showing the L_2 errors for the solution of the discrete Poisson equation with various values of λ. These plots indicate a quadratic convergence rate, with $\lambda = 1$ consistently producing the most accurate results. We thus used $\lambda = 1$ systematically for all our geometry processing applications. For reference, we also compute the residual using the discrete Laplacian from [Alexa and Wardetzky 2011] (with their recommended value of $\lambda = 2$) as well as from [Bunge et al. 2020], and observed results of the same order of accuracy as ours.

Finally, we test the robustness of our discrete operators on directional fields in Figure 5. Starting from a sparse set of user-specified directions, we solve the method of [Knöppel et al. 2013] using our operators for polygonal meshes in order to generate a smooth direction field that best aligns to the constrained directions. Figure 5 shows qualitatively similar results produced on surface meshes with various polygonal discretizations. We also compare close-ups of these direction fields sampled on a dense set of scattered points, which were computed by parallel-transporting the vertex-based vectors to their adjacent faces and interpolating them to the sample points using generalized barycentric coordinates.

Fig. 9. **Shape editing:** From a neutral face **B** and two other embeddings of this polygonal mesh **A** and **C** (top), Poisson editing based on our polygonal discretization can blend parts of these faces following user-prescribed falloffs (indicated in red) in order to create new shape embeddings (bottom).

7.2 Applications

We now use our discrete differential operators on representative geometry processing applications. Most of these tasks involve the solve of linear systems, for which we use [Chen et al. 2008]. Timings for these linear solves on polygonal meshes are nearly identical to their simplicial counterparts due to the simplicity of our operators.

Shape Editing. In Figure 9, we show examples of partial blends of 3D facial poses using our polygonal version of the Poisson editing technique described in §6.1. From a quad mesh with 5k vertices, we first compute the deformation gradients for every polygonal face from the rest embedding (indicated by the label **B**) to the pose shapes (**A** and **C**). We then generate target Jacobian matrices per polygon by interpolating these deformation gradients weighted by user-specified falloff functions (indicated in red). Lastly, we minimize the quadratic energy in Eq.(21), resulting in a composite shape.

Parametrization. Our discrete operators are also well suited to extend most parameterizations methods to polygonal meshes. Similar to [Alexa and Wardetzky 2011], we show in Figure 7(left) a spectral conformal parametrization [Mullen et al. 2008], computed with our

Fig. 10. **Polygonal versus triangulated meshes:** The spectral conformal parametrization of [Mullen et al. 2008] computed on the blub model using our discrete operators on polygonal faces (left) produces improved results compared to two different triangulations of the input mesh (center and right). Pseudo-colors indicate the quasi-conformal error computed per mesh face, totalizing an area-weighted mean error of (1.0014, 1.0063, 1.0085) and a max error of (1.14, 1.18, 1.47), respectively.

discrete Laplacian and the gradient of the polygonal areas on boundary vertices, for a quadrangulated mesh with 5k vertices. Figure 10 compares the spectral conformal parametrization of a quad mesh with 7k vertices obtained with our discrete operators versus the original method of [Mullen et al. 2008] on two possible triangulations of the quad mesh. In this example, we also display the quasi-conformal error per face computed via the singular values of the deformation gradient, demonstrating that our polygonal approach avoids having to choose a triangulation while ensuring high quality results. As mentioned in §6.1, we can also generate mesh parameterizations by optimizing non-linear distortion energies via the efficient projected Newton solver of Smith et al. [2019]. For instance, Figure 7(middle) shows the result of a quadrangulated mesh with 3k vertices flattened by minimizing the Symmetric Dirichlet energy [Smith and Schaefer 2015], which took 26 iterations to converge. We point out here that the flip-avoiding line search introduced in [Smith and Schaefer 2015] can be used directly for polygonal faces by noticing that the polygonal vector area in 2D is still a quadratic equation in terms of the 2D vertex positions of the parameterization. Finally, Figure 7(right) shows a mesh formed by 60k vertices with mostly hexagon polygons, parametrized using the As-Rigid-As-Possible (ARAP) energy [Liu et al. 2008], which took only two iterations to converge (zoom in on the parameterization to visualize individual polygons). Note that triangulating the polygons before computing the mesh parameterization (either through adding diagonal edges or inserting virtual nodes) renders the evaluation of the non-linear distortion energy and its derivatives more costly as it now involves many more mesh faces. For instance, the ARAP parametrization of a triangulated version of the model in Figure 7(right) leads to an increase in the number of iterations by a factor 5 and of the total computational time by 60%. In all these tests, input meshes were rescaled to have unit surface area, their initial parameterizations were generated using a Tutte embedding onto a unit disk, and their optimizations were terminated when the energy gradient L_∞ norm reached below 10^{-4}.

