Designing 2D and 3D Non-Orthogonal Frame Fields

David Desobry, Yoann Coudert-Osmont, Etienne Corman, Nicolas Ray, Dmitry Sokolov

Designing 2D and 3D Non-Orthogonal Frame Fields

David Desobry, Yoann Coudert-Osmont, Etienne Corman, Nicolas Ray, Dmitry Sokolov

Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

Abstract

We present a method for direction field design on surface and volumetric meshes supporting non-orthogonality. Our approach is a generalization of the representation of 3D cross fields in spherical harmonic basis. As such it induces a geometrically meaningful measure of smoothness, allows orthogonality control by a simple parameter and enables orientation constraints of a single direction. To the best of our knowledge this is the first work to propose non-orthogonal 3D frame field design. We demonstrate the applicability of our method to generate anisotropic quadrangular and hexahedral meshes which are particularly useful for remeshing CAD models.

Keywords: Geometry processing, vector field design, quadrilateral remeshing, hexahedral remeshing

Introduction

Designing tangent vector fields is a fundamental component of geometry processing. If it is predominantly used for surface remeshing [1] it has also applications in stylized rendering [2], BRDF modification [3], fabrication [4] and sketch based modeling [5, 6]. Vector field design is often a single step of a more involved pipeline. Since each application has its own needs it is important to provide robust and flexible algorithms.

Typical applications like parametrization require to design *N-Directions* vector fields: 2D frame fields with 2-directions for quad-meshes [1] and 3D frame fields with 3-directions on a volumetric domain for hex-meshing [7]. Early works only allowed the design of orthogonal fields, often called cross fields, to guide the orientation of the quads or hexes. However, the orthogonality constraint is often too restrictive and does not offer enough design freedom.

Meshing of CAD models is a typical example of how limiting orthogonality is. In these models it is common to encounter chamfers and sharp corners. Cross fields simply cannot meet these design constraints as demonstrated in Figure 4 and Figure 11. Many articles propose custom solutions to stray away from the dull isotropic-right-angled frames, by creating a new representation [8], by deforming the mesh [9] or by modifying the metric tensor [10]. When doing so, three challenges must be overcome.

First they must design a new representation or framework in which to define their vector fields. This representation is highly impacted by the dimensionality of the targeted space. In many cases, representation well-suited for surfaces, like the representation vector for 2D cross field with complex numbers [11, 12, 8], cannot be directly extended to volumetric meshes. Instead a completely different approach was introduced taking advantages of the symmetries of the spherical harmonic basis [13]. This situation leads to large body of literature declining vector field design in various setups: surface or volumetric meshes, isotropic or anisotropic, orthogonal or non-orthogonal, sometimes cube symmetry or higher order symmetry... with no or little overlaps in these methods.

A common design requirement is to find the smoothest vector field interpolating between constraints. So once the representation is fixed the second challenge is to build a geometrically meaningful measure of smoothness of the field. This is of key importance as a non-geometrically based measure might lead to non-intuitive vector field design.

The third challenge is to be able to impose user defined constraints within the chosen representation. Such constraints may be as-simple-as a hard constraint on the vector field, more subtle like fixing one direction of the frame and let the other directions rotate freely or allow control over the orthogonality of the frame.

Contributions. In this paper, we represent sets of directions (or *direction fields*) as a polynomial function on a circle or a sphere (depending on the dimensionality). This representation is directly inspired by the spherical harmonic representation of 3D cross fields introduced in [13]. By designing polynomials, we can choose the number of symmetries and the shape of the frame as illustrated in Figures 2 and 8. This representation allows rotation of each direction independently which creates a parametric space of anisotropic non-orthogonal frames in any dimension. Unlike previous methods, ours addresses efficiently the three challenges:

1. The field representation is readily working for surfaces and volumetric meshes;
2. It offers a geometrically meaningful notion of smoothness;
3. Any type of direction fields can be designed and a simple parameter allows control over the orthogonality.

The rest of the paper will be organized as follow. Section 1 gives a general "recipe" to design direction fields of any dimension, thanks to a representation based on a sum of atomic polynomials. This theoretical framework is then applied to 2D frame field design in Section 2 and to generation of 3D frame fields in Section 3. In these sections, we will demonstrate the usefulness of our representation for quad-meshing and hex-meshing.

Related work

Frame fields on surfaces and volumes have a wide range of applications in computer graphics. In this paper, we will focus on global parametrization. Cross fields and more generally frame fields have been largely used in two dimension for quadrangular remeshing [1], design of 6-symmetry vector fields can produce highly regular triangular remeshes [14] and 3D frame fields are a common tool for hexahedral mesh generation [7].

Although symmetric vector fields of two and three dimensions share many applications, their representation are vastly divergent. In fact, many well established 2D vector field design methods cannot be extended to 3D.

Frame field design on surfaces

Cross fields have been explored extensively in the past decade. They were first introduced in computer graphics to place hatches in non photorealistic rendering [2]. They were soon generalized to n-symmetry vector fields [15, 11] by representing a cross as a n^{th} root of a complex number. The smoothness of a field is simply obtained by measuring the similarity of pairs of adjacent frames. This leads to a simple quadratic optimization problem allowing alignment constraints [11, 12] or singularity constraints [16]. Cross field design surveys can be found in [17] and [18].

Liu *et al.* [19] proposed a generalization of cross fields by allowing them to be non-orthogonal. The goal was to generate quad meshes with the possibility for each element to stray away from a perfect square. However, this method requires to compute an explicit matching between the vector set to disambiguate the symmetry. More recent works generate anisotropic and non-orthogonal frame fields by deforming the mesh or designing a new metric [9, 10]. Processing metric tensors is a costly which implies to solve an intricate optimization problem with multiple objectives. These methods could be generalized to volumetric meshes but at great computational cost since symmetric 3-tensors have twice as much degrees of freedom as 2-tensors.

Closer to our work is the representation of n-symmetry, non-orthogonal vectors as a root of a complex polynomial [8, 20]. The authors remarked that a set of 2D directions can be seen as the set of roots of a complex polynomial. In this setup, a deformation of the frame amounts for a modification of the polynomial coefficients. Therefore, they use polynomial coefficients to uniquely represent each frame. Like the representation described in the present paper, the complex polynomial representation is invariant by reordering of the direction set. However, this choice of variables blurs the link with the geometry of the frame and create two limitations not present in our work. First, in order to find smooth frame fields, a smoothness energy is applied to the polynomial coefficients. Although the smoothness of the field implies the smoothness of the coefficients, this notion of smoothness is *not* geometrically rooted and may introduce bias. An in depth discussion is provided in Section 2.5. Second, this representation does not generalize to higher dimensions.

3D frame fields and volumetric representation

The 3D cross fields, sometimes referred to as *octahedral fields*, are commonly used for hexahedral meshing [14]. A single frame consists of three orthogonal vectors and their opposite, thus a 3D cross has the same symmetries than a cube. Huang *et al.* [13] proposed to represent each individual frame by the spherical function $(x, y, z) \mapsto x^4 + y^4 + z^4$ whose level sets are cube-like. While seemingly different than the two dimensional case, Ray *et al.* [21] noted that it can be understood has a generalization of 2D complex representation vector. In this paper, we will build upon this observation.

As for the 2D case, generalization to n-symmetry 3D vector fields [22] and to anisotropic orthogonal frame field [23, 24] have been proposed by studying the properties of the spherical harmonic basis. However, to the best of our knowledge, there is no existing work on non-orthogonal 3D frame field design. Moreover, a representation unifying 2D and 3D direction fields allowing independent direction constraints is still missing.

