



The Diamond Laplace for Polygonal and Polyhedral Meshes

A. Bunge¹ M. Botsch¹  M. Alexa² 

¹TU Dortmund ²TU Berlin

Abstract

We introduce a construction for discrete gradient operators that can be directly applied to arbitrary polygonal surface as well as polyhedral volume meshes. The main idea is to associate the gradient of functions defined at vertices of the mesh with diamonds: the region spanned by a dual edge together with its corresponding primal element — an edge for surface meshes and a face for volumetric meshes. We call the operator resulting from taking the divergence of the gradient Diamond Laplacian. Additional vertices used for the construction are represented as affine combinations of the original vertices, so that the Laplacian operator maps from values at vertices to values at vertices, as is common in geometry processing applications. The construction is local, exactly the same for all types of meshes, and results in a symmetric negative definite operator with linear precision. We show that the accuracy of the Diamond Laplacian is similar or better compared to other discretizations. The greater versatility and generally good behavior come at the expense of an increase in the number of non-zero coefficients that depends on the degree of the mesh elements.

Keywords: Discrete Laplace Operator, Discrete Differential Geometry, DDFV

CCS Concepts

• Mathematics of computing → Discretization; • Computing methodologies → Mesh models;

1. Introduction

Discrete Laplace operators are an important tool in geometry processing, for surface as well as volumetric meshes [SCV14]. In many graphics applications, the mesh is given and not altered throughout interaction and processing. It provides the natural domain for discretizing differential operators. We dare say that, unlike in scientific computing, in graphics and geometry processing we fit the operator to the mesh rather than optimizing the mesh for solving the differential equation. Depending on the application, these meshes may contain different types of elements, such as triangles, quads, and general polygons for surface meshes, or tetrahedra, hexahedra, and general polyhedra for volumetric meshes.

If the mesh happens to be composed entirely of triangles or tetrahedra, the one operator that is used almost exclusively in geometry processing is the *cotan* Laplacian [PP93, MDSB03, DMSB99, Dzi88]. For surface meshes composed of higher order elements, i.e. polygons, possibly non-planar, there are several generalizations that reduce to the *cotan* Laplacian for triangle meshes [BHKB20, dGBD20, AW11, HKA15]. For volumetric meshes, besides tetrahedral elements [AHKSH20, Cra19], most other discretizations are based on hexahedra [SDG*19]. There are only few approaches for more general polyhedral meshes and they typically require numerical integration to obtain the basis functions for each element [WBG07, MKB*08, KBT17, MRS14].

In this paper we propose a simple and unified construction of gradient, divergence, and Laplace operators for general polygonal and polyhedral meshes. Our main idea, following the Discrete Duality Finite Volume approach (DDFV, Section 4), is to incorporate the primal as well as the (typically non-orthogonal) dual mesh and accommodate the oblique intersection of primal and corresponding dual elements. Specifically, we define discrete gradients, respectively divergences, per *diamond*: the region spanned by a dual edge and corresponding simplicial primal element. In 2D, the corresponding primal element is an edge; in 3D it is a triangle. If facets have degree higher than three, we insert an additional virtual vertex and in this way triangulate the facet. In all cases, the primal element is incident on two cells, and the dual vertices in these cells define two simplices. In other words, in 2D a diamond is spanned by two triangles; in 3D it is spanned by two tetrahedra.

In the spirit of the original mesh defining the domain for discretization we also want that the Laplace operator maps from values at vertices to values at vertices. We achieve this by combining the DDFV approach with the “sandwiching” of Bunge et al. [BHKB20]. This means all vertices except the primal ones are *virtual*: They are defined as affine combinations of primal vertices. These affine combinations are encoded as prolongation matrices and multiplied onto the DDFV Laplacian, thereby effectively hiding from the user the involved diamonds and virtual vertices.

The resulting construction is comparatively simple and seamlessly generalizes from arbitrary polygons (Section 5) to arbitrary polyhedra (Section 6). We discuss properties like locality, linear precision, semi-definiteness, the kernel, and the maximum principle in Section 7. The prolongation matrices lead to a decreased sparsity, resulting in increased computation cost. A range of numerical experiments (Section 8) suggests that the effort is well-spent: The accuracy is, in most cases, at least as good and even better than other constructions.

2. Problem Statement

We assume a mesh is given. The particular types of meshes we consider are two-dimensional surface meshes immersed in 3D and volumetric meshes embedded in 3D. Because they are most common, we focus here on the gradient, the divergence, and the Laplacian resulting from taking the divergence of the gradient.

We denote edges by tuples (i, j) , triangles by (i, j, k) , and tetrahedra by (i, j, k, l) , and use $\|(i, j)\|$, $|(i, j, k)|$, and $|(i, j, k, l)|$ to refer to their length, (signed) area, and (signed) volume, respectively. We use the operator $*$ to map between primal and dual entities, such that, e.g., $*i$ denotes the dual region of a vertex i and $*(i, j)$ denotes the dual edge of the primal edge (i, j) .

Given a polygonal or polyhedral mesh with vertices \mathcal{V} , edges \mathcal{E} , faces \mathcal{F} , cells \mathcal{C} , and diamonds \mathcal{D} , our aim is to generate the following matrices, representing discrete linear approximation of differential operators:

- The gradient $\mathbf{G} \in \mathbb{R}^{d \cdot |\mathcal{D}| \times |\mathcal{V}|}$, where $d \in 2, 3$ is the intrinsic dimension of the mesh and $|\mathcal{D}|$ is the number of diamonds to which the operator assigns constant gradients.
- The divergence $\mathbf{D} \in \mathbb{R}^{|\mathcal{V}| \times d \cdot |\mathcal{D}|}$, which we assume to be constructed as $\mathbf{D} = \mathbf{G}^T \mathbf{M}_\diamond$. Here, $\mathbf{M}_\diamond \in \mathbb{R}^{d \cdot |\mathcal{D}| \times d \cdot |\mathcal{D}|}$ is a diagonal matrix containing d -times the diamond masses.
- The operator matrix for the Laplacian is then constructed as $-\mathbf{D}^T \mathbf{G} = \mathbf{L} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$.

This construction ensures that the discrete operator is symmetric negative semi-definite. These properties are commonly considered desirable. In addition, we ask that the Laplacian operator maps constant vectors to zero and has linear precision, i.e. maps linear functions to themselves. For more information on these and potential additional properties we point the reader to the excellent discussion by Wardetzky et al. [WMKG07].

3. Related Work

The literature on discretizations of differential operators is vast. In computer graphics and geometry processing, discretizations based on the finite element method (FEM) and discrete exterior calculus (DEC) are most frequently used, with the cotan Laplacian for triangle meshes [PP93, MDSB03, DMSB99, Dzi88] and tetrahedral meshes [AHKSH20, Cra19] being the most prominent operator. Since our goal is a Laplacian operator for general polygonal surface meshes or general polyhedral volume meshes, we focus our discussion on operators that are not restricted to standard triangle/tetrahedral or quadrangle/hexahedral meshes.

There have been a couple of approaches for discretizing differential operators on polygonal meshes. These methods have to handle the problem that non-planar polygons do not bound a canonical surface in 3D. Alexa and Wardetzky [AW11] circumvent this problem by considering the projection of the polygon resulting in the largest area, combined with a MFD-based inner product stabilization [BLS05]. A drawback of this discretization is that it requires the suitable choice of a scalar parameter, and that the potentially large number of negative entries in the Laplacian matrix violates the discrete maximum principle [WMKG07].

Herholz et al. [HKA15] extend the definition of desirable properties of discrete Laplacians to polygon meshes, where they characterize polygon tessellations that admit a “perfect” operator with only non-negative weights. Sharp et al. [SSC19] handle potentially non-planar polygons by deriving a version of the “connection” Laplacian, which, in contrast to the standard expression as the divergence of the gradient, is given by the trace of the second covariant derivative. This operator fulfills many of the same properties as the cotan Laplacian and enables the diffusion of vectors from one tangent space to another on curved domains.

De Goes et al. [dGBD20] go beyond the Laplace operator and generalize a whole set of discrete differential operators to general polygonal meshes. They extend the approach of [AW11] by defining a new discrete gradient operator, which can be interpreted as a generalization of mimetic finite differences [BLS05] and the virtual element method [dVBM13]. Using the weak form of the cogradient operator and the divergence theorem, they bypass the need for an interpolation of the non-planar polygon surface. Bunge et al. [BHKB20] adapt the virtual node method [DLN07, TWZZ09] to polygon meshes by refining each face with a virtual vertex to span an implicit triangle fan, on which they apply the standard cotangent Laplacian. These virtual vertices are removed from the differential operators using special prolongation/restriction matrices. We make use of their idea of virtually refining the mesh and also use similar prolongation matrices, but use different, finite-volume-based discrete operators on the virtually refined mesh.

An alternative use of prolongation/restriction matrices was proposed by de Goes et al. [dGDMD16]. Their Subdivision Exterior Calculus (SEC) combines the Laplacian operator introduced by [AW11] with prolongation matrices corresponding to Catmull-Clark or Loop subdivision. Similar to our work, the prolongation results in a larger stencil for the discrete differential operators. However, SEC was designed to work on the geometry of the *limit surface* of the subdivision process, and hence is not suitable for computing directly on the user-provided polygon mesh.

While there is a certain variety of approaches for polygonal surface meshes, we are not aware of *simple* discrete differential operators for polyhedral volume meshes. Early works in computer graphics extended conforming FEM frameworks by using generalized barycentric coordinates [HS17] as custom shape functions for polyhedral elements, using either mean value coordinates [WBG07], harmonic coordinates [MKB*08, SDG*19], or maximum entropy coordinates [HS08]. This idea, which is also common in the computational mechanics literature [MRS14], has the drawback that both the computation and the numerical integration of these custom shape functions is rather complex and computationally expensive.

These methods do therefore not meet our goal of a *simple* Laplacian operator for polyhedral domains.

We believe that finite volume (FV) discretizations [Dro14], in particular the Discrete Duality Finite Volume (DDFV) method [Her00, DO05, Her09, CH11], offer an interesting alternative to deriving discrete differential operators for geometry processing applications. As our approach is inspired by and extends upon DDFV, we describe this approach in more detail in the next section.