Suggestive Contours. The availability of a shape operator for polygonal meshes allows us to extend existing shape analysis techniques such as suggestive contours [DeCarlo et al. 2004]. For each face and vertex of an input mesh, we compute a 3D vector $\boldsymbol{u}$ from the camera origin $\boldsymbol{o}$ to the element location ($\mathbf{x}_v$ for vertices and $\mathbf{c}_f$ for faces), which are then projected onto their respective local tangent plane and normalized. Next, we construct our adjoint shape operator $\widetilde{\mathbf{S}}_v$ (Eq. (26)) to estimate the radial curvature along $\boldsymbol{u}_v$ via $\kappa_v = \boldsymbol{u}_v^{\mathsf{t}}\, \widetilde{\mathbf{S}}_v\, \boldsymbol{u}_v$. Suggestive contours are then extracted by tracing the zero iso-contours of κ_v. We further trim these contour curves by removing segments whose directional gradient $\boldsymbol{u}_f^{\mathsf{t}}(\mathbf{G}_f \kappa_v)$ is smaller than a user-specified threshold. We complement these suggestive contours by tracing mesh silhouettes corresponding to the zero level-sets of $\mathbf{n}_v^{\mathsf{t}}(\mathbf{x}_v - \boldsymbol{o})$. Figure 11 shows an example of suggestive contours generated by our method on a quadrangulated mesh with 32k vertices. For comparison purposes, we also include our result for the triangulated cow model used in [DeCarlo et al. 2004].

Texture Synthesis. Vector field design is another common task in geometry processing that can be extended to polygonal meshes. As Figure 12 illustrates, given a few user-drawn strokes on a polygonal mesh, we can compute the smoothest direction field aligned to these strokes using the method of Knöppel et al. [2013] where the vector Laplacian discretization is replaced by our polygonal operator. We then use the resulting direction field to synthesize a texture pattern from a given exemplar using the method of [Heitz and Neyret 2018], which is rendered using a displacement shader. In this example, the input mesh has 4k vertices and the direction field generation was interactive, taking a few milliseconds per solve.

Fig. 11. **Suggestive contours:** With our shape operator, we can compute any curvature (left, mean curvature produced by the trace of the shape operator) and even extend the suggestive contour approach of DeCarlo et al. [2004] (right) on polygonal meshes. In addition to the quad mesh with 32k vertices (top), we tested our discretization on a triangle mesh with 48k vertices (bottom) provided by DeCarlo et al. [2004] for comparison purposes.

Grooming. Our final application is shown in Figure 1 and tackles grooming. Based on a quadrangulated input surface, a user selects a few root locations to attach guiding polylines (displayed in blue, offset from the surface for clarity) with the same number of segments each. For every point along these polylines, we compute the vector connecting them to their respective roots and decompose these vectors into a tangent component orthogonal to the surface normal and a height scalar. A scalar Poisson equation using our polygonal Laplacian is then solved to interpolate the heights from the guiding polylines to all the mesh vertices. Similarly, a vector Poisson equation is used to interpolate the tangent guiding vectors throughout the surface. We then combine these tangent vectors and heights back into 3D vectors and reconstruct a polyline per mesh vertex to mimic hair growth. The same strategy can also be employed to place and deform feathers on the wings, as illustrated at the bottom of Figure 1. Since our discrete operators on polygonal meshes are basically as computationally efficient as traditional operators on triangle meshes, a grooming example like this can be easily designed in real-time on meshes with 10k vertices.