1. Direction field of any dimension as spherical polynomials

Given a simplicial n-complex $\mathcal{F}$, we consider each facet $f \in \mathcal{F}$ as a discrete tangent space. Consequently, in this paper tangent vector fields will be piecewise constant. To be more specific, on a triangle mesh our tangent vector fields are defined on faces of triangles and on a tetrahedral mesh tangent vector fields are constant per tetrahedron. This is a standard setup for global parametrization and structured meshing [1, 7]. We denote $\mathcal{E}$ the set of pairs of adjacent facets; namely $i, j \in \mathcal{F}$ are adjacent if and only if $(i, j) \in \mathcal{E}$. For imposing boundary conditions, we will need $\partial\mathcal{F}$, the set of boundary facets.

To ease the exposition, we restrict ourselves to *flat* domains of $\mathbb{R}^n$. All the concepts introduced in this paper can be transposed for *curved* surfaces. One only has to define a notion of parallel transport as done in previous papers [8, 11].

1.1. Representation of a single frame

We will call a *N-Direction set* an unordered set of N *unit* tangent vectors and their opposite; namely $\{u_1, -u_1, u_2, -u_2, \ldots, u_N, -u_N\}$. A *N-Direction field* is an assignment of a direction set per facet. Our goal is to find the smoothest N-Direction field on a triangle or tetrahedral mesh satisfying user-defined direction constrains. As a direction set is unordered, measuring the similarity of two sets can be rather painful as it requires to correctly match each set. Therefore, we need a representation of a single direction set that is invariant to sign change and reordering. Inspired by the 2D and 3D cross fields representation introduced in [13, 21], we will identify a direction to a spherical polynomial whose shape and symmetry is identical.

As a direction is a set of two opposite vectors, it must be represented by an *even degree* polynomial in order to match the symmetry. Indeed, given a unit vector u, the spherical polynomial $p(s) = \langle u, s\rangle^{2k}$ for $k \geq 1$ integer is independent of the sign of u. To represent the N-Direction set $\{u_i, -u_i \mid i = 1 \ldots N\}$, we simply sum over these atomic polynomials:

$$p(s) = \sum_{i=1}^{N} \langle u_i, s\rangle^{2k}. \tag{1}$$

Figure 1 illustrates the polynomial representing a unique direction in polar and spherical coordinates. Most interesting to parametrization applications are the 2D and 3D frames illustrated in Figures 2 and 8 for $k = 2$.

Dealing with polar or spherical polynomial may seem difficult but it is made simple by decomposing it into canonical orthonormal bases.

1.2. Tangent space of a Direction set

Considering a N-Direction set for all possible unit vectors spans a polynomial subspace denoted $\mathbb{P}_{2k,N}$:

$$\mathbb{P}_{2k,N} := \left\{ \sum_{i=1}^{N} \langle u_i, s\rangle^{2k} \quad | \quad \forall u_i \in \mathbb{S}^n \right\}.$$

The function space $\mathbb{P}_{2k,N}$ contains only *homogeneous* polynomials of degree $2k$. As such, it is a finite dimensional space and is spanned by a finite number of orthogonal functions Y_ℓ. Later, we will make use of the Fourier basis for 2D direction fields and the Spherical Harmonic basis for 3D direction fields. Therefore,

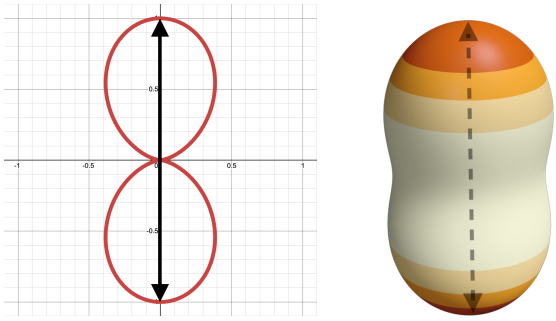
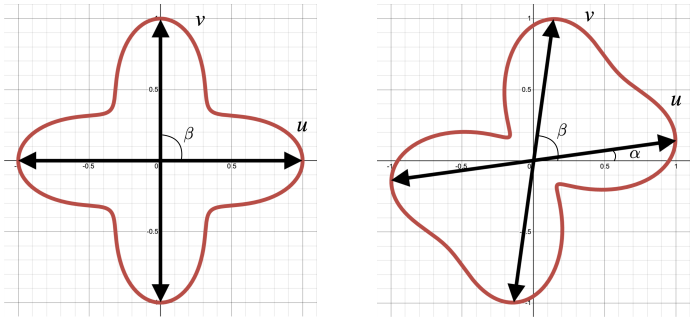


Figure 1: A unique direction (black arrows) represented by a polar (left) and spherical (right) polynomial of degree 2. The value of the polynomial at each point of the circle is represented as a variation of the radius in polar coordinates.



(a) Orthogonal frame (b) Non-orthogonal deformation

Figure 2: A 2-Direction set $\{u, -u, v, -v\}$ (black arrows) represented by a polar polynomial of degree four.

any N-Direction set $U = \{u_i, -u_i \mid i = 1 \dots N\}$ is uniquely represented by the vector of coefficients $c_\ell(U), \ell = 1, \dots, L$ in the decomposition in the orthogonal basis:

$$p(s) = \sum_{\ell=1}^{L} c_\ell(U) Y_\ell(s).$$

Crucially, the coefficients $c_\ell(U)$, like the polynomial $p(s)$, are independent to reordering and sign changes of the vectors in the set U. They only depend on the positions of the vectors as a whole. This representation greatly simplifies the computation of a distance between two N-Direction sets as it suffices to compute a distance between their polynomial representations. The representation polynomials have the same shapes as the sets (see Figures 2 and 8 for an example of frame field) so this distance has a clear geometrical meaning.

Using the fact that the basis Y_ℓ is orthogonal, the distance between a polynomial p_1 representing the set of directions U_1 and p_2 representing the set of directions U_2 is simply the norm of the difference of the coefficients c:

$$\begin{aligned} \|p_1 - p_2\|_2^2 &= \int_{\mathbb{S}^n} (p_1(s) - p_2(s))^2 \, \mathrm{d}s \\ &= \sum_{\ell=1}^{L} (c_\ell(U_1) - c_\ell(U_2))^2 . \end{aligned}$$

1.3. Piecewise constant polynomial PolyVector

A N-Direction field assigns a set U_i per facet $i \in \mathcal{F}$. Our representation replaces the sets U_i by the coefficients $c_\ell(U_i), \ell = 1, \dots, L$. Our goal is to design smooth N-Direction fields. A N-Direction field is considered perfectly smooth across an edge if its polynomial, and thus its coefficients c, are equal. Following this intuition, we introduce a smoothness energy, akin to a discrete Dirichlet energy, by accumulating the distance between adjacent frames:

$$E_s(U) = \sum_{(ij)\in\mathcal{E}} \sum_{\ell=1}^{L} \left(c_\ell(U_i) - c_\ell(U_j)\right)^2 . \tag{2}$$

Designing smooth direction fields interpolating constraints amounts for minimizing the smoothness energy E_s with direction constraints directly imposed on the vector set U. Making design problems much more accessible than before.

1.4. Degree of the representation polynomial

The degree of the polynomial is a parameter that influences the shape of the frame and the complexity of the optimization problem. There is a lower bound for the polynomial degree. In [22], the authors studied spherical polynomial for representing N-Direction sets evenly distributed in 3D. They remarked that a polynomial of degree at least 4 was needed to represent 3 orthogonal directions and a 6 degree polynomial was necessary for 6 directions forming the vertices of a regular dodecahedron. Moreover, in order for the N directions of a N-Direction set to rotate independently the polynomial must be of degree high enough to represent these variations. However, the complexity of the coefficients c increases with the polynomial degree thus the minimization of the smoothness energy in Eq. 2 becomes more challenging .