4. Finite Volume Discretizations

Finite Volume (FV) methods are based on the idea to consider the *integral* of a *differential* in a small region. There exist a number of identities that allows expressing such integrals of a differential as an integral over only the boundary of the region. For the *divergence* of a vector-valued function \mathbf{u} over a flat two-dimensional region Ω we have

$$\iint_{\Omega} \operatorname{div} \mathbf{u} \, dA = \oint_{\partial\Omega} \mathbf{u}^T \mathbf{n} \, ds, \quad (1)$$

where \mathbf{n} is the outward normal along the boundary $\partial\Omega$. For $\mathbf{u} = u\mathbf{c}$ with constant vector \mathbf{c} and a scalar function u we can exploit the “product rule”

$$\operatorname{div}(u\mathbf{c}) = \underbrace{u \operatorname{div} \mathbf{c}}_{=0} + \mathbf{c}^T \nabla u = \mathbf{c}^T \nabla u. \quad (2)$$

Plugging this into the divergence theorem above for $\mathbf{c} = \mathbf{e}_k$ (the canonical unit vectors) and combining the results we find the vector-valued identity

$$\iint_{\Omega} \nabla u \, dA = \oint_{\partial\Omega} u \mathbf{n} \, ds. \quad (3)$$

Applying the divergence theorem to the vector field ∇u we get

$$\iint_{\Omega} \Delta u \, dA = \oint_{\partial\Omega} \nabla u^T \mathbf{n} \, ds. \quad (4)$$

All identities straightforwardly extend to higher dimensions.

The basic derivation of the Laplace operator with FVs in 2D makes the assumption that the mesh is Delaunay. This means the dual mesh is the Voronoi diagram, such that primal and dual edges are orthogonal. Consider a vertex i with position \mathbf{x}_i , the dual region associated to it is its Voronoi cell Ω_i . The function values of the unknown piecewise linear function u at vertex i are $u_i = u(\mathbf{x}_i)$. For the integrated Laplacian over the region Ω_i , the boundary $\partial\Omega_i$ is piecewise linear and consists of the dual edges $*(i, j)$, with $j \in N(i)$ denoting the one-ring neighbors of vertex i . If we denote by $\mathbf{e}_{ij} = \mathbf{x}_j - \mathbf{x}_i$ and $\mathbf{e}_{ij}^* = \mathbf{x}_j^* - \mathbf{x}_i^*$ the primal/dual edge vectors, respectively, and exploit that the normal \mathbf{n} on the dual edge $*(i, j)$ is

parallel to \mathbf{e}_{ij} and that the gradient of the piecewise linear function on the vertices points along this edge, we get

$$\begin{aligned} \iint_{\Omega_i} \Delta u \, dA &= \sum_{j \in N(i)} \int_{*(i,j)} \nabla u^T \mathbf{n} \, ds \\ &= \sum_{j \in N(i)} \int_{*(i,j)} \nabla u^T \frac{\mathbf{e}_{ij}}{\|\mathbf{e}_{ij}\|} \, ds \\ &= \sum_{j \in N(i)} \int_{*(i,j)} (u_j - u_i) \frac{1}{\|\mathbf{e}_{ij}\|} \, ds \\ &= \sum_{j \in N(i)} \frac{|\mathbf{e}_{ij}^*|}{\|\mathbf{e}_{ij}\|} (u_j - u_i). \end{aligned} \quad (5)$$

The last expression directly describes the construction of a linear operator that maps values at vertices $\{u_i\}$ to the integral over the region associated to vertex i of the Laplacian. Note that the (i, j) entry in the matrix \mathbf{L} is given by the (signed) length of the dual edge divided by the length of the primal edges. One obtains exactly the same result with the DEC approach [Hir03], which is based on similar arguments. Interestingly, also the Finite Element Method applied to triangles leads to these weights [PP93]. This suggests that the derivation extends to arbitrary triangulations, albeit carefully assigning *signed* lengths to the dual edges. The resulting negative coefficients for edges without the Delaunay property have undesirable consequences (see, for example, the discussion in [SSC19]). While this is often accepted for applications in graphics, in the FV community it is not considered admissible, which restricts the meshes to be (weighted) Delaunay. From a practical perspective, however, one is often given a mesh and it is costly to generate an orthogonal dual, if one exists [Aur87, Ale20].

Discrete Duality Finite Volume (DDFV)

The FV method that deals with this problem is to give up orthogonality. Rather, the idea is to construct a gradient operator and associate it to the region spanned by a pair of a primal edge (with endpoints \mathbf{x}_1 and \mathbf{x}_2) and its corresponding dual edge (with endpoints \mathbf{x}_l and \mathbf{x}_r). This region, depicted in Figure 1, is called a *diamond*.

Notice that a diamond is always a quadrilateral, regardless of the degree of the faces. The DDFV approach is to associate function values u_l and u_r to the dual vertices – thereby introducing a second set of degrees of freedom – and to associate a constant gradient with the diamond. As shown in Figure 1, left, we denote by D the diamond built from the four points $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_l, \mathbf{x}_r)$, by $(i, j) \in D$ its edges, and by $\mathbf{e}_{ij} = \mathbf{x}_j - \mathbf{x}_i$ the edge vectors. Making use of Stokes’ theorem gives

$$\begin{aligned} \iint_D \nabla u \, dA &= \oint_{\partial D} u \mathbf{n} \, ds \\ &= \sum_{(i,j) \in \partial D} \frac{\mathbf{e}_{ij}^\perp}{\|\mathbf{e}_{ij}\|} \int_0^1 \|\mathbf{e}_{ij}\| ((1-t)u_i + tu_j) \, dt \\ &= \sum_{(i,j) \in \partial D} \mathbf{e}_{ij}^\perp \frac{u_i + u_j}{2}. \end{aligned} \quad (6)$$

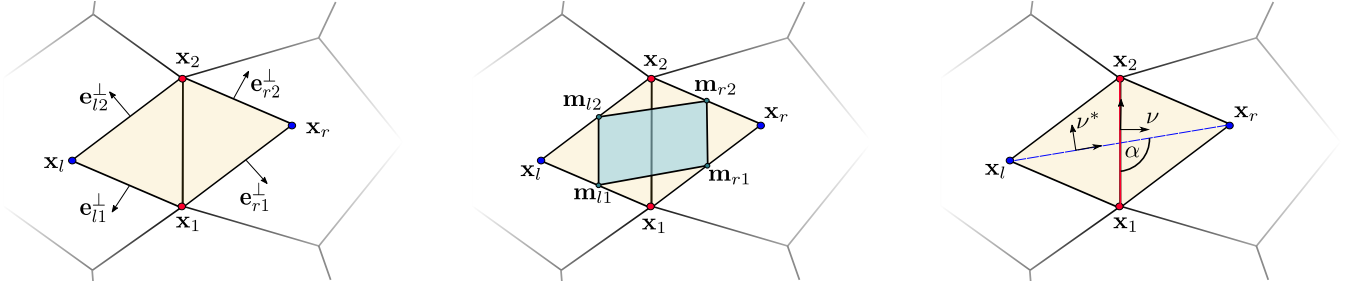


Figure 1: Different formulae and interpretations of the per-diamond gradient in 2D DDFV. Left: The vectors \mathbf{e}_{ij}^\perp orthogonal to the diamond edges \mathbf{e}_{ij} needed to compute our gradient for the diamond cell. Center: The midpoints \mathbf{m}_{ij} of the diamond edges and their enclosed subarea. Fitting an affine function to the function values at these midpoints is another possibility to obtain the 2D derivation of the diamond gradient. Right: Constructing gradients from primal and dual axes \mathbf{v}, \mathbf{v}^* and their enclosed angle α .

For the discrete gradient of the diamond we take the mean over the region, so we have to divide the integral by the area $|D|$, leading to

$$\nabla u|_D = \frac{1}{2|D|} \sum_{(i,j) \in \partial D} \mathbf{e}_{ij}^\perp (u_i + u_j). \quad (7)$$

The literature on DDFV [Her00, DO05, Her09, CH11] provides different derivations for the per-diamond gradient $\nabla u|_D$ and several interpretations and corresponding formulae:

- Fitting the gradient to directional derivatives along the primal and dual edges:

$$\begin{aligned} \nabla u|_D \cdot (\mathbf{x}_l - \mathbf{x}_r) &= u_l - u_r, \\ \nabla u|_D \cdot (\mathbf{x}_1 - \mathbf{x}_2) &= u_1 - u_2. \end{aligned} \quad (8)$$

- Fitting an affine function $w(x,y)$ to the function values $u_{ij} = \frac{1}{2}(u_i + u_j)$ at the midpoints $\mathbf{m}_{ij} = \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_j)$ of the four diamond edges $(i,j) \in \partial D$, and taking the gradient ∇w of this affine function (see Figure 1, center). Note that fitting an affine function to the four diamond vertices is an over-determined problem, while the midpoint fit is uniquely determined.
- A formulation based on the primal/dual axes $\mathbf{v} = (\mathbf{x}_2 - \mathbf{x}_1)^\perp / \|\mathbf{x}_2 - \mathbf{x}_1\|$ and $\mathbf{v}^* = (\mathbf{x}_r - \mathbf{x}_l)^\perp / \|\mathbf{x}_r - \mathbf{x}_l\|$ as well as their enclosed angle α (Figure 1, right):

$$\nabla u|_D = \frac{1}{\sin \alpha} \left(\frac{u_l - u_r}{\|\mathbf{x}_l - \mathbf{x}_r\|} \mathbf{v} + \frac{u_1 - u_2}{\|\mathbf{x}_1 - \mathbf{x}_2\|} \mathbf{v}^* \right), \quad (9)$$

Although these formulations are all equivalent, we believe our formulation (6) to be more intuitive when handling boundary cases and when generalizing to polyhedral meshes in Section 6. For boundary edges, the diamond consists of a single triangle $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_l)$ only. The typical DDFV approach is to replace \mathbf{x}_r by the edge midpoint $\frac{1}{2}(\mathbf{x}_1 + \mathbf{x}_2)$ and to properly deal with degenerate edges/faces. In contrast, our formulation (7) remains unchanged.