Discussion. Although we have highlighted only a few applications in this paper, our polygon-based differential operators can be applied seamlessly to a wide range of geometry processing algorithms originally designed for simplicial meshes. They can also help simplify computations on polygonal meshes agnostic to any underlying triangulation and without the need for special numerical treatment. For instance, the gradient normalization step in the geodesics method of Crane et al. [2013b] (see inset) can be trivially computed on polygonal meshes as we still have one single gradient vector per face. Note that the short-time diffusion used by the heat method requires a time-step proportional to the maximum length over all polygon diagonals, instead of the mean edge length, so that any pair of vertices sharing a face can be reached as already mentioned in [de Goes et al. 2016b].

8 CONCLUSION

We have presented a family of discrete differential operators valid on general polygonal meshes. Our approach provides polygonal operators that are numerically stable and simple to implement, while mimicking key structural properties of their smooth counterpart. We have also showed theoretical developments that extend prior 2D polygonal discretizations to surface meshes, and adapt triangle-based methods to arbitrary polygons with only minor modifications. We demonstrated the numerical accuracy of our discrete operators through convergence tests, and proved their practicality by putting them to use in various geometry processing tasks.

We believe this work opens up different possibilities for future work. First, we are interested in establishing a principled way to select the parameter λ, or even the matrix $\boldsymbol{Z}_f$, required by our inner product matrix $\mathbf{M}_f$ per face f in order to further increase the accuracy of our discrete operators. Since our projection operator extends the stabilization term found in VEM to non-planar polygons, it would also be interesting to revisit the Virtual Element Method and, in particular, its extensions to high-order approximations based on our formulation. From a functional analysis standpoint, we would like to derive precise convergence estimates for our construction using, e.g., the geometric analysis for surface finite element methods proposed in [Holst and Stern 2012]. Finally, extending VEM to arbitrary polyhedral meshes is an exciting direction, since current VEM methods are limited to 3D cells with only planar polygonal faces [Gain et al. 2014].

ACKNOWLEDGMENTS

We thank Athena Xenakis, Laura Hainke, and Colin Thompson for help with Figures 1 and 12, and Mark Meyer for proof-reading. Meshes are courtesy of Bay Raitt (big guy), Keenan Crane (blub, spot), David Bommes (buddha), Mario Botsch (loki), Stanford 3D Scanning Repository (bunny), and Viewpoint Animation Engineering / Sun Microsystems (cow). All other meshes and images are copyrighted by Disney/Pixar. Finally, MD gratefully acknowledges the hospitality of ShanghaiTech University during his sabbatical.

REFERENCES

Ralph Abraham, Jerrold E. Marsden, and Tudor Ratiu. 1988. *Manifolds, Tensor Analysis, and Applications.* Applied Mathematical Sciences, Vol. 75. Springer Verlag.

Marc Alexa and Max Wardetzky. 2011. Discrete Laplacians on General Polygonal Meshes. *ACM Transactions on Graphics* 30, 4 (2011), Article 102.

Bernhard Auchmann and Stefan Kurz. 2006. A geometrically defined discrete Hodge operator on simplicial cells. *IEEE Transactions on Magnetics* 42, 4 (2006), 643–646.

Omri Azencot, Maks Ovsjanikov, Frédéric Chazal, and Mirela Ben-Chen. 2015. Discrete Derivatives of Vector Fields on Surfaces – An Operator Approach. *ACM Transactions on Graphics* 34, 3 (2015), Article 29.