In the rest of the paper, we focus on the special case of frame fields of 2 and 3 dimensions. We will choose the lowest acceptable polynomial degree, in the present case degree 4 or $k = 2$ in Eq.(1). Figures 2b and 8b shows that polynomials of degree 4 are sufficient to catch deformations of a 2D or 3D frame.

2. 2D frame fields

In this section, we focus on frame fields (or 2-Direction fields) on 2D domains. First introduced in [9], they are extremely useful for quad-meshing. For 2D domain the representation polynomial p is defined on a circle and the set of admissible tangent frames $\mathbb{P}_{2k,2}$ is decomposed on the Fourier basis. We will see that with this choice of basis, and when the two directions are orthogonal, our representation is exactly equal to the complex representation of cross fields used in many articles [15, 11, 12, 8].

2.1. One direction as a polynomial of degree 4

As noted in [21], 2D cross fields can be represented by the polar polynomial of degree four: $p(s) = s_1^4 + s_2^4$. In fact, this is the polynomial of the lowest degree presenting the same symmetry as a cross. For this reason, we will use degree 4 polynomials to represent our frame field.

A 2D frame is the set of two directions $\{u, -u, v, -v\}$ parametrized respectively by the angles α and β, such that $u = (\cos\alpha, \sin\alpha)$ and $v = (\cos\beta, \sin\beta)$. These notations are summarized by Figure 2. Following Eq. (1) our representing polynomial of degree four reads as the sum:

$$p(s) = \langle u, s\rangle^4 + \langle v, s\rangle^4. \tag{3}$$

Figure 2 gives an illustration of the shape of an orthogonal frame (Fig. 2a) and a non-orthogonal frame as represented by the polar polynomial in Eq. (3).

2.2. Fourier basis

The representing polynomial in Eq. (3) is defined on a unit circle. It seems natural to use the Fourier basis as an orthogonal basis to decompose this periodic signal. We will note $c_{\ell,1}(\alpha,\beta)$ the coefficients corresponding to the basis functions $\cos(\ell t)$ and $c_{\ell,2}(\alpha,\beta)$ the coefficients corresponding to the basis functions $\sin(\ell t)$ where $\ell \in \mathbb{N}$ is the frequency index.

The sine/cosine basis appears naturally when expressing $s = (\cos t, \sin t)$ in the polar coordinate system. The direction polynomial in u simply reads:

$$\begin{aligned}\langle u, s\rangle^4 &= |\cos t \cos\alpha + \sin t \sin\alpha|^4 \\ &= \frac{3}{4} + \frac{1}{2}\cos(2t)\cos(2\alpha) + \frac{1}{2}\sin(2t)\sin(2\alpha) \\ &+ \frac{1}{8}\cos(4t)\cos(4\alpha) + \frac{1}{8}\sin(4t)\sin(4\alpha)\end{aligned}$$

Thus the coefficients of the decomposition of polynomial frame boils down to:

$$\left|\begin{array}{l} c_{2,1}(\alpha,\beta) = 4\cos(2\alpha) + 4\cos(2\beta) \\ c_{2,2}(\alpha,\beta) = 4\sin(2\alpha) + 4\sin(2\beta) \\ c_{4,1}(\alpha,\beta) = \cos(4\alpha) + \cos(4\beta) \\ c_{4,2}(\alpha,\beta) = \sin(4\alpha) + \sin(4\beta) \end{array}\right. \tag{4}$$

Our frame is represented by only four coefficients in frequencies $\ell = 2$ and $\ell = 4$. Crucially, the frame is orthogonal (*i.e.* $\beta = \alpha + \pi/2$) if and only if the coefficients of second order are zero. In this case, the frame is solely represented by the vector $(\cos(4\alpha), \sin(4\alpha))$, thus we recover the classic complex representation of cross fields [15, 11].

2.3. Designing 2D frame fields for quad meshing

In this section, we will set up an optimization problem in order to find the smoothest frame field interpolating between user prescribed direction constraints. A parameter will allow control over the orthogonality of the field.

Variables. To ease the resolution of the optimization problem, we will not directly use the angle α,β or the vector u, v as variables. Instead, we use the double angle vectors $\bar{u} = (\cos(2\alpha), \sin(2\alpha))$ and $\bar{v} = (\cos(2\beta), \sin(2\beta))$. The vector $\bar{u}$ is mapped to either u or $-u$ so the change of variables is valid. With these new variables the coefficients in Eq. (4) now read:

$$\left|\begin{array}{l} c_{2,1}(\bar{u},\bar{v}) = 4\bar{u}_1 + 4\bar{v}_1 \\ c_{2,2}(\bar{u},\bar{v}) = 4\bar{u}_2 + 4\bar{v}_2 \\ c_{4,1}(\bar{u},\bar{v}) = 2\bar{u}_1^2 + 2\bar{v}_1^2 - 2 \\ c_{4,2}(\bar{u},\bar{v}) = 2\bar{u}_1\bar{u}_2 + 2\bar{v}_1\bar{v}_2 \end{array}\right. \tag{5}$$

Direction constraints. The most common constraint for frame field design is to impose constant frames at certain faces. In our framework, setting these constraints is straightforward : one can compute the angle of each target direction and constraint $\bar{u}$ and $\bar{v}$ to the double angle vectors.

A more interesting constraint, often used for non-orthogonal quad remeshing (see Fig. 5), is to fix only one direction out of two. To do so, one can constrain either $\bar{u}$ or $\bar{v}$ to the desired direction. Note that the choice of $\bar{u}$ or $\bar{v}$ has no impact on the results as the polynomial representation is invariant by reordering of the directions. Constraining either $\bar{u}$ or $\bar{v}$ does not affect the second vector of the frame which can rotate freely.

However, when only one direction is fixed, the second direction can become degenerate if no unit constraint forces it to be non-zero. To avoid such phenomenon, we add, for boundary triangles only, a unit norm energy:

$$E_b(\bar{u},\bar{v}) = \sum_{i\in\partial\mathcal{F}} \left(|\bar{u}_i|^2 - 1\right)^2,$$

where $\bar{u}$ is the unconstrained direction. This energy is weighted by the parameter b.

For our quad meshing results, we constrain the frames at the boundary to have one direction tangent to the boundary edge. If a triangle has two boundary edges then the frame is fully constrained. In introduction, we emphasis the challenge of remeshing models with sharp corners such as in Figures 4 and 11. To catch these geometric features, it is important that the directions are fully constrained where several feature lines meet with acute or obtuse angles.