The divergence $\operatorname{div} u$ and the Laplacian $\Delta u = \operatorname{div} \nabla u$ can be obtained through very similar derivations. The resulting DDFV gradient operator maps from function values at primal and dual vertices to gradients at diamonds, the divergence operator from vectors at diamonds to scalars at primal/dual vertices. The DDFV Laplacian therefore maps functions values at primal/dual vertices to their Laplacians sampled at primal/dual vertices.

5. Diamond Laplace for Surface Meshes

In its standard formulation, the DDFV operators are not directly useful for applications in computer graphics and geometry processing, since they have two main drawbacks: First, by introducing function values at dual vertices it significantly increases the number of degrees of freedom (DoF) to be solved for. For instance, the DoFs are roughly tripled for triangle meshes and roughly doubled for quad meshes. Second, the approach is defined for planar 2D meshes only, but not for two-manifold surface meshes embedded in 3D, which we are mostly interested in.

In this section we address both problems. Replacing the extrinsic per-diamond gradient by a version that is intrinsic w.r.t. the polygonal mesh allows us to generalize the 2D DDFV scheme to embedded surface meshes (Section 5.1). Following Bunge et al. [BHKB20] and representing the dual DoFs as interpolations of the primal DoFs and incorporating this through special prolongation/restriction matrices will remove the dual DoFs and eventually keep only the primal ones.

5.1. Intrinsic Gradient

Compared to mesh faces, diamonds are the better entity to associate gradients with, since for general polygonal meshes higher-degree faces might not be planar and typically cannot be flattened without introducing distortion. Diamonds spanned by a pair of primal vertices $\mathbf{x}_1, \mathbf{x}_2$ and dual vertices $\mathbf{x}_l, \mathbf{x}_r$ are not necessarily planar in the 3D embedding, but can be isometrically unfolded into the plane around their primal “hinge” edge $(\mathbf{x}_1, \mathbf{x}_2)$. We can therefore represent the diamond in an intrinsic 2D coordinate system, which allows us to then directly apply the gradient construction of (7).

It is convenient to choose the primal edge as the first coordinate axis, i.e.,

$$\mathbf{u} = \frac{\mathbf{x}_j - \mathbf{x}_i}{\|\mathbf{x}_j - \mathbf{x}_i\|}. \quad (10)$$

The second coordinate axis has to be orthogonal to this axis, contained in the planes spanned by the two triangles $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_l)$ and $(\mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_r)$, and consistently oriented w.r.t. \mathbf{u} . We achieve this by

projecting the edges $(1, l)$ and $(1, r)$ onto the orthogonal complement of the first axis \mathbf{u} and normalizing the result:

$$\begin{aligned}\tilde{\mathbf{v}}_l &= (\mathbf{I} - \mathbf{u}\mathbf{u}^\top)(\mathbf{x}_l - \mathbf{x}_1), & \mathbf{v}_l &= \tilde{\mathbf{v}}_l / \|\tilde{\mathbf{v}}_l\|, \\ \tilde{\mathbf{v}}_r &= (\mathbf{I} - \mathbf{u}\mathbf{u}^\top)(\mathbf{x}_r - \mathbf{x}_1), & \mathbf{v}_r &= \tilde{\mathbf{v}}_r / \|\tilde{\mathbf{v}}_r\|.\end{aligned}\quad (11)$$

Notice that both directions \mathbf{v}_l and \mathbf{v}_r are consistently oriented, are intrinsically in one plane, and form an orthonormal frame with \mathbf{u} . In this frame, the 2D coordinates of the four diamond vertices are

$$\begin{aligned}\mathbf{x}_1^{2D} &= (0, 0)^\top, \\ \mathbf{x}_2^{2D} &= (\|\mathbf{x}_2 - \mathbf{x}_1\|, 0)^\top, \\ \mathbf{x}_l^{2D} &= (\mathbf{u}, \mathbf{v}_l)^\top(\mathbf{x}_l - \mathbf{x}_1), \\ \mathbf{x}_r^{2D} &= (\mathbf{u}, \mathbf{v}_r)^\top(\mathbf{x}_r - \mathbf{x}_1).\end{aligned}\quad (12)$$

These 2D coordinates can now be used in the gradient construction of Equation (7), yielding an *intrinsic* 2D gradient per diamond.

From (7) we can then directly read off the the entries for the diamond's *gradient operator* matrix $\mathbf{G}_D \in \mathbb{R}^{2 \times 4}$ by noticing that the value i depends only on the two diamond edges incident on it. Therefore, the i -th column of \mathbf{G}_D is

$$\mathbf{G}_D(:, i) = \frac{1}{2|D|} \sum_{(i,j) \in \partial D} \mathbf{e}_{ij}^\perp. \quad (13)$$

The matrix $\hat{\mathbf{G}}$ for the global gradient operator, mapping from function values at primal and dual vertices to gradients at diamonds, is then assembled from all diamond gradient matrices

$$\hat{\mathbf{G}} = \bigoplus_{D \in \mathcal{D}} \mathbf{G}_D, \quad (14)$$

where \bigoplus is the assembly operator that scatters and accumulates the entries of the local matrices into the global matrix.

5.2. Dual Vertices as Affine Combinations

Our approach to remove the dual DoFs from the DDFV formulation is inspired by Bunge et al. [BHKB20], who also introduce dual vertices into primal faces, but represent their position and function values as affine combinations of the positions/values of the face's vertices.

Consider a general polygonal face f with n vertices \mathbf{x}_i , $i \in f$. We construct the dual face point \mathbf{x}_f , which takes the role of \mathbf{x}_l or \mathbf{x}_r in the gradient construction described above, as an affine combination of the face vertices

$$\mathbf{x}_f = \sum_{i \in f} a_{i,f} \mathbf{x}_i \quad \text{with} \quad \sum_{i \in f} a_{i,f} = 1. \quad (15)$$

The dual DoFs, i.e., the function values at dual face vertices \mathbf{x}_f , are then represented in terms of the primal DoFs by the same affine combination:

$$u(\mathbf{x}_f) = \sum_{i \in f} a_{i,f} u_i. \quad (16)$$

For a polygonal mesh with $|\mathcal{V}|$ vertices and $|\mathcal{F}|$ faces, this construction can be packed into an $(|\mathcal{V}| + |\mathcal{F}|) \times |\mathcal{V}|$ prolongation matrix \mathbf{P} with entries

$$\mathbf{P}_{ij} = \begin{cases} 1 & \text{if } i = j \text{ and } i < |\mathcal{V}|, \\ a_{i,f} & \text{if } i = |\mathcal{V}| + f \text{ and vertex } j \in \text{face } f, \\ 0 & \text{otherwise.} \end{cases} \quad (17)$$

Combining the gradient matrix (14) and the prolongation matrix (17) yields our gradient operator for polygonal meshes

$$\mathbf{G} = \hat{\mathbf{G}}\mathbf{P} \in \mathbb{R}^{2|\mathcal{E}| \times |\mathcal{V}|}, \quad (18)$$

where $|\mathcal{E}|$ denotes the number of edges (and therefore of diamonds) in the mesh. This matrix maps scalar function values u_i at *primal vertices* to 2D intrinsic gradient vectors $\nabla u|_D$ at diamonds.

While for standard FV methods with orthogonal duals the dual point \mathbf{x}_f is the circum-center of triangle f , the canonical choice for general polygonal meshes in the DDFV literature [Her00, DO05, Her09, CH11] is the face's barycenter. However, as the barycenter is not necessarily inside a non-convex (planar) face, we instead follow [BHKB20] and compute \mathbf{x}_f (respectively its affine weights $a_{i,f}$) by minimizing the sum of squared triangle areas

$$\min_{\mathbf{x}_f} \sum_{(i,j) \in f} |(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_f)|^2. \quad (19)$$

For a face f with n vertices this minimization requires solving a small $n \times n$ linear system (see [BHKB20] for details) and indeed yields more accurate results than other point choices in case of non-convex polygons (see Section 8).

5.3. Divergence and Laplacian

With the intrinsic gradient (18) in place, we can now define the discrete divergence and Laplacian. The DDFV discretization of the divergence operator leads to a matrix $-\hat{\mathbf{G}}^\top \mathbf{M}_\diamond$, which we combine with the transposed prolongation (or restriction) matrix to get the diamond divergence matrix

$$\mathbf{D} = -\mathbf{P}^\top \hat{\mathbf{G}}^\top \mathbf{M}_\diamond. \quad (20)$$

Here, $\hat{\mathbf{G}}$ is the gradient matrix of (14) and \mathbf{M}_\diamond is a $2|\mathcal{E}| \times 2|\mathcal{E}|$ diagonal matrix with the area $|D_i|$ of diamond D_i in its entries $(2i, 2i)$ and $(2i+1, 2i+1)$. This operator maps intrinsic 2D vectors at diamonds to scalar values at primal vertices.

The (integrated) diamond Laplace operator is finally defined as the diamond divergence of the diamond gradient, i.e.,

$$\mathbf{L} = \mathbf{D}\mathbf{G} = -\mathbf{P}^\top \hat{\mathbf{G}}^\top \mathbf{M}_\diamond \hat{\mathbf{G}}\mathbf{P}, \quad (21)$$

and maps from vertices to vertices. The pointwise Laplacian is obtained as $\mathbf{M}^{-1}\mathbf{L}$ by multiplying with the inverse of the mass matrix \mathbf{M} . This mass matrix is defined as

$$\mathbf{M} = \mathbf{P}^\top \hat{\mathbf{M}}\mathbf{P} \quad (22)$$

from the standard DDFV diagonal mass matrix $\hat{\mathbf{M}}$

$$\hat{\mathbf{M}}_{ii} = \begin{cases} \sum_{D \ni i} \frac{1}{4} |D| & \text{if } i \text{ is a primal vertex,} \\ \sum_{D \ni i} \frac{1}{4} |D| & \text{if } i \text{ is a dual face vertex,} \\ 0 & \text{otherwise,} \end{cases} \quad (23)$$

which assigns to the four (primal and dual) vertices of a diamond one fourth of its area. The “sandwiching” with \mathbf{P}^T and \mathbf{P} distributes the mass from primal and dual vertices to the primal vertices only. Note that the sandwiching leads to a non-diagonal mass matrix \mathbf{M} . We avoid lumping this matrix to a diagonal matrix, since numerical results have shown that the initial matrix leads to higher accuracy of our operator. Notice that in above construction, the dual points do not have to be inserted into the mesh explicitly, nor do the matrices $\hat{\mathbf{G}}, \mathbf{M}_\diamond, \mathbf{P}$ have to be built explicitly. Instead, the matrices \mathbf{G} and \mathbf{M} can directly be constructed through a diamond-based matrix assembly, which *implicitly* computes the virtual vertices and their affine weights, similar to the construction of Bunge et al. [BHKB20].