Lourenço Beirão da Veiga, Franco Brezzi, Andrea Cangiani, Gianmarco Manzini, Luisa Donatella Marini, and Alessandro Russo. 2013. Basic principles of Virtual Element Methods. *Mathematical Models and Methods in Applied Sciences* 23, 1 (2013), 199–214.

Lourenço Beirão da Veiga, Carlo Lovadina, and Alessandro Russo. 2017. Stability Analysis for the Virtual Element Method. *Mathematical Models and Methods in Applied Sciences* 27, 13 (2017), 2557–2594.

Franco Brezzi, Konstantin Lipnikov, and Valeria Simoncini. 2005. A Family Of Mimetic Finite Difference Methods On Polygonal And Polyhedral Meshes. *Mathematical Models and Methods in Applied Sciences* 15, 10 (2005), 1533–1551.

Astrid Bunge, Philipp Herholz, Misha Kazhdan, and Mario Botsch. 2020. Polygon Laplacian Made Simple. *Computer Graphics Forum* 39, 2 (2020).

Frédéric Cazals and Marc Pouget. 2005. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design* 22, 2 (2005), 121–146.

Fig. 12. **Texture synthesis:** From a few directional strokes (in red) on the quad mesh of a tree (left), we compute a smooth direction field by simply substituting the triangle-based vector Laplacian from [Knöppel et al. 2013] with our polygonal approximation (center), over which a texture is synthesized using [Heitz and Neyret 2018] based on a bark exemplar (inset). The result is rendered with displacement mapping (right).

Yanqing Chen, Timothy A. Davis, William W. Hager, and Sivasankaran Rajamanickam. 2008. Algorithm 887: CHOLMOD, Supernodal Sparse Cholesky Factorization and Update/Downdate. *ACM Trans. Math. Software* 35, 3 (2008), 14.

David Cohen-Steiner and Jean-Marie Morvan. 2003. Restricted Delaunay Triangulations and Normal Cycle. In *Symposium on Computational Geometry*. 312–321.

Dylan M. Copeland. 2009. Boundary-Element-Based Finite Element Methods for Helmholtz and Maxwell Equations on General Polyhedral Meshes. *International Journal of Mathematical and Computational Sciences* 3, 9 (2009), 661–674.

Keenan Crane, Fernando de Goes, Mathieu Desbrun, and Peter Schröder. 2013a. Digital Geometry Processing with Discrete Exterior Calculus. In *ACM SIGGRAPH 2013 Courses*. Course #7.

Keenan Crane, Mathieu Desbrun, and Peter Schröder. 2010. Trivial Connections on Discrete Surfaces. *Computer Graphics Forum* 29, 5 (2010), 1525–1533.

Keenan Crane, Clarisse Weischedel, and Max Wardetzky. 2013b. Geodesics in Heat: A New Approach to Computing Distance Based on Heat Flow. *ACM Transactions on Graphics* 32, 5 (2013), Art. 152.

Ke-Yang Dai, Gui-Rong Liu, and Thoi-Trung Nguyen. 2007. An n-sided polygonal smoothed finite element method (nSFEM) for solid mechanics. *Finite Elements in Analysis and Design* 43, 11 (2007), 847–860.

Fernando de Goes, Mathieu Desbrun, Mark Meyer, and Tony DeRose. 2016b. Subdivision Exterior Calculus for Geometry Processing. *ACM Transactions on Graphics* 35, 4 (2016), Article 133.

Fernando de Goes, Mathieu Desbrun, and Yiying Tong. 2016a. Vector Field Processing on Triangle Meshes. In *ACM SIGGRAPH Courses*. Article 27, 49 pages.

Fernando de Goes, Beibei Liu, Max Budninskiy, Yiying Tong, and Mathieu Desbrun. 2014. Discrete 2-tensor Fields on Triangulations. *Computer Graphics Forum* 33, 5 (2014), 13–24.

Doug DeCarlo, Adam Finkelstein, and Szymon Rusinkiewicz. 2004. Interactive Rendering of Suggestive Contours with Temporal Coherence. In *Symposium on Non-Photorealistic Animation and Rendering*. 135–145.