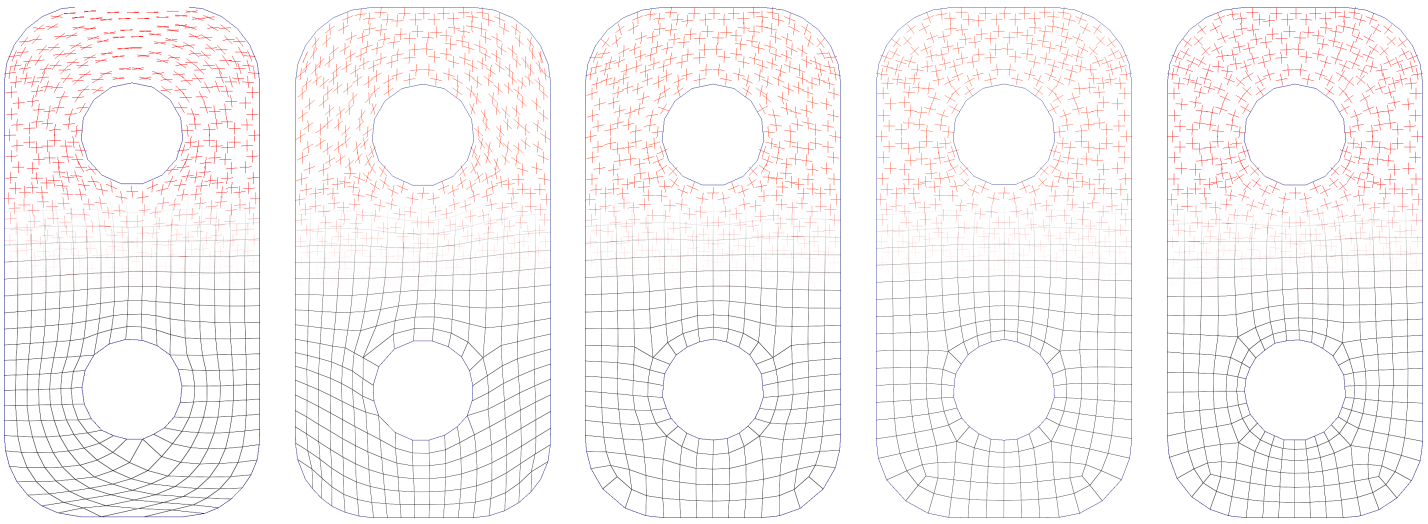
Smoothness energy. Following Section 1.3, the smoothness energy E_s on a 2D frame field depends only on the coefficients of the decomposition Eq. (5). We slightly modify the energy to add a parameter λ balancing the frequency of order 2:

$$\begin{aligned}E_s(\bar{u},\bar{v}) = \lambda \sum_{(ij)\in\mathcal{E}} \sum_{m=1,2} \left(c_{2,m}(\bar{u}_i,\bar{v}_i) - c_{2,m}(\bar{u}_j,\bar{v}_j)\right)^2 \\ + \sum_{(ij)\in\mathcal{E}} \sum_{m=1,2} \left(c_{4,m}(\bar{u}_i,\bar{v}_i) - c_{4,m}(\bar{u}_j,\bar{v}_j)\right)^2.\end{aligned}$$

Orthogonality parameter. As mentioned in Section 2.2, the orthogonality is directly related to second order frequencies. In the smoothness energy, we weight differently the coefficients of order 2, thus putting increasing penalty in these frequencies will lead to frames with constant angle over the entire mesh. If the field have singularities, due the topology of the domain, the frames are forced to be orthogonal. Indeed, a non-orthogonal frame possesses only a π rotation symmetry. So for a singularity to appear, the field must either rotates and change angle at the same time or have index $1/2$ singularities. Both these solutions are far more costly in E_s than having orthogonal frames. Therefore, introducing a simple weight λ on second order coefficients naturally favors orthogonal frames around singularities and by extension, over the entire shape.

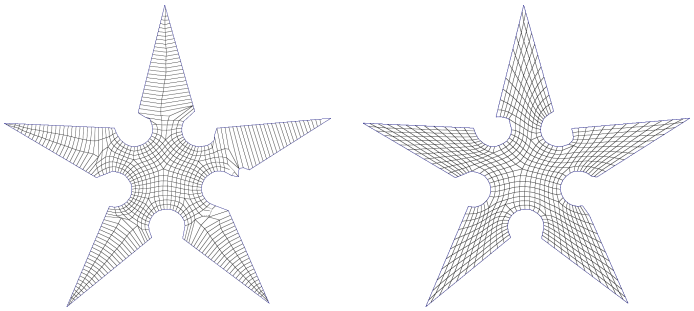
The effect of the orthogonality parameter λ on the frame field and the quad-meshes is illustrated in Figure 3. The value $\lambda = 1$ correspond to the *geometric* notion of smoothness explained in Section 1.3. A value greater than one promotes orthogonality and the frame field becomes increasingly similar to the orthogonal frame field. A λ lower the one gives more freedom to the frame field which can be degenerate. As a result, the singularities are pushed on the boundary of the domain.

The choice of this parameter is dictated by the application and is often the result of a trade-off. On one hand, orthogonal frames lead to quad-meshes with elements close to perfect square but can not deal with sharp corners, omnipresent in CAD models. On the other hand, non-orthogonal frame fields compensate sharp features by introducing shearing and on extreme cases removing singularities from the interior of the domain.



(a) $\lambda = 0.1$ (b) $\lambda = 0.5$ (c) $\lambda = 1$ (d) $\lambda = 1.5$ (e) Orthogonal [25]

Figure 3: Results of our frame field design method for increasing values of the orthogonality parameter λ. Low values (left) push the singularities out of the domain while high values approaches the cross field obtained by Viertel *et al.* [25] (right).



(a) Orthogonal (b) $\lambda = 1.5$

Figure 4: An orthogonal frame field (4a) cannot be used to generate a quad-mesh for a model with sharp corners. Our non-orthogonal frame field (4b) provides a clean solution.

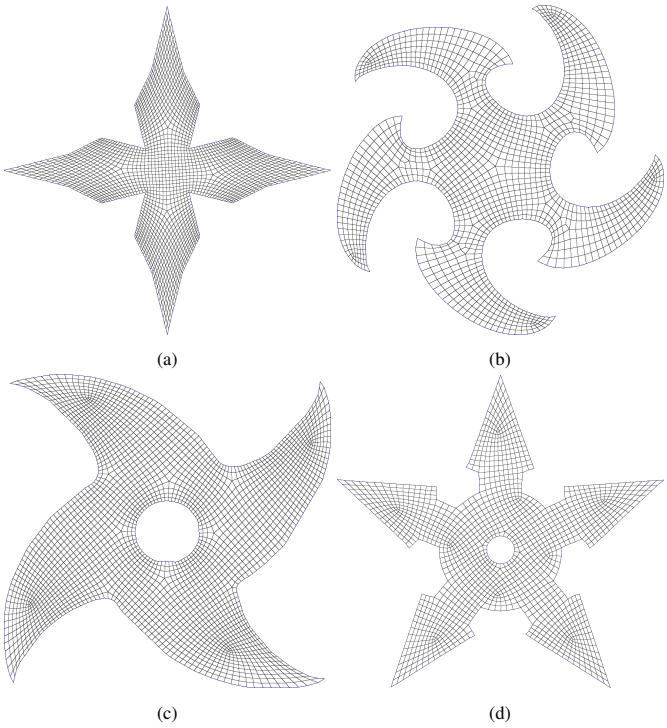
Optimization. Our optimization scheme is based on the MBO algorithm introduced in [25] to optimize singularity position in cross fields. The idea is to alleviate the unit norm constraint on $\bar{u}$ and $\bar{v}$ by alternating between a diffusion step and renormalization step. Namely, the pair of variables $(\bar{u}, \bar{v})_{n+1}$ at iteration $n+1$ are obtained by minimizing the smoothness energy while staying close to the iteration n:

$$(\bar{u}, \bar{v})_{n+1} = \arg\min_{x,y} E_s(x,y) + bE_b(x,y) + \tau \|(x,y) - (\bar{u}, \bar{v})_n\|_2^2, \quad (6)$$

and then renormalizing $(\bar{u}, \bar{v})_{n+1}$ to unit norm. This scheme proves to be the most efficient in term of smoothness of the field and singularity placement. The minimization step is done with LBFGS [26].

2.4. Results

We assess the usability of our frame field design method in the context of quad-meshing. All the frame fields are constrained to have one direction tangent to the boundary of the domain. To avoid zero norm solutions, the energy promoting unit norm at the boundary is weighted by $b = 10$. In all our experiments, the diffusion parameter is set to $\tau = 1/2$. The orthogonality parameter λ is tuned according to our need but the value 1.5 proves to be a good compromise in most cases.