6. Diamond Laplace for Volume Meshes

One particular advantage of our diamond Laplace is that it can be generalized to 3D polyhedral meshes in an intuitive and consistent manner. Given a general polyhedral mesh with vertices \mathcal{V} , edges \mathcal{E} , faces \mathcal{F} , and cells \mathcal{C} , we will define diamonds \mathcal{D} from primal and dual vertices. The starting point of our construction is the generalization of the (integrated) gradient of a function u over a diamond. Analogous to the 2D case, given the gradient operator \mathbf{G} , we also have a divergence operator \mathbf{D} and can then assemble the Laplacian $\mathbf{L} = \mathbf{D}\mathbf{G}$. Representing the dual vertices as affine combinations of primal vertices will again define the sandwiching operator $\mathbf{P}^T(\cdot)\mathbf{P}$ that removes the dual DoFs. In the following we provide the details of these steps, in particular where they deviate from the case for surface meshes.

6.1. Integrated Gradient

Before we focus on the particular shape and construction of the diamonds, we derive the integral of the gradient (and, by analogy, divergence) for an arbitrary region Ω bounded by a *triangulated* surface. We assume the function over the triangulated boundary to be linear on the triangles, defined by values u_i at vertices i . If the triangles are given as triples of indices $(i, j, k) \in \partial\Omega$, we get

$$\begin{aligned} \iiint_{\Omega} \nabla u \, dV &= \iint_{\partial\Omega} u \mathbf{n} \, dA \\ &= \sum_{(i,j,k) \in \partial\Omega} \frac{\mathbf{a}_{ijk}}{\|\mathbf{a}_{ijk}\|} \int_0^1 \int_0^t \|\mathbf{a}_{ijk}\| ((1-s-t)u_i + su_j + tu_k) \, ds \, dt \\ &= \sum_{(i,j,k) \in \partial\Omega} \mathbf{a}_{ijk} \frac{u_i + u_j + u_k}{3}, \end{aligned} \quad (24)$$

where

$$\mathbf{a}_{ijk} = \frac{1}{2} (\mathbf{x}_j - \mathbf{x}_i) \times (\mathbf{x}_k - \mathbf{x}_i) \quad (25)$$

is the *area vector* of triangle (i, j, k) , i.e., the vector pointing in outward normal direction and with magnitude equal to the area of the triangle (see Figure 3, left). Taking the mean over the region by dividing the integral by the volume $|\Omega|$ leads to

$$\nabla u|_{\Omega} = \frac{1}{3|\Omega|} \sum_{(i,j,k) \in \partial\Omega} \mathbf{a}_{ijk} (u_i + u_j + u_k). \quad (26)$$

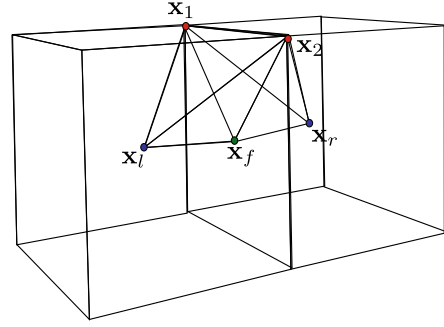


Figure 2: A minimal diamond spanned by two cell points $\mathbf{x}_l, \mathbf{x}_r$, a face point \mathbf{x}_f , and a primal edge $\mathbf{x}_1, \mathbf{x}_2$.

The local gradient operator for the region Ω , mapping values at the vertices $i \in \partial\Omega$ to a constant 3D gradient vector, is then built in a column-wise manner as

$$\mathbf{G}_{\Omega}(:, i) = \frac{1}{3|\Omega|} \sum_{(i,j,k) \in \partial\Omega} \mathbf{a}_{ijk}, \quad (27)$$

which is consistent with the 2D version of Equation (13).

6.2. Diamond Rings and Minimal Diamonds

The canonical choice for a diamond in a volumetric mesh would be associated with a dual edge (l, r) with endpoints $\mathbf{x}_l, \mathbf{x}_r$. These two dual vertices, together with the primal vertices $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$ of the face $f = *(l, r)$ that is dual to (l, r) , define a region that is bounded by two triangle fans spanned by \mathbf{x}_l or \mathbf{x}_r and two neighboring vertices $\mathbf{x}_i, \mathbf{x}_{i+1}$ of the face f . Given that the integrated gradient can be computed easily for this region as shown above, it may be tempting to assign a gradient to each such diamond (as done in [Her09]). Yet, similar to other constructions for non-simplicial meshes [AW11, dGBD20], constructing the gradient, divergence, and Laplacian in this way and then sandwiching the resulting matrices leads to *spurious modes*, i.e., a Laplacian operator with more than the constant functions in its kernel. This would be a serious drawback, and is a known limitation of the CeVe DDFV method [Her09], as discussed for instance in [ABH13].

Since we are adding a dual vertex to each cell, all vertices in a cell become connected in the operator. Consequently, adding a dual vertex \mathbf{x}_f to each face f introduces no additional non-zeros in the operator. Based on this virtual face vertex, the diamond is decomposed into a *ring of diamonds*, where each individual diamond is *minimal*, i.e., consists of two tetrahedra with tips $\mathbf{x}_l, \mathbf{x}_r$ and a base triangle $(\mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{x}_f)$, as shown in Figure 2. Basing the construction in these minimal elements ensures that the kernel of the Laplace operator only contains the constants.

Incidentally, minimal diamonds are the right analogy to 2D diamonds, in the following sense: Consider a minimal diamond defined by $\mathbf{x}_l, \mathbf{x}_r$ and $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ and the midpoints of the 6 edges emanating from \mathbf{x}_l and \mathbf{x}_r (see Figure 3, right):

$$\mathbf{m}_{li} = \frac{1}{2} (\mathbf{x}_l + \mathbf{x}_i), \quad \mathbf{m}_{ri} = \frac{1}{2} (\mathbf{x}_r + \mathbf{x}_i), \quad i = 1, 2, 3. \quad (28)$$

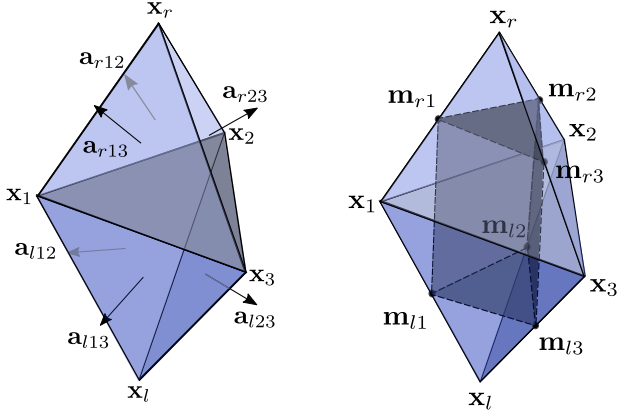


Figure 3: For a minimal diamond consisting of two tetrahedra, the gradient can be computed from the area vectors \mathbf{a}_{ijk} of its faces (left) or by fitting an affine function to edge midpoints \mathbf{m}_{ij} (right).

We observe that these six midpoints form a *parallelopete*:

1. The two triangles $(\mathbf{m}_{l1}, \mathbf{m}_{l2}, \mathbf{m}_{l3})$ and $(\mathbf{m}_{r1}, \mathbf{m}_{r2}, \mathbf{m}_{r3})$ are parallel to the triangle $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$, in fact, translates scaled by a factor of $\frac{1}{2}$.
2. The quad $\mathbf{m}_{l1}, \mathbf{m}_{r1}, \mathbf{m}_{r2}, \mathbf{m}_{l2}$ is a planar parallelogram, and likewise for the other two quads. All edges $\mathbf{m}_{li} - \mathbf{m}_{ri}$ connecting corresponding points on opposite sides are copies of the vector $\mathbf{x}_l - \mathbf{x}_r$, scaled by a factor of $\frac{1}{2}$.

This means any two edges of the triangle $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ together with the vector $\mathbf{x}_l - \mathbf{x}_r$ span the linear part of the affine space defined by the six points. Hence, an affine function can uniquely be fitted to these midpoints – analogously to the 2D parallelogram version show in Figure 1, center – and the gradient of this function can be used as the diamond gradient (giving the same result as (27)).

6.3. Dual Vertices as Affine Combinations

There have been several extensions of the DDFV scheme to volumetric meshes, see Hubert and colleagues [CH11, ABHK12] for a good overview. Most 3D DDFV methods define the gradient on minimal diamonds, as proposed above, but require the insertion of additional vertices (and their associated DoFs) per cell, face, and edge, thereby significantly increasing the number of degrees of freedom. Our construction requires virtual vertices per cell and face only, but their DoFs are eventually removed by the sandwiching operator.

Analogous to the surface case, we first insert for each face the point that minimizes the sum of squared triangle areas (see Equation (19)), turning each face into a fan of (virtual) triangles. For a polyhedral cell c , the virtual point \mathbf{x}_c is then computed to minimize the sum of squared tetrahedron volumes:

$$\min_{\mathbf{x}_c} \sum_{(i,j,k) \in \partial c} |(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \mathbf{x}_c)|^2. \quad (29)$$

For a cell c with m vertices (consisting of primal vertices and virtual face vertices), the above minimization requires to solve an $m \times m$ linear system for the affine weights defining \mathbf{x}_c .