Mathieu Desbrun, Eva Kanso, and Yiying Tong. 2008. Discrete Differential Forms for Computational Modeling. In *Discrete Differential Geometry*, A.I. Bobenko et al. (Ed.). Birkhäuser Basel, Chapter 16, 287–324.

Qiang. Du, Vance. Faber, and Max. Gunzburger. 1999. Centroidal Voronoi Tessellations: Applications and Algorithms. *SIAM Rev.* 41, 4 (1999), 637–676.

Michael Floater. 2003. Mean value coordinates. *Computer Aided Geometric Design* 20, 1 (2003), 19–27.

Arun L. Gain, Cameron Talischi, and Glaucio H. Paulino. 2014. On the Virtual Element Method for three-dimensional linear elasticity problems on arbitrary polyhedral meshes. *Computer Methods in Applied Mechanics and Engineering* 282 (2014), 132–160.

Andrew Gillette, Alexander Rand, and Chandrajit Bajaj. 2016. Construction of Scalar and Vector Finite Element Families on Polygonal and Polyhedral Meshes. *Computational Methods in Applied Mathematics* 16, 4 (2016), 667–683.

Eitan Grinspun, Yotam Gingold, Jason Reisman, and Denis Zorin. 2006. Computing discrete shape operators on general meshes. *Computer Graphics Forum* 25, 3 (2006), 547–556.

Eric Heitz and Fabrice Neyret. 2018. High-Performance By-Example Noise Using a Histogram-Preserving Blending Operator. *Symposium on High-Performance Graphics* 1, 2 (2018), Article 31.

Philipp Herholz, Jan Eric Kyprianidis, and Marc Alexa. 2015. Perfect Laplacians for Polygon Meshes. *Computer Graphics Forum* 34, 5 (2015), 211–218.

Klaus Hildebrandt and Konrad Polthier. 2011. Generalized shape operators on polyhedral surfaces. *Computer Aided Geometric Design* 28, 5 (2011), 321–343.

Anil N. Hirani. 2003. *Discrete Exterior Calculus*. Ph.D. Dissertation. Caltech.

Michael Holst and Ari Stern. 2012. Geometric Variational Crimes: Hilbert Complexes, Finite Element Exterior Calculus, and Problems on Hypersurfaces. *Foundations of Computational Mathematics* 12, 3 (2012), 263–293.

Evangelos Kalogerakis, Patricio Simari, Derek Nowrouzezahrai, and Karan Singh. 2007. Robust Statistical Estimation of Curvature on Discretized Surfaces. In *Symposium on Geometry Processing*. 13–22.

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2013. Globally Optimal Direction Fields. *ACM Transactions on Graphics* 32, 4 (2013), Article 59.

Konstantin Lipnikov, Gianmarco Manzini, and Mikhail Shashkov. 2014. Mimetic Finite Difference Method. *J. Comput. Phys.* 257 (2014), 1163–1227.

Beibei Liu, Yiying Tong, Fernando de Goes, and Mathieu Desbrun. 2016. Discrete Connection and Covariant Derivative for Vector Field Analysis and Design. *ACM Transactions on Graphics* 35, 3 (2016), Article 23.

Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J. Gortler. 2008. A Local/Global Approach to Mesh Parameterization. *Computer Graphics Forum* 27, 5 (2008), 1495–1504.

Michael Mengolini, Matías F. Benedetto, and Alejandro M. Aragón. 2019. An engineering perspective to the virtual element method and its interplay with the standard finite element method. *Computer Methods in Applied Mechanics and Engineering* 350 (2019), 995–1023.

Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. 2003. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In *Visualization and Mathematics III*. 35–57.

Patrick Mullen, Yiying Tong, Pierre Alliez, and Mathieu Desbrun. 2008. Spectral Conformal Parameterization. *Computer Graphics Forum* 27, 5 (2008), 1487–1494.