(a) (b) (c) (d)

Figure 5: Our method is able generate nicely shaped quad-meshes by adjusting the parameter $\lambda = 1.5$.

The optimization problem being non-convex, a good initialization is crucial. We initialize with an orthogonal frame field [11] to avoid completely degenerate solutions. The choice of the initial field and its impact on the solution of the optimization are discussed for the three-dimensional case in Section 3.3 but the conclusion also applies in 2D. Our optimization typically needs 10 iterations to converge.

The quad-meshes are extracted from a frame field using the state-of-the-art methods [1, 27].

Quad meshing of CAD models. Generating quad meshing for CAD models is made challenging by the presence of sharp cor-

Fig.	Faces	Init	Opti	Meshing
3	658	0.017	<1	2
4b	1282	0.026	<1	3
5c	9968	0.177	26	13
5a	18430	0.355	66	30
5d	29000	0.611	120	45
5b	36047	0.775	161	46

Table 1: Computation time in second for each step of our method: initialization with the orthogonal frame field from [11], solving our non-convex optimization problem and the quad mesh extraction from [27].

ners: at such places a perfect square cannot fit. Figure 4 illustrates this issue. The orthogonal frame field in Fig. 4a is unable to satisfy the constraints on both side of the sharp angle, thus failing to produce a valid quad-mesh. Our method generate anisotropic quad-mesh (Fig. 4b) naturally fitting the sharp corner constraints. Figure 5 shows a collection of quad-meshes generated from our non-orthogonal frame fields, see how the boundary constraints are nicely satisfied with a small number of singularities while staying away from highly degenerate quads.

Timings. Table 1 shows the timings of each step of our method for each figure of the paper. The most time consuming step is the computation of the non-orthogonal frame field as it requires to solve a non-convex optimization problem. Our algorithm is implemented in C++ but has not been optimized to reach high performances. For instance, we remarked that waiting until convergence of the LBFGS step is often unnecessary and the optimization can be stopped earlier. Also, a Newton method could be used. All the experiments are conducted on a laptop with a four-core, 2.6 GHz Intel Core i5-10400H and 32 GB of RAM.

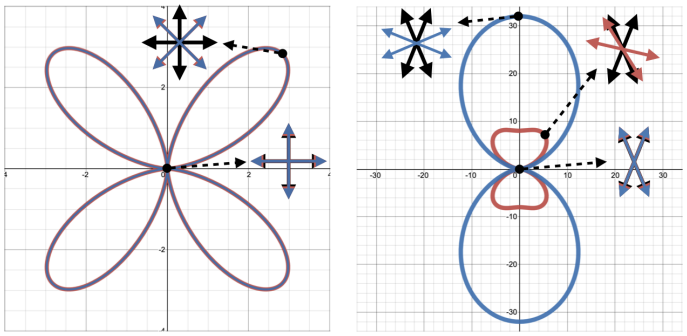
2.5. Comparison with PolyVector field [20]

Diamanti *et al.* [20] uses a representation of frame fields as coefficients of a complex polynomial. Like ours, their representation is independent of the direction set ordering. Moreover, they introduce an energy promoting the *integrability* of the field, *i.e.* such that each direction is the gradient of a function. Like us, they directly use direction vectors as variables in their numerical scheme. Thus, their integrability energy could be immediately added to our optimization.

Distance between frames. Leaving aside numerical optimization aspects, the main difference resides in the smoothness energy. The representation used in Diamanti *et al.* [20] obfuscates the geometrical meaning of the frame representation, leading the authors to seek smoothness of the coefficients of the complex polynomial instead of the smoothness of the field. In our notation, the PolyVector representation of a unit frame boils down to replacing our coefficients c by $\tilde{c}$:

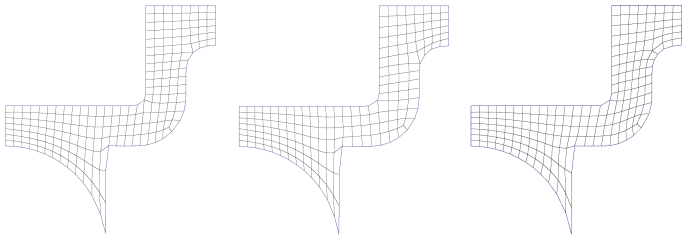
$$\left|\begin{array}{l} \tilde{c}_{2,1}(\alpha,\beta) = 2\cos(2\alpha) + 2\cos(2\beta) \\ \tilde{c}_{2,2}(\alpha,\beta) = 2\sin(2\alpha) + 2\sin(2\beta) \\ \tilde{c}_{4,1}(\alpha,\beta) = 2\cos(2\alpha + 2\beta) \\ \tilde{c}_{4,2}(\alpha,\beta) = 2\sin(2\alpha + 2\beta) \end{array}\right. . \tag{7}$$

If the second order coefficients matches ours in Eq. (4), the fourth order are significantly different. To compare the distance defined by our coefficients c and the coefficients $\tilde{c}$ in Eq. (7), we plot the distance between a frame and all its rotation in Figure 6. For orthogonal frames, both distances are identical (Fig. 6a). For non-orthogonal frames the distances differs significantly on two



(a) Crosses (b) Non-orthogonal frames

Figure 6: Polar plot of the distance between two orthogonal frames (Fig. 6a) and two non-orthogonal frames (Fig. 6b) with respect to the rotation angle. The distance defined by our representation is plotted in blue and the PolyVector [8] distance in red. Insets highlights frames positions at local extrama. The PolyVector distance is not geometrically sound as the maximum distance is reached for overlapping non-orthogonal frames.



(a) PolyVector field [8] (b) $\lambda = 0.5$ (c) $\lambda = 1$

Figure 7: The smoothness energy from PolyVector field [20] achieves similar results than ours for $\lambda = 0.5$.

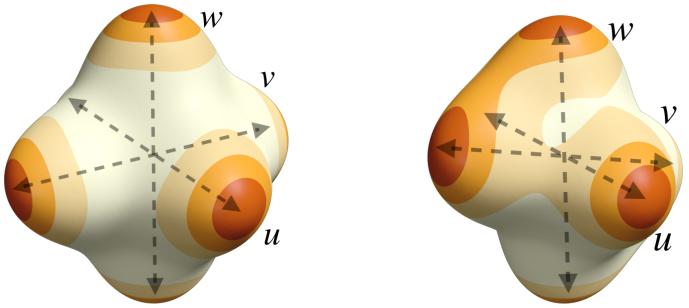
aspects. First, our distance has globally larger values, thus, with $\lambda = 1$ we penalize more the change of angle between frames than PolyVector. Second, the PolyVector distance does not reach its maximum at angle $\pi/2$ but at an angle where the frames overlap (Fig. 6b).

Frame field comparison. Using the change of variables $(\alpha,\beta) \rightarrow (\bar{u},\bar{v})$ in Eq. (7):

$$\left|\begin{array}{l} \tilde{c}_{2,1}(\bar{u},\bar{v}) = 2\bar{u}_1 + 2\bar{v}_1 \\ \tilde{c}_{2,2}(\bar{u},\bar{v}) = 2\bar{u}_2 + 2\bar{v}_2 \\ \tilde{c}_{4,1}(\bar{u},\bar{v}) = 2\bar{u}_1\bar{v}_1 - 2\bar{u}_2\bar{v}_2 \\ \tilde{c}_{4,2}(\bar{u},\bar{v}) = 2\bar{u}_1\bar{v}_2 + 2\bar{u}_2\bar{v}_1 \end{array}\right. ,$$

we can generate PolyVector fields by replacing c by $\tilde{c}$ in our smoothness energy E_s.