This two-step sandwiching procedure results in two prolongation matrices \mathbf{P}_F and \mathbf{P}_C for inserting face and cell points, respectively, which are then combined to the prolongation matrix

$$\mathbf{P} = \mathbf{P}_C \mathbf{P}_F. \quad (30)$$

6.4. Gradient, Divergence, Laplace

The global gradient operator $\hat{\mathbf{G}}$, mapping values at primal vertices and dual face/cell points, is again constructed by assembling per-diamond gradient matrices \mathbf{G}_D , and is then combined with the prolongation matrix to yield \mathbf{G}

$$\hat{\mathbf{G}} = \bigoplus_{D \in \mathcal{D}} \mathbf{G}_D, \quad \mathbf{G} = \hat{\mathbf{G}} \mathbf{P}, \quad (31)$$

where \bigoplus again is the matrix assembly operator. Following the 2D derivation, the divergence and Laplacian operators for polyhedral meshes become

$$\mathbf{D} = -\mathbf{P}^T \hat{\mathbf{G}}^T \mathbf{M}_\diamond, \quad \mathbf{L} = \mathbf{D} \mathbf{G}, \quad (32)$$

with a diagonal matrix \mathbf{M}_\diamond containing diamond volumes. The mass matrix $\mathbf{M} = \mathbf{P}^T \hat{\mathbf{M}} \mathbf{P}$, required for the point-wise Laplacian $\mathbf{M}^{-1} \mathbf{L}$, is defined in terms of the diagonal matrix $\hat{\mathbf{M}}$, which distributes the volumes of the (minimal) diamonds D to the primal vertices, face vertices, and cell vertices:

$$\hat{\mathbf{M}}_{ii} = \begin{cases} \sum_{D \ni i} \frac{1}{6} |D| & \text{if } i \text{ is a primal vertex,} \\ \sum_{D \ni i} \frac{1}{6} |D| & \text{if } i \text{ is a face vertex,} \\ \sum_{D \ni i} \frac{1}{4} |D| & \text{if } i \text{ is a cell vertex,} \\ 0 & \text{otherwise.} \end{cases} \quad (33)$$

Analogous to the surface construction, we avoid lumping the mass matrix and instead work with the non-diagonal matrix \mathbf{M} .

7. Properties

We analyze the properties of our Diamond Laplacian with respect to the criteria listed in [WMKG07].

Symmetry, Definiteness By construction, the Diamond Laplacian $\mathbf{L} = \mathbf{D} \mathbf{G} = -\mathbf{P}^T \hat{\mathbf{G}}^T \mathbf{M}_\diamond \hat{\mathbf{G}} \mathbf{P}$ is a real-valued symmetric negative semi-definite matrix, since the diagonal matrix \mathbf{M}_\diamond (containing diamond areas/volumes) is symmetric positive definite.

Linear Precision If the mesh is flat, i.e., a polygon mesh embedded in a plane, respectively a polyhedral mesh embedded in 3D, then we expect that the discrete Laplacian vanishes on linear functions away from the boundary of the domain. The DDFV gradient $\hat{\mathbf{G}}$ of a linear function over a closed region with polygonal or polyhedral boundary reproduces the constant gradient of this function. The divergence operator $-\hat{\mathbf{G}}^T \mathbf{M}_\diamond$ is exact on the resulting constant vector fields, leading to linear precision of the DDFV Laplacian $-\hat{\mathbf{G}}^T \mathbf{M}_\diamond \hat{\mathbf{G}}$ [DO05, Her09, CH11]. The sandwiching $\mathbf{P}^T(\cdot)\mathbf{P}$ preserves this linear precision [BHKB20].

Null-Space The Diamond Laplacian has a one-dimensional kernel containing only the constant functions. It is obvious that constant functions are sufficient for the gradient to vanish, implying that they are in the kernel of the Laplacian. It remains to show that constant values are necessary for the gradient to vanish.

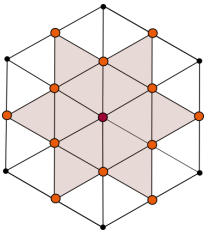
We explain the situation for minimal diamonds in 3D – the case for surface meshes works analogously. The gradient of a minimal diamond can be interpreted as the gradient of the affine function interpolating the values on the edge midpoints \mathbf{m}_{li} , \mathbf{m}_{ri} , $i = 1, 2, 3$ (Section 6.2). For these values to be identical it is necessary that (i) the values at \mathbf{x}_1 , \mathbf{x}_2 , and \mathbf{x}_3 are identical and (ii) the values at \mathbf{x}_l and \mathbf{x}_r are identical. Because the mesh is connected, it follows that the values at all primal vertices and at all dual vertices need to be identical. Notice that this argument cannot be extended to diamonds with a non-triangular base: If the base is a polygon with more than three vertices already the gradient within this polygon may vanish for non-constant values on the vertices. This problem for diamonds with non-triangular base has also been described in the DDFV literature (cf. [ABH13]).

It remains to explain why the constant values on primal vertices and the constant values on dual vertices are identical. In our setup this follows directly from the fact that values on dual vertices are affine combinations of the values in primal vertices. In other words, while $\hat{\mathbf{G}}$ may have a two-dimensional kernel, the kernel of $\hat{\mathbf{G}}\mathbf{P}$ is guaranteed to contain only the constant functions. As long as the diamond mass matrix \mathbf{M}_\diamond has full rank this implies that also \mathbf{L} has the desired kernel.

Lastly, note that the DDFV literature only considers meshes with boundary, where values on primal and dual vertices are connected through identification on boundary edges. In this case, $\hat{\mathbf{G}}$ already has the desired kernel [ABH13]. For meshes without boundary this fails. Our sandwiching approach rectifies the situation.

Locality The Diamond Laplacian is local, but less local than related existing schemes, since the diamond gradient couples neighboring cells and the sandwiching couples all primal vertices incident on a cell. For simplicial meshes, the cotan Laplacian of a vertex i depends on all vertices sharing an edge with i . For the polygonal Laplacians [AW11, BHKB20, dGBD20] it depends on all vertices sharing a face with i . For the Diamond Laplacian it depends on the vertices of (i) the cells incident on vertex i and (ii) the cells sharing a face with these cells.

This set is larger than the vertices in the edge-based or cell-based one-ring neighborhood, but generally smaller than edge-based two-ring neighborhood. For instance, on a regular triangle mesh the Laplacian of vertex i depends on 12 neighbors, which is in between the 6 and 18 vertices of the one-ring and two-ring neighborhoods, respectively (see inset figure).



Maximum principle The maximum principle for discrete Laplace operators is commonly derived from the sign structure of the operator matrix. If the diagonal elements are all negative and the off-diagonal elements are all positive, then the operator is an M -matrix,

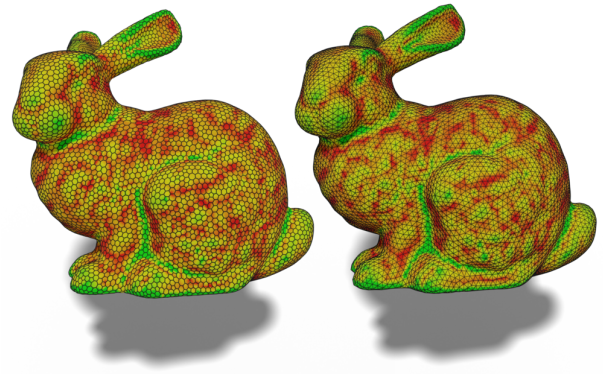


Figure 4: Color-coded absolute mean curvature for two different tessellations computed with the Diamond Laplacian (left: hexagon-dominant, right: triangles). The results are visually identical.

implying the maximum principle. The sign of the off-diagonal entries in \mathbf{L} depends on the input mesh, and like other Laplacians the Diamond Laplacian, in general, has no maximum principle. A Delaunay *triangle* mesh guarantees the desired signs for the coefficients of the cotan operator. For Delaunay *tetrahedral* meshes, the cotan operator has no maximum principle. The DEC construction does provide the maximum principle for Delaunay meshes, but may lack semi-definiteness if the mesh is not Delaunay [AHKSH20]. The Diamond Laplacian, in contrast, has no maximum principle even for Delaunay triangle or tetrahedral meshes.

8. Results

We evaluate the Diamond Laplacian in a variety of geometry processing operations for both volume and surface meshes. We focus our presentation of results on quantitative tests, in particular on comparisons to other constructions for non-simplicial meshes.

8.1. Surface Meshes

On surface meshes, we compare our Diamond Laplacian to the polygon Laplacians of Alexa and Wardetzky [AW11], Bunge et al. [BHKB20], and de Goes et al. [dGBD20]. According to the recommendation of the original authors, we chose the involved hyperparameters as $\lambda = 2$ for [AW11] and $\lambda = 1$ for [dGBD20].

Mean Curvature When applied to the coordinate function of the surface mesh, the Laplace operator yields an approximation of the integrated mean curvature vector. We approximate the point-wise mean curvature H at a vertex i as

$$H(i) = \frac{1}{2} \left\| \left(\mathbf{M}^{-1} \mathbf{L} \mathbf{X} \right)_i \right\|,$$

as visualized for a triangle mesh and a general polygon mesh in Figure 4. Quantitative comparisons to other methods are provided in Figure 5, where we compare root-mean-square errors (RSME) on different tessellations of the unit sphere. Since the convergence of point-wise mean curvatures under refinement is not guaranteed and depends on the type of tessellation, we restrict our evaluation

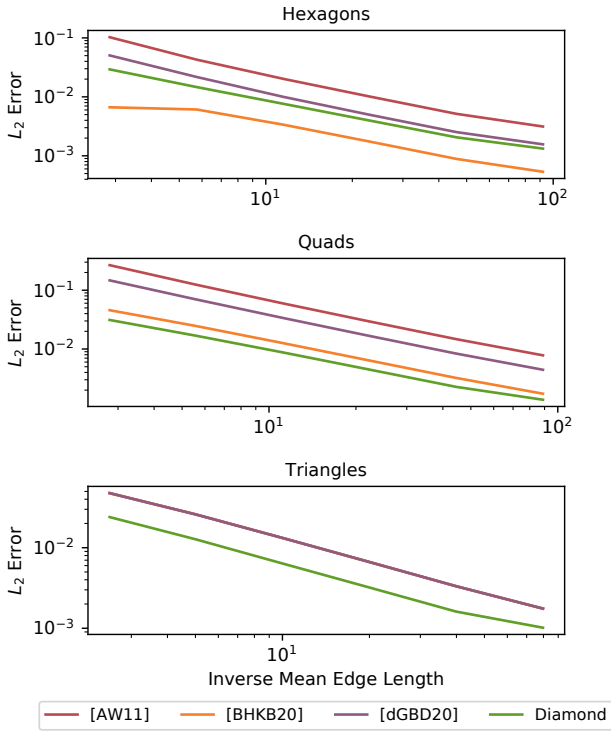


Figure 5: Root-mean-square error of absolute mean curvatures in log-log scale for different tessellations of the unit sphere. The different tessellation types are depicted in Figure 9.

to tessellations for which we observe convergence. The Diamond Laplacian generally yields the lowest errors for triangle and quad meshes, and is only bested on hexagon meshes, where it comes second. Note that in contrast to the polygon Laplacians by [AW11, BHKB20, dGBD20], the Diamond Laplacian does *not* reduce to the cotan Laplacian in case of triangle meshes, but instead provides a more accurate discretization.