Konrad Polthier and Eike Preuss. 2003. Identifying Vector Field Singularities Using a Discrete Hodge Decomposition. *Visualization and Mathematics* 3 (2003).

Pierre-Arnaud Raviart and Jean-Marie Thomas. 1977. A Mixed Finite Element Method for Second Order Elliptic Problems. 66 (1977), 292–315.

Sergej Rjasanow and Steffen Weißer. 2012. Higher Order BEM-Based FEM on Polygonal Meshes. *SIAM J. Numer. Anal.* 50, 5 (2012), 2357–2378.

Szymon Rusinkiewicz. 2004. Estimating Curvatures and Their Derivatives on Triangle Meshes. In *Symposium on 3D Data Processing, Visualization, and Transmission*. 486–493.

Nicholas Sharp, Yousuf Soliman, and Keenan Crane. 2019. The Vector Heat Method. *ACM Transactions on Graphics* 38, 3 (2019), Article 24.

Breannan Smith, Fernando de Goes, and Theodore Kim. 2019. Analytic Eigensystems for Isotropic Distortion Energies. *ACM Transactions on Graphics* 38, 1, Article 3 (2019), 15 pages.

Jason Smith and Scott Schaefer. 2015. Bijective Parameterization with Free Boundaries. *ACM Transactions on Graphics* 34, 4 (2015).

Justin Solomon, Mirela Ben-Chen, Adrian Butscher, and Leonidas Guibas. 2011. As-Killing-As-Possible Vector Fields for Planar Deformation. *Computer Graphics Forum* 30, 5 (2011), 1543–1552.

Natarajan Sukumar and Alireza Tabarraei. 2004. Conforming polygonal finite elements. *Internat. J. Numer. Methods Engrg.* 61, 12 (2004), 2045–2066.

John M. Sullivan. 2008. *Curvatures of Smooth and Discrete Surfaces*. Birkhäuser Basel, Chapter 9, 175–188.

Robert W. Sumner and Jovan Popoviç. 2004. Deformation Transfer for Triangle Meshes. *ACM Transactions on Graphics* 23, 3 (2004).

Xu-Hai Tang, Sheng-Chuan Wu, Chao Zheng, and Jian-Hai Zhang. 2009. A novel virtual node method for polygonal elements. *Appl. Math. Mech.* 30 (2009), Article 1233.

Gabriel Taubin. 1995. Estimating the tensor of curvature of a surface from a polyhedral approximation. In *IEEE International Conference on Computer Vision*. 902–907.

Yiying Tong, Santiago Lombeyda, Anil N. Hirani, and Mathieu Desbrun. 2003. Discrete Multiscale Vector Field Decomposition. *ACM Transactions on Graphics* 22, 3 (2003), 445–452.

Amir Vaxman, Marcel Campen, Olga Diamanti, David Bommes, Klaus Hildebrandt, Mirela Ben-Chen Technion, and Daniele Panozzo. 2017. Directional Field Synthesis, Design, and Processing. In *ACM SIGGRAPH Courses*. Article 12, 30 pages.

Eugene L. Wachspress. 1975. *A Rational Finite Element Basis*. Mathematics in Science and Engineering, Vol. 114. Academic Press.

Max Wardetzky. 2006. *Discrete Differential Operators on Polyhedral Surfaces: Convergence and Approximation*. Ph.D. Dissertation. Freie Universit at Berlin.

Yong-Liang Yang, Yu-Kun Lai, Shi-Min Hu, and Helmut Pottmann. 2006. Robust Principal Curvatures on Multiple Scales. In *Symposium on Geometry Processing*. 223–226.

Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2004. Mesh Editing with Poisson-Based Gradient Field Manipulation. In *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. 644–651.

Eugene Zhang, Konstantin Mischaikow, and Greg Turk. 2006. Vector Field Design on Surfaces. *ACM Transactions on Graphics* 25, 4 (2006), 1294–1326.