Figure 7 compares quad-meshes generated from our non-orthogonal direction fields with the one obtained from the coefficients $\tilde{c}$. We evaluate our method with parameter $\lambda = 0.5$ and $\lambda = 1$, all other parameters being equal. As expected with $\lambda = 1$, our method leads to a quad-mesh with an additional singularity in the upper right fillet that is not present in the PolyVector method [20]. We found that both methods leads to similar results when we set λ to 0.5. Indeed, although the PolyVector frame distance does not follow a geometric intuition, the difference with our smoothness energy is most prominent when the the frames are faraway from each other. Therefore, it does not greatly impact design of smooth fields as adjacent frames are close to each other.



(a) Orthogonal frame (b) Non-orthogonal deformation

Figure 8: 3-Direction sets $\{u, -u, v, -v, w, -w\}$ represented by spherical polynomials of degree 4.

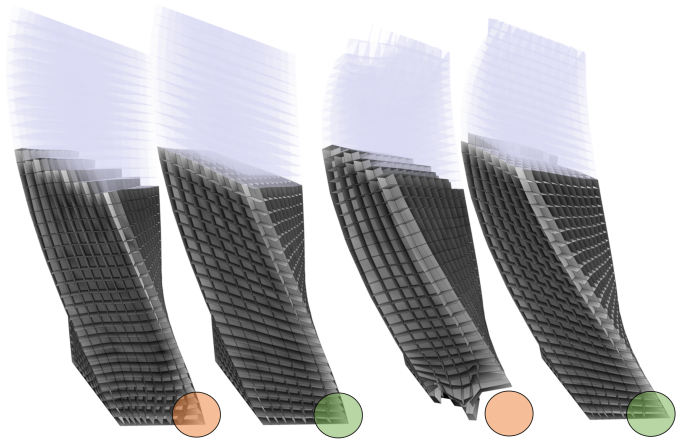


Figure 9: Extruded square with curl. The first result uses an orthogonal field, the second uses our fields. The following two images simply add more curl to the extrusion. Sharp corners (disk highlights) are better captured with our fields. The layer of hexahedra inside the volume also remains parallel to the extremities.

3. 3D frame fields

In the particular case of 3D frame fields, each frame has three directions noted u, v, w. Instead of the Fourier basis, the space of polynomial $\mathbb{P}_{2k,3}$ is decomposed with the Spherical Harmonic basis. We will see that when using polynomial of degree 4 our representation of 3D cross field only needs fourth order Spherical Harmonic functions much like the most used representation [13].

As noted in [13, 21], 3D cross fields can be represented by the spherical polynomial of degree four: $p(s) = s_1^4 + s_2^4 + s_3^4$. This is, in fact, the polynomial of lowest degree presenting the same symmetry as a 3D cross. However, higher order polynomials of even degree could be consider for frame field generation. In order to keep the computational complexity as low as possible, we will use degree 4 polynomials.

In this case, the representation polynomial adapted from Eq. (1) reads:

$$p(s) = \langle u, s\rangle^4 + \langle v, s\rangle^4 + \langle w, s\rangle^4, \tag{8}$$

and is plotted in two configurations in Figure 8.

3.1. Spherical Harmonic decomposition

The Spherical Harmonic functions constitute an orthogonal basis for spherical functions. In computer graphics, they have notably been used for BRDFs approximation for real-time rendering [28, 29].

Spherical Harmonics functions $Y_\ell^m : \mathbb{S}^2 \to \mathbb{R}$ are identified by two indices : ℓ the band index, corresponding to the frequency of the signal, and m spanning all functions within the frequency ℓ. Any spherical function $f : \mathbb{S}^2 \to \mathbb{R}$ is uniquely decomposed as:

$$f(s) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} f_{\ell,m} Y_\ell^m(s).$$

Zonal harmonics. Decomposing the representation polynomial of Eq. (8) in the Spherical Harmonics basis is made simple due to their rotational invariance. In particular, the spherical polynomial $f : s \mapsto s_3^4$, representing the direction z, is invariant by rotation around the z-axis. Thus, its decomposition in SH only uses the functions $Y_\ell^0,\ \ell \geq 0$:

$$f(s) = 63Y_0^0 + 36Y_2^0(s) + 8Y_4^0(s).$$

Such functions are called *zonal harmonic function* and have one fundamental property: their rotation to an arbitrary unit axis u is easily computed in the SH basis [30]:

$$\langle u, s\rangle^4 = 63Y_0^0Y_0^0 + 36 \sum_{m=-2}^{2} Y_2^m(u)Y_2^m(s) + 8 \sum_{m=-4}^{4} Y_4^m(u)Y_4^m(s).$$

Thus, the decomposition of the frame field polynomial in Eq. (8), is simply the sum of the above decomposition:

$$\left| \begin{array}{l} c_{2,m}(u, v, w) = Y_2^m(u) + Y_2^m(v) + Y_2^m(w), \\ c_{4,m}(u, v, w) = Y_4^m(u) + Y_4^m(v) + Y_4^m(w). \end{array} \right. \tag{9}$$

Very much like 2D frames, 3D frames only have coefficients in band $\ell = 2$ and $\ell = 4$. One can check from the expression of the SH that a frame with uniform length (*i.e.* $|u| = |v| = |w|$) is orthogonal if and only if all the coefficients $c_{2,m}$ are zero. So, for orthogonal frames we recover the basic property used when designing 3D cross fields [13, 21]: all rotations of the polynomial $\sum_{i=1,2,3} s_i^4$ are spanned by the SH of band 4. However, Equation (9) is more precise and gives the exact relation between the SH coefficients and the frame deformation.

3.2. Designing 3D frame fields with constraints

Unlike the 2D case, there is no obvious change of variables simplifying the expression of the coefficients in Eq. (9). Therefore, we keep the three unit vectors $u, v, w \in \mathbb{S}^2$ as variables. Like previous works [13, 21, 24], performing computation in the space of direction fields requires to solve a non-convex optimization problem.

Direction constraints. Direction constraints are easily set by directly constraining u,v or w. We can fix one, two or three directions at a time. The choice of which of these variables is fixed does not impact the results as our representation is agnostic to reordering of the direction set. This is very convenient for boundary conditions in global parametrization problems.

In all our experiments, a frame associated to a boundary facet has its directions constrained by vectors normal to the boundary. If a tet has one, two or three boundary triangles then the frame must satisfy one, two or three direction constraints.

Smoothness energy. The smoothness energy is a simple distance between coefficients of the decomposition of two adjacent frames

as explained in Section 1.3:

$$E_s(u,v,w) = \lambda \sum_{(ij)\in\mathcal{E}} \sum_{m=-2}^{2} \Big(c_{2,m}(u_i,v_i,w_i) - c_{2,m}(u_j,v_j,w_j)\Big)^2 + \sum_{(ij)\in\mathcal{E}} \sum_{m=-4}^{4} \Big(c_{4,m}(u_i,v_i,w_i) - c_{4,m}(u_j,v_j,w_j)\Big)^2 .$$

Orthogonality parameter. As noted in Section 3.1, the orthogonality of the frame is related to the coefficients of the second band. Thus, we penalize the band coefficients $\ell = 2$ in the energy with a weight λ. Increasing λ means that it is harder for the field to change its angle and favor fields with constant angle between its directions. As singularities have smaller energy when the field is orthogonal around singularities, this simple parameter on the order 2 coefficients forces global orthogonality of the field.