Geodesic in Heat The heat method of Crane et al. [CWW13] approximates geodesic distances from a vertex i to every other vertex. Since the gradient and divergence operators are directly involved in several computation steps, the gradient defined on the diamonds makes the application of this method natural.

An important parameter in the heat method is the time step used for heat diffusion. While Crane et al. [CWW13] suggest the *mean* edge length squared, a more conservative choice is the square of the *maximum* edge length [dGDMD16], which we used in our experiments. A qualitative comparison of the distances on different tessellations is shown in Figure 6. The quantitative evaluation displayed in Figure 7 shows the results for different tessellations of the unit sphere (higher-resolution versions of the sphere meshes depicted in Figure 9). The errors are based on the analytical values of the great-circle distance. With few exceptions, the Diamond Laplacian yields the lowest errors. This is a good indicator for the quality of the gra-

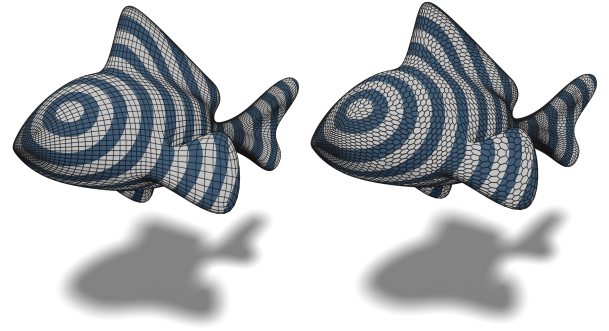


Figure 6: Geodesic distances obtained with the Diamond Laplacian, being visually identical despite different tessellations.

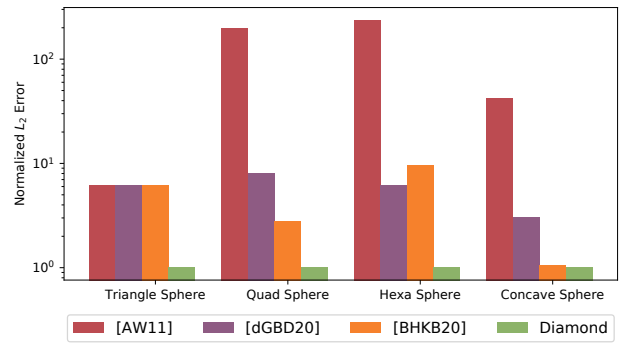


Figure 7: Root-mean-square error of geodesic distances for different tessellations of the unit sphere.

dent and divergence operators defined on diamonds compared to other constructions.

We also evaluated the accuracy of the Diamond operator constructed by placing dual vertices \mathbf{x}_f at face *centroids* instead of the minimizer of the squared triangle areas. While the centroid is typically employed in the DDFV literature, its performance on non-convex tessellations (similar to the L-plane and Tetris meshes used in [BHKB20]) yields worse results than the area minimizer, since flipped triangles lead to unfavorable diamond cells, which in turn reduce the overall performance of the operator.

Poisson Problems We also analyze the performance of the Diamond Laplacian for Poisson equations $\mathbf{L}\mathbf{u} = \mathbf{M}\mathbf{b}$ on different tessellations of the unit square, and compare it to existing approaches. We employ two different Dirichlet boundary conditions \mathbf{b} :

1. The Laplacian of Franke's test function [Fra79]

$$F_{2D}(x, y) = \frac{3}{4}e^{-\frac{(9x-2)^2 + (9y-2)^2}{4}} + \frac{3}{4}e^{-\frac{(9x+1)^2}{49} - \frac{9y+1}{10}} + \frac{1}{2}e^{-\frac{(9x-7)^2 + (9y-3)^2}{4}} - \frac{1}{5}e^{-(9x-4)^2 - (9y-7)^2}.$$

We then measure the deviation of the reconstruction from the true function values.

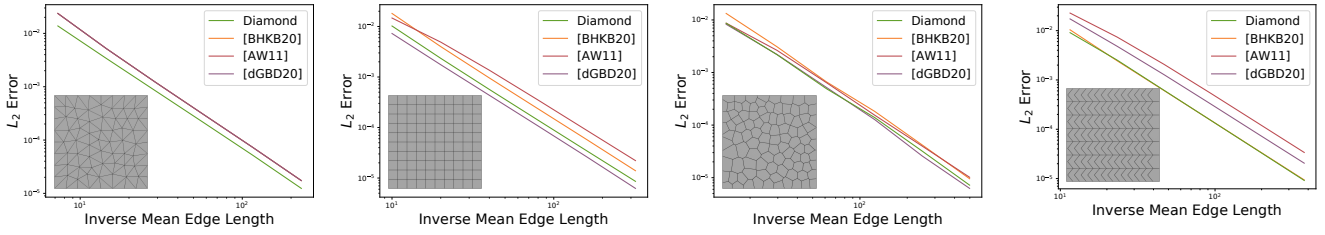


Figure 8: L_2 error in log-log scale of the Poisson system solved for Franke’s test function on planar grids with triangles (left), quads (center left), Voronoi cells (center right), and concave faces (right). On triangle meshes, the operators [AW11, BHKB20, dGBD20] reduce to the cotangent Laplacian and hence yield the same results.

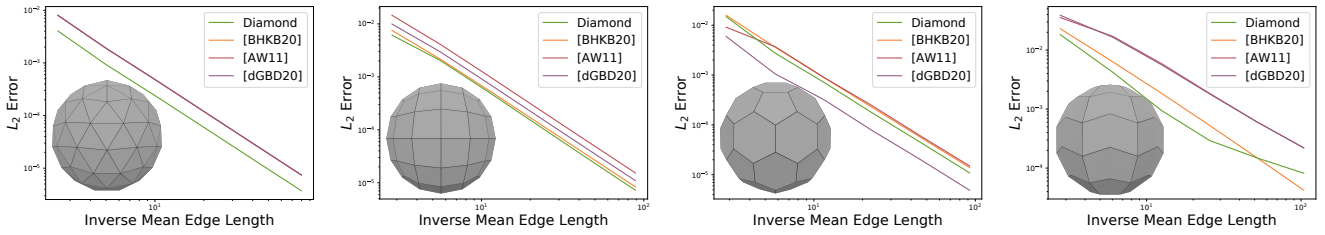


Figure 9: L_2 error in log-log scale of the Poisson system solved for the scaled real-valued spherical harmonic function $Y_3^{-1}(x, y, z)$ on differently tessellated unit spheres: Triangles (left), quads (center left), Hexagons (center right), and concave faces (right). On triangle meshes, the operators [AW11, BHKB20, dGBD20] reduce to the cotangent Laplacian and hence yield the same results.

2. The values of the scaled spherical harmonic function

$$Y_3^{-1}(x, y, z) = y(4z^2 - x^2 - y^2).$$

We then measure the deviation of the reconstruction from the input Y_3^{-1} divided by the corresponding eigenvalue $\lambda = 12$.

As can be seen in Figures 8 and 9, the Diamond Laplacian has the expected convergence rate and provides lower errors than other discretizations for the majority of test cases. For triangle meshes, the Diamond Laplacian performs favorably for both problems and is more accurate than the cotangent operator, to which the polygon Laplacians of [AW11, BHKB20, dGBD20] reduce.

Sparsity and Timings The operator matrix for polygon Laplacians has more non-zero elements than the corresponding adjacency matrix. This reflects that at least the vertices belonging to the same face are connected, since they all influence the (integrated) differential properties of the face. The choice of diamonds as regions for estimating the differentials allows a more accurate estimation of the gradient across (primal) edges. This comes at the expense of introducing additional non-zero entries in the operator matrix that reflect this connection. The reduced sparsity results in higher computational complexity for solving the involved linear systems. Table 1 lists the timings of sparse Cholesky factorization and back-substitution, computed using the supernodal LLT solver of CHOLMOD [CDHR08]. All timings were measured on a standard workstation with a six-core Intel Xeon 3.6 GHz CPU and were taken in single-threaded mode.

8.2. Volume Meshes

Discretizations of the Laplacian for general volume meshes are less common, leading to fewer possibilities for comparison than in the surface case. For tetrahedral meshes, we compare the Diamond Laplacian to the well known volume cotangent Laplacian, from now on called *Primal Laplacian*, as well as to the *Dual Laplacian* that was introduced in [AHKSH20].

For comparisons on general polyhedra, we generalize the polygon Laplacian of Bunge et al. [BHKB20] to volumetric meshes. To this end, we insert virtual vertices into faces and cells (minimizing squared areas/volumes as described in Section 6.3), construct gradient, divergence, and Laplacian on the virtually refined tetrahedral mesh using the primal/cotan discretization [Cra19], and employ the sandwiching of Equation (30).

Moreover, we compare to the two major DDFV methods *CeVe* [Her09] and *CeVeFE* [CH11]. Both DDFV approaches introduce a significant amount of additional degrees of freedom, with *CeVe* working on values per vertices and per cells, and *CeVeFE* with values at vertices, cells, faces, and edges. To compare to our approach, mapping values on vertices to values on vertices, we restrict their results to the per-vertex values and ignore the additional values. This restricts the comparison to the Poisson tests – how to compare eigenmodes remains unclear.