Figure 10 illustrates the effect on parameter λ on hexmeshes generation. Values lower than one generates polycube like parametrization with the singularities are at the boundary of the domain while values greater than one approach the cross field solution generated by [21].

3.3. Hex-meshing of CAD models

We evaluate the quality of our 3D frame fields by their ability to generate hex-meshes. In all our experiments, hex-meshes are extracted from the frame field using the state-of-the-art method [7, 31]. The orthogonal frames are generated using [21].

The optimization is done the same way as for 2D frame fields. We iterate between minimizing Eq. (6) with a diffusion parameters $\tau = 1, b = 100$ and renormalization of the vectors u, v and w.

The presence of sharp corners is a classic challenge for hex-remeshing methods. This situation occurs frequently for CAD model and is often badly handled by cross fields which collapses the hexes at the corner as in Figure 11a. Whereas the direction field designed by our method introduces a singularity and a valid hexmesh can be extracted (Figure 11b). To produce such results, it is primordial that frames at sharp corners are fully constrained.

Figure 9 also highlights the necessity of non-orthogonal frame fields to capture sharp corners. Cross fields generates hexmeshes with heavily distorted or missing elements while our method is able to fit a valid hex element in the corner.

In Figure 12, the presence of shear forces the orthogonal frames to oscillate to satisfy the boundary constraints while our frame field can remain constant.

Initialization. The energy E_s is non-convex, thus its outcome is highly dependent of the initialization. Figure 13 compares solutions of our optimization problem for three different starting points. The initialization with a random orthonormal frame field always converges to the same solution (Fig. 13b). An initialization with a smooth cross field [21] needs less iterations to reach convergence but the solution is still the same (Fig. 13a). As reported in Table 2, this reduces (sometimes considerably) the computation time. However, our optimization scheme gets trapped in a local minimum when initiated with a random field with nearly flat frames (Fig. 13c). At convergence, the total energy is higher than the optimal solution. Moreover, the field is corrupted by degenerate frames (circled in red) making it useless for meshing applications.

In all our experiments, we initialize the optimization with the smooth orthonormal frame field obtained by [21].

Fig.	Tets	Init	Opti	Meshing
13a	862	0.016	22	0.7
13b	862	–	25	0.7
13c	862	–	26	–
9	4420	0.096	31	3.0
10	13225	0.349	454	8.4
12b	10788	0.332	305	7.7

Table 2: Computation time in second for each step of our 3D frame field design method: initialization with an orthogonal field [21], solving our non-convex optimization problem and the hex-mesh extraction using [31].

Timings. Table 2 lists the computation time for generating a non-orthogonal frame field with our method. Again, the most time consuming step is the computation of the non-orthogonal frame field as it requires to solve a non-convex optimization problem. All the experiments are conducted on a laptop with a four-core, 2.6 GHz Intel Core i5-10400H and 32 GB of RAM. All algorithms are implemented in C++.

4. Conclusion

In this paper, we introduced a representation of direction fields common to surfaces and volumes, allowing constraints on one, two or three directions independently with an intuitive control on the frame orthogonality. We demonstrated that it can be useful for many geometry processing applications such as anisotropic planar and volumetric remeshing.

While our optimization is non-convex and more challenging than standard cross field representation, it allows more flexible constraints which are better suited for global parametrization problems.

In the future, we plan to study *integrable* direction fields, *i.e.* fields whose directions are gradients of scalar functions. This is fundamental to guarantee the existence of a *bijective* global parametrization. This work has been conducted for fields on surfaces but is still missing for volumetric meshes. With our representation we can hope to extend two dimensional results to 3D.

References

[1] F. Kälberer, M. Nieser, K. Polthier, Quadcover-surface parameterization using branched coverings, in: Computer graphics forum, Vol. 26, Wiley Online Library, 2007, pp. 375–384.

[2] A. Hertzmann, D. Zorin, Illustrating smooth surfaces, in: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 2000, pp. 517–526.

[3] C. Brandt, L. Scandolo, E. Eisemann, K. Hildebrandt, Modeling n-symmetry vector fields using higher-order energies, ACM Transactions on Graphics (TOG) 37 (2) (2018) 1–18.

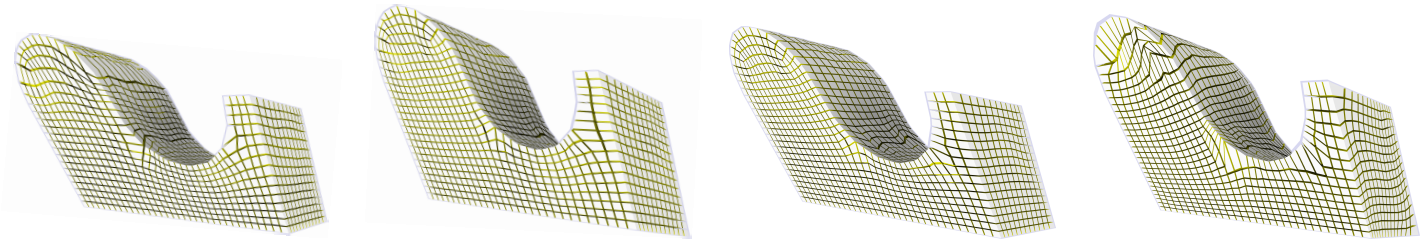
[4] A. O. Sageman-Furnas, A. Chern, M. Ben-Chen, A. Vaxman, Chebyshev nets from commuting polyvector fields, ACM Transactions on Graphics (TOG) 38 (6) (2019) 1–16.

[5] E. Iarussi, D. Bommes, A. Bousseau, Bendfields: Regularized curvature fields from rough concept sketches, ACM Transactions on Graphics (TOG) 34 (3) (2015) 1–16.

[6] M. Bessmeltsev, J. Solomon, Vectorization of line drawings via polyvector fields, ACM Transactions on Graphics (TOG) 38 (1) (2019) 1–12.

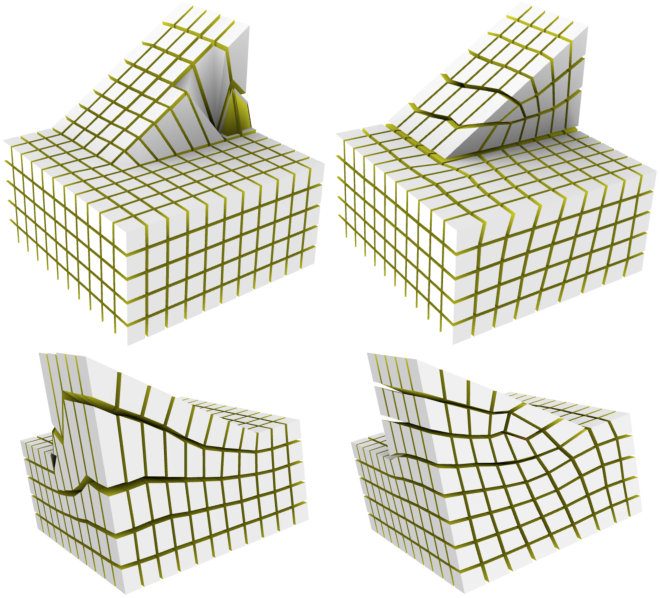
[7] M. Nieser, U. Reitebuch, K. Polthier, Cubecover–parameterization of 3d volumes, in: Computer graphics forum, Vol. 30, Wiley Online Library, 2011, pp. 1397–1406.

[8] O. Diamanti, A. Vaxman, D. Panozzo, O. Sorkine-Hornung, Designing n-polyvector fields with complex polynomials, in: Computer Graphics Forum, Vol. 33, Wiley Online Library, 2014, pp. 1–11.



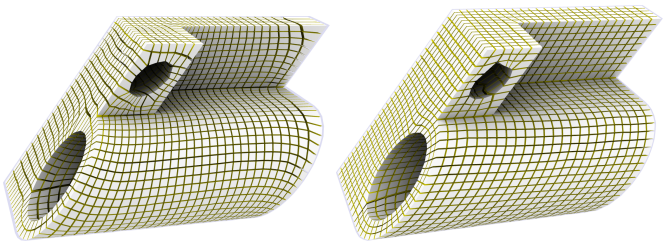
(a) $\lambda = 0.7$ (b) $\lambda = 1$ (c) $\lambda = 5$ (d) Orthogonal [21]

Figure 10: Results of our 3D frame field design method for increasing values of the orthogonality parameter λ. Low values (left) push the singularities out of the domain while high values approaches the 3D cross field parametrization obtained by Ray *et al.* [21] (right).



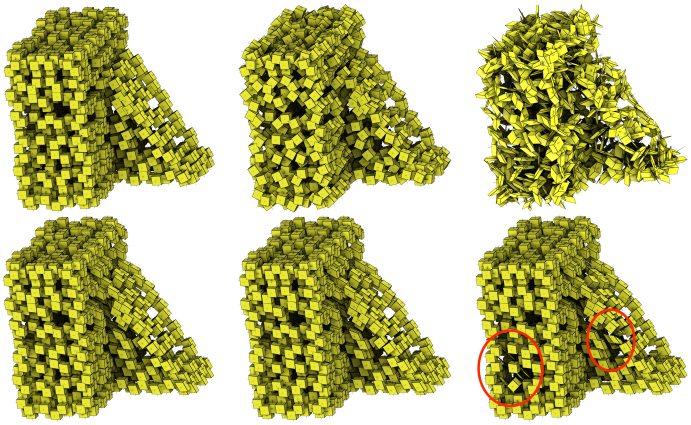
(a) Cross field [21] (b) Ours $\lambda = 1$

Figure 11: On a CAD model with a sharp corner, the 3D cross field collapses part of the model (Fig. (11a)). Our method successfully adds a singularity to compensate the slope (Fig. (11b)).



(a) Cross field [21] (b) Ours $\lambda = 1$

Figure 12: On a CAD model with heavy shear, the 3D cross field from creates unnecessary oscillation to compensate (Fig. (12a)). Our method outputs nicely shaped and regular hexes (Fig. (12b)).



(a) Orthogonal $E = 26.8$ (b) Random ortho $E = 26.8$ (c) Random $E = 38.1$

Figure 13: Results of our 3D frame field design method starting from three different initializations. Starting from a smooth orthogonal frame field [21] (Fig. 13a top) or from a random orthogonal field (Fig. 13b top) leads to the same solution (Fig. 13a–Fig. 13b bottom). The initialization with a completely random field, including degenerate frames, gets trapped in a degenerate local minimum, resulting in a higher total energy E (Fig. 13c).

[9] D. Panozzo, E. Puppo, M. Tarini, O. Sorkine-Hornung, Frame fields: Anisotropic and non-orthogonal cross fields, ACM Transactions on Graphics (TOG) 33 (4) (2014) 1–11.

[10] T. Jiang, X. Fang, J. Huang, H. Bao, Y. Tong, M. Desbrun, Frame field generation through metric customization, ACM Transactions on Graphics (TOG) 34 (4) (2015) 1–11.

[11] N. Ray, B. Vallet, W. C. Li, B. Lévy, N-symmetry direction field design, ACM Transactions on Graphics (TOG) 27 (2) (2008) 1–13.

[12] F. Knöppel, K. Crane, U. Pinkall, P. Schröder, Globally optimal direction fields, ACM Transactions on Graphics (ToG) 32 (4) (2013) 1–10.

[13] J. Huang, Y. Tong, H. Wei, H. Bao, Boundary aligned smooth 3d cross-frame field, ACM transactions on graphics (TOG) 30 (6) (2011) 1–8.

[14] M. Nieser, J. Palacios, K. Polthier, E. Zhang, Hexagonal global parameterization of arbitrary surfaces, IEEE Transactions on Visualization and Computer Graphics 18 (6) (2011) 865–878.

[15] J. Palacios, E. Zhang, Rotational symmetry field design on surfaces, ACM Transactions on Graphics (TOG) 26 (3) (2007) 55–es.

[16] K. Crane, M. Desbrun, P. Schröder, Trivial connections on discrete surfaces, in: Computer Graphics Forum, Vol. 29, Wiley Online Library, 2010, pp. 1525–1533.

[17] A. Vaxman, M. Campen, O. Diamanti, D. Panozzo, D. Bommes, K. Hildebrandt, M. Ben-Chen, Directional field synthesis, design, and processing, in: Computer Graphics Forum, Vol. 35, Wiley Online Library, 2016, pp. 545–572.

[18] F. de Goes, M. Desbrun, Y. Tong, Vector field processing on triangle meshes, in: ACM SIGGRAPH 2016 Courses, 2016, pp. 1–49.

[19] Y. Liu, W. Xu, J. Wang, L. Zhu, B. Guo, F. Chen, G. Wang, General planar quadrilateral mesh design using conjugate direction field, ACM Transactions on Graphics (TOG) 30 (6) (2011) 1–10.

[20] O. Diamanti, A. Vaxman, D. Panozzo, O. Sorkine-Hornung, Integrable

polyvector fields, ACM Transactions on Graphics (TOG) 34 (4) (2015) 1–12.
[21] N. Ray, D. Sokolov, B. Lévy, Practical 3d frame field generation, ACM Transactions on Graphics (TOG) 35 (6) (2016) 1–9.
[22] Z. Shen, X. Fang, X. Liu, H. Bao, J. Huang, Harmonic functions for rotational symmetry vector fields, in: Computer Graphics Forum, Vol. 35, Wiley Online Library, 2016, pp. 507–516.
[23] A. Chemin, F. Henrotte, J. Remacle, J. Van Schaftingen, Representing three-dimensional cross fields using 4th order tensors, imr2018: 27th international meshing roundtable (x. roca and a. loseille, eds.), Lecture Notes in Computational Science and Engineering 127 89–108.
[24] D. Palmer, D. Bommes, J. Solomon, Algebraic representations for volumetric frame fields, ACM Transactions on Graphics (TOG) 39 (2) (2020) 1–17.
[25] R. Viertel, B. Osting, An approach to quad meshing based on harmonic cross-valued maps and the ginzburg–landau theory, SIAM Journal on Scientific Computing 41 (1) (2019) A452–A479.
[26] C. Zhu, R. H. Byrd, P. Lu, J. Nocedal, Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization, ACM Transactions on Mathematical Software (TOMS) 23 (4) (1997) 550–560.
[27] D. Bommes, M. Campen, H.-C. Ebke, P. Alliez, L. Kobbelt, Integer-grid maps for reliable quad meshing, ACM Transactions on Graphics (TOG) 32 (4) (2013) 1–12.
[28] P.-P. Sloan, J. Kautz, J. Snyder, Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments, in: Proceedings of the 29th annual conference on Computer graphics and interactive techniques, 2002, pp. 527–536.
[29] R. Green, Spherical harmonic lighting: The gritty details, in: Archives of the game developers conference, Vol. 56, 2003, p. 4.
[30] P.-P. Sloan, Stupid spherical harmonics (sh) tricks, in: Game developers conference, Vol. 9, 2008, p. 42.
[31] M. Lyon, D. Bommes, L. Kobbelt, Hexex: Robust hexahedral mesh extraction, ACM Transactions on Graphics (TOG) 35 (4) (2016) 1–11.