The different types of polyhedral tessellations used in the following evaluation are depicted in Figure 10.

	# Vertices	Diamond			[AW11]			[BHKB20]			[dGBD20]		
Triangles	92k	1198k	2630	59	645k	1491	40	645k	1592	41	645k	1967	41
Quads	99k	2064k	3273	68	884k	1659	42	884k	1574	41	884k	2014	42
Hexagons	82k	3030k	3357	87	1064k	1796	40	1064k	2617	42	1064k	2162	41
Concave	130k	4149k	5049	112	1558k	2391	54	1558k	2450	55	1558k	2296	54

Table 1: Timings for solving Poisson system on polygonal meshes using sparse Cholesky factorization, listing the number of mesh vertices and, for each method, the number of non-zero matrix entries and the time for factorization and back-substitution (in ms).

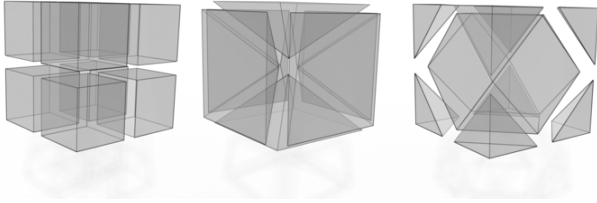


Figure 10: The types of non-simplicial polyhedral meshes used for evaluation in the volumetric case. They will be referred to as Hexahedra (left), Pyramids (center), and Truncated (right).

Poisson Problems As in the surface case, we analyze the convergence behavior for Poisson systems $\mathbf{L}\mathbf{u} = \mathbf{M}\mathbf{b}$, with \mathbf{b} being the Laplacian of the Franke test function

$$F_{3D}(x, y, z) = \frac{3}{4}e^{-\frac{(9x-2)^2+(9y-2)^2+(9z-2)^2}{4}} + \frac{3}{4}e^{-\frac{(9x+1)^2}{49} - \frac{9y+1}{10} - \frac{9z+1}{10}} + \frac{1}{2}e^{-\frac{(9x-7)^2+(9y-3)^2+(9z-5)^2}{4}} - \frac{1}{5}e^{-(9x-4)^2 - (9y-7)^2 - (9z-5)^2}$$

We solve the Poisson system on different tessellations of the 3D unit cube, fixing boundary vertices to the values of F_{3D} . As shown in Figure 11, the Diamond Laplacian has the expected convergence behavior and yields lower errors than other methods.

Laplacian Eigenmodes When solving the eigenvalue problem of the Laplacian, better known as Helmholtz equation, on a 3-ball \mathcal{B}^3

$$-\Delta u = \lambda u \text{ in } \mathcal{B}^3 \quad \text{with } u = 0 \text{ on } \partial\mathcal{B}^3,$$

its discrete solution can analytically be expressed in terms of the spherical Bessel functions. Therefore, given a stiffness matrix \mathbf{L} and mass matrix \mathbf{M} for a polyhedral tessellation of \mathcal{B}^3 , the discrete analogue of the Helmholtz equation can be formulated as the generalized eigenvalue problem $\mathbf{L}\mathbf{u} = \lambda\mathbf{M}\mathbf{u}$. The plots in Figures 12 and 13 show that the Diamond Laplacian successfully recovers the desired eigenvalues, rather independent of the tessellation. In particular on non-uniform adaptive tetrahedral meshes (Figure 13) our operator is considerably more accurate than the primal and dual tetrahedral Laplacians.

Timings and Sparsity The additional cell and face vertices that we (virtually) insert to construct the minimal diamonds lead to a

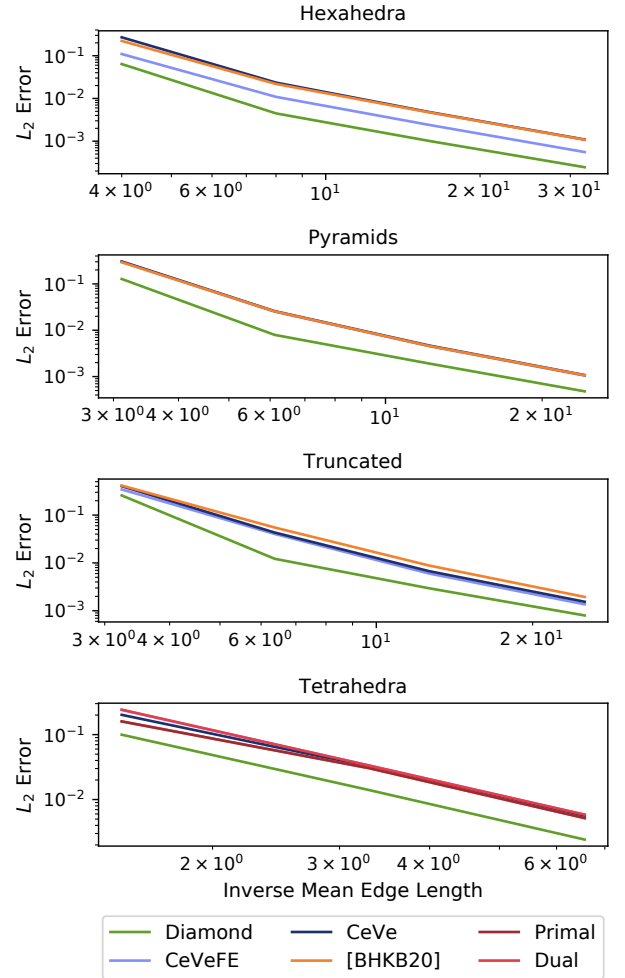


Figure 11: L_2 error in log-log scale of the Poisson system solved for Franke's test function on different tessellations of the unit cube.

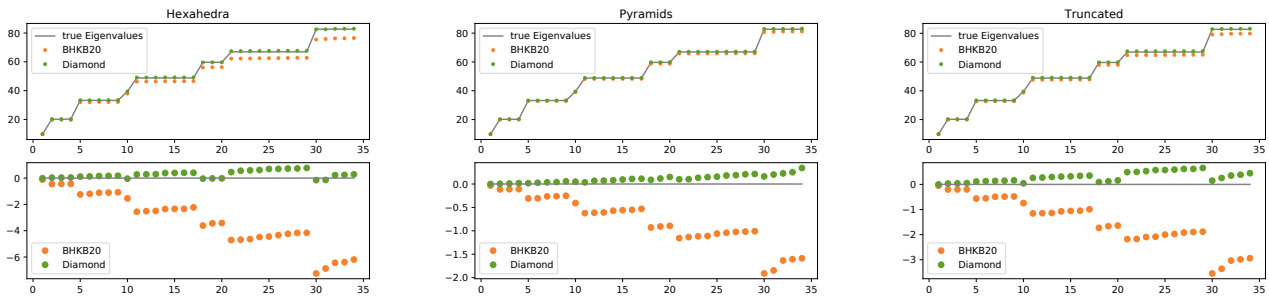


Figure 12: The 34 smallest eigenvalues of the Laplacian, computed on different polyhedral tessellations of the unit ball. The top row shows the obtained eigenvalues, the bottom row the deviation from the true values. In all cases, our operator is more accurate than [BHKB20].

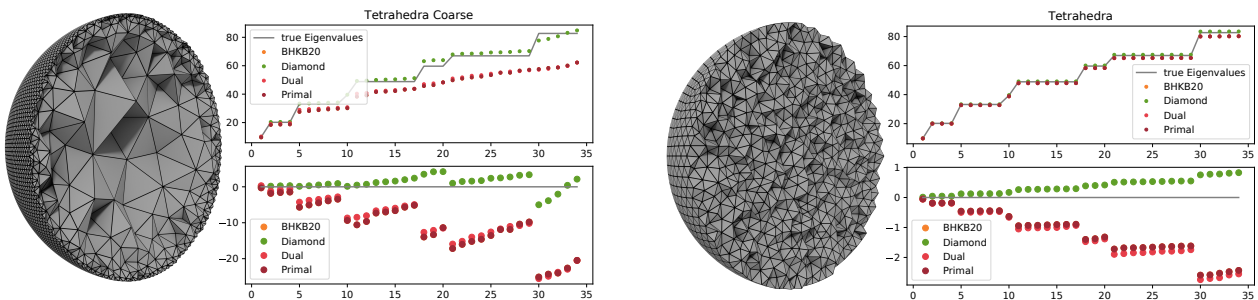


Figure 13: The 34 smallest eigenvalues of the Laplacian, computed on uniform and adaptive tetrahedral tessellations of the unit ball. For tetrahedral meshes the generalization of Bunge et al. [BHKB20] reduces to the Primal cotan operator, therefore yielding the same results. The Diamond Laplacian is not affected by adaptive tessellations and achieves a higher accuracy.

higher density in the Diamond stiffness matrix compared to both tetrahedral operators presented in [AHKSH20] and the polyhedral generalization of [BHKB20]. Table 2 compares the matrix density for the different Laplace discretizations and the time for solving the resulting Poisson system, again using the supernodal LLT solver of CHOLMOD [CDHR08]. As expected, the Diamond Laplacian is more expensive than the Primal and Dual tetrahedral Laplacians and the polyhedral version of [BHKB20]. Compared to the volumetric DDFV methods, timings are comparable to CeVe [Her09], which however suffers from spurious modes, and it is significantly faster than CeVeFE DDFV [CH11], while at the same time being more accurate than all other methods.

9. Conclusion

We improve on DDFV methods by generalizing the 2D DDFV formulation to surface meshes immersed in 3D (*intrinsic gradients*) and by providing a formulation for polyhedral meshes (*ring of minimal diamonds*) that avoids the spurious modes of CeVe [Her09] and is considerably simpler than CeVeFE [CH11]. We combine this with a generalization of the approach of Bunge et al. [BHKB20] from polygon meshes to polyhedral meshes and use the resulting *double prolongation* to remove the additional degrees of freedom of dual cell and face vertices from the improved DDFV operators. The

Diamond Laplacian provides, to the best of our knowledge, the *first simple* Laplacian for general polyhedral meshes that maps values at vertices to values at vertices while having the appropriate kernel, linear precision, and the desired semi-definiteness. The source code for the polygonal and polyhedral Diamond Laplace is available at <https://github.com/mbotsch/polyLaplace>.

In extensive numerical evaluations on prototypical geometry processing applications we compare the Diamond Laplacian to seven existing methods from the graphics and DDFV communities as well as to the unpublished generalization of [BHKB20] to polyhedral meshes. In almost all experiments the Diamond Laplacian is superior to all its competitors, otherwise it is placed second. In contrast to existing polygon Laplacians, it does not reduce to the cotan formulation on triangle meshes. It is a viable alternative also for pure simplicial meshes, as it provides more accurate results in particular if the gradient operator is involved.

The price for generality and accuracy is a higher number of non-zero elements, leading to a slight increase in computation time. We believe that this is a useful trade-off in graphics and geometric modeling, where meshes are mostly processed without altering them. We cannot resist to make the obvious remark: Diamonds \diamond are attractive but somewhat expensive.

	# Vert.	Diamond			[BHKB20]			CeVeFE			CeVe		
Tetrahedra	4.4k	127k	378	5	63k	122	4	1664k	4382	95	380k	1017	23
Hexahedra	36k	2587k	4487	110	912k	1670	43	1497k	20813	487	736k	4727	133
Pyramids	23k	1037k	1255	45	398k	880	24	5021k	22380	479	1231k	4890	129
Truncated	19k	1988k	1652	44	476k	989	22	2461k	5051	104	580k	1714	41

Table 2: Timings for solving Poisson system on polyhedral meshes using sparse Cholesky factorization, listing the number of mesh vertices and, for each method, the number of non-zero matrix entries and the time for factorization and back-substitution (in ms).

Acknowledgements

The authors are grateful to Hendrik Meyer for his help in rendering the figures, to Fernando de Goes for providing an extensive set of test meshes, and to Philipp Herholz for his help with the volumetric dual Laplacian. Marc Alexa acknowledges support by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – The Berlin Mathematics Research Center MATH+ (EXC-2046/1, project ID: 390685689).

References

- [ABH13] ANDREIANOV B., BENDAHMANE M., HUBERT F.: On 3D DDFV discretization of gradient and divergence operators. II. Discrete functional analysis tools and applications to degenerate parabolic problems. *Computational Methods in Applied Mathematics* 13, 4 (June 2013), pp. 369–410. 6, 8
- [ABHK12] ANDREIANOV B., BENDAHMANE M., HUBERT F., KRELL S.: On 3D DDFV discretization of gradient and divergence operators. I. Meshing, operators and discrete duality. *IMA Journal of Numerical Analysis* 32, 4 (2012), 1574–1603. 7
- [AHKSH20] ALEXA M., HERHOLZ P., KOHLBRENNER M., SORKINE-HORNUNG O.: Properties of Laplace operators for tetrahedral meshes. *Computer Graphics Forum* 39, 5 (2020). 1, 2, 8, 10, 12
- [Ale20] ALEXA M.: Conforming weighted Delaunay triangulations. *ACM Transactions on Graphics* 39, 6 (2020). 3
- [Aur87] AURENHAMMER F.: A criterion for the affine equivalence of cell complexes in \mathbb{R}^d and convex polyhedra in \mathbb{R}^{d+1} . *Discrete Comput. Geom.* 2, 1 (1987), 49–64. 3
- [AW11] ALEXA M., WARDETZKY M.: Discrete Laplacians on general polygonal meshes. *ACM Transactions on Graphics* 30, 4 (2011), 102:1–102:10. 1, 2, 6, 8, 9, 10, 11
- [BHKB20] BUNGE A., HERHOLZ P., KAZHDAN M., BOTSCH M.: Polygon Laplacian made simple. *Computer Graphics Forum* 39, 2 (2020), 303–313. 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
- [BLS05] BREZZI F., LIPNIKOV K., SIMONCINI V.: A family of mimetic finite difference methods on polygonal and polyhedral meshes. *Mathematical Models and Methods in Applied Sciences* 15, 10 (2005), 1533–1551. 2
- [CDHR08] CHEN Y., DAVIS T. A., HAGER W. W., RAJAMANICKAM S.: Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Trans. Math. Softw.* 35, 3 (2008). 10, 12
- [CH11] COUDIÈRE Y., HUBERT F.: A 3D discrete duality finite volume method for nonlinear elliptic equations. *SIAM J. Sci. Comput.* 33, 4 (2011), 1739–1764. 3, 4, 5, 7, 10, 12
- [Cra19] CRANE K.: The n -dimensional cotangent formula. <https://www.cs.cmu.edu/~kmc Crane/Projects/Other/nDCotanFormula.pdf>, 2019. 1, 2, 10
- [CWW13] CRANE K., WEISCHEDEL C., WARDETZKY M.: Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Transactions on Graphics* 32, 5 (2013), 152:1–152:11. 9
- [dGBD20] DE GOES F., BUTTS A., DESBRUN M.: Discrete differential operators on polygonal meshes. *ACM Transactions on Graphics* 39, 4 (2020). 1, 2, 6, 8, 9, 10, 11
- [dGDM16] DE GOES F., DESBRUN M., MEYER M., DEROSE T.: Subdivision exterior calculus for geometry processing. *ACM Transactions on Graphics* 35, 4 (2016), 133:1–133:11. 2, 9
- [DLN07] DAI K., LIU G., NGUYEN T.: An n -sided polygonal smoothed finite element method (nSFEM) for solid mechanics. *Finite Elements in Analysis and Design* 43, 11 (2007), 847–860. 2
- [DMSB99] DESBRUN M., MEYER M., SCHRÖDER P., BARR A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of ACM SIGGRAPH* (1999), pp. 317–324. 1, 2
- [DO05] DOMELEVO K., OMNÈS P.: A finite volume method for the Laplace equation on almost arbitrary two-dimensional grids. *Math. Model. Numer. Anal. (M2AN)* 39, 6 (2005), 1203–1249. 3, 4, 5, 7
- [Dro14] DRONIOU J.: Finite volume schemes for diffusion equations: Introduction to and review of modern methods. *Mathematical Models and Methods in Applied Sciences* 24, 08 (May 2014), 1575–1619. 3
- [dVBM13] DA VEIGA L. B., BREZZI F., MARINI L. D.: Virtual elements for linear elasticity problems. *SIAM J. Numer. Anal.* 51, 2 (2013), 794–812. 2
- [Dzi88] DZIUK G.: *Finite Elements for the Beltrami operator on arbitrary surfaces*. Springer Berlin Heidelberg, 1988, pp. 142–155. 1, 2
- [Fra79] FRANKE R.: *A critical comparison of some methods for interpolation of scattered data*. Tech. rep., Naval Postgraduate School, 1979. 9
- [Her00] HERMELINE F.: A finite volume method for the approximation of diffusion operators on distorted meshes. *J. Comput. Phys.* 160, 2 (2000), 481–499. 3, 4, 5
- [Her09] HERMELINE F.: A finite volume method for approximating 3D diffusion operators on general meshes. *J. Comput. Phys.* 228, 16 (2009), 5763–5786. 3, 4, 5, 6, 7, 10, 12
- [Hir03] HIRANI A. N.: *Discrete exterior calculus*. PhD thesis, California Institute of Technology, 2003. 3
- [HKA15] HERHOLZ P., KYPRIANIDIS J. E., ALEXA M.: Perfect Laplacians for polygon meshes. *Computer Graphics Forum* 34, 5 (2015), 211–218. 1, 2
- [HS08] HORMANN K., SUKUMAR N.: Maximum entropy coordinates for arbitrary polytopes. *Computer Graphics Forum* 27, 5 (2008), 1513–1520. 2
- [HS17] HORMANN K., SUKUMAR N.: *Generalized Barycentric Coordinates in Computer Graphics and Computational Mechanics*. CRC Press, 2017. 2
- [KBT17] KOSCHIER D., BENDER J., THUEREY N.: Robust extended finite elements for complex cutting of deformables. *ACM Transactions on Graphics* 36, 4 (2017). 1

- [MDSB03] MEYER M., DESBRUN M., SCHRÖDER P., BARR A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, Hege H.-C., Polthier K., (Eds.). Springer-Verlag, 2003, pp. 35–57. [1](#), [2](#)
- [MKB*08] MARTIN S., KAUFMANN P., BOTSCH M., WICKE M., GROSS M.: Polyhedral finite elements using harmonic basis functions. *Computer Graphics Forum* 27, 5 (2008), 1521–1529. [1](#), [2](#)
- [MRS14] MANZINI G., RUSSO A., SUKUMAR: New perspectives on polygonal and polyhedral finite element methods. *Mathematical Methods Models and Sciences* 24, 8 (2014), 1665–1699. [1](#), [2](#)
- [PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experim. Math.* 2 (1993), 15–36. [1](#), [2](#), [3](#)
- [SCV14] SOLOMON J., CRANE K., VOUGA E.: Laplace-Beltrami: The Swiss army knife of geometry processing. Symposium on Geometry Processing Graduate School (Cardiff, UK, 2014), 2014. [1](#)
- [SDG*19] SCHNEIDER T., DUMAS J., GAO X., BOTSCH M., PANOZZO D., ZORIN D.: Poly-spline finite-element method. *ACM Transactions on Graphics* 38, 3 (2019). [1](#), [2](#)
- [SSC19] SHARP N., SOLIMAN Y., CRANE K.: The vector heat method. *ACM Transactions on Graphics* 38, 3 (2019). [2](#), [3](#)
- [TWZZ09] TANG X.-H., WU S.-C., ZHENG C., ZHANG J.-H.: A novel virtual node method for polygonal elements. *Applied Mathematics and Mechanics* 30 (2009). [2](#)
- [WBG07] WICKE M., BOTSCH M., GROSS M.: A finite element method on convex polyhedra. *Computer Graphics Forum* 26, 3 (2007), 355–364. [1](#), [2](#)
- [WMKG07] WARDETZKY M., MATHUR S., KÄLBERER F., GRINSPUN E.: Discrete Laplace operators: No free lunch. In *Proceedings of Eurographics Symposium on Geometry Processing* (2007), pp. 33–37. [2](#), [7](#)