Modeling n-Symmetry Vector Fields using Higher-Order Energies

CHRISTOPHER BRANDT, LEONARDO SCANDOLO, ELMAR EISEMANN, and KLAUS HILDEBRANDT,
Delft University of Technology, The Netherlands

We introduce a variational approach for modeling n-symmetry vector and direction fields on surfaces that supports interpolation and alignment constraints, placing singularities and local editing, while providing real-time responses. The approach is based on novel biharmonic and m-harmonic energies for n-fields on surface meshes and the integration of hard constraints to the resulting optimization problems. Real-time computation rates are achieved by a model reduction approach employing a Fourier-like n-vector field decomposition, which associates frequencies and modes to n-vector fields on surfaces. To demonstrate the benefits of the proposed n-field modeling approach, we use it for controlling stroke directions in line-art drawings of surfaces and for the modeling of anisotropic BRDFs which define the reflection behavior of surfaces.

CCS Concepts: • **Mathematics of computing → Discretization**; • **Computing methodologies → Mesh models**; *Non-photorealistic rendering*; *Reflectance modeling*;

Additional Key Words and Phrases: Vector field design, RoSy-fields, direction fields, spectral methods, model reduction, non-photorealistic rendering, anisotropic BRDFs

ACM Reference Format:
Christopher Brandt, Leonardo Scandolo, Elmar Eisemann, and Klaus Hildebrandt. 2018. Modeling n-Symmetry Vector Fields using Higher-Order Energies. *ACM Trans. Graph.* 37, 2, Article 18 (February 2018), 18 pages. https://doi.org/10.1145/3177750

1 INTRODUCTION

The design, synthesis and processing of tangential vector fields on surfaces is essential for various applications in computer graphics. Often, we are not dealing with classical vector fields, but with n-fold rotational symmetry vector fields (or n-vector fields), like line fields (n=2) or cross fields (n=4). The structure of n-vector fields differs significantly from that of classical vector fields. For example, n-vector fields allow for more general singularities of fractional degree, and, compared to vector fields, the calculus of n-vector fields is scarcely developed. Therefore, the processing of n-vector fields poses challenging problems and promises rewarding benefits for the applications.

The goal of this paper is to develop the techniques needed for a *modeling tool* for n-vector fields on surfaces that includes the following features:

(1) *Hard interpolation constraints.* This enables users to create smooth n-vector fields by specifying a sparse sets of interpolation constraints.
(2) *Smooth dependence on the constraints.* The constructed n-vector fields should depend smoothly on the constraints. This allows users to edit n-vector fields by modifying the constraints as changing the constraints smoothly changes the n-vector field that is modeled.
(3) *Local editing.* To fine-tune results, the user should be able to mark a region and only model the field inside the region without affecting the n-vector field outside the marked region.
(4) *Real-time responses.* To enable modeling and fine-tuning of n-vector fields, immediate responses are needed.
(5) *n-direction fields.* The tool should allow modeling of n-direction fields, which are n-vector fields consisting only of unit vectors.

For surface modeling and other modeling tasks, tools offering analogous to the first four features proved to be effective. Therefore we are convinced that a modeling tool for n-vector fields that combines these features is helpful for a variety of graphics applications. The fifth feature is important for applications in which directions rather than vectors are needed and the magnitude of vectors is not relevant.

To realize these goals, we introduce *n-field splines*, a variational approach for modeling n-vector fields and n-direction fields on surfaces. The basis of our approach are novel higher-order fairness energies for n-vector fields: a biharmonic energy and more generally m-harmonic energies. The n-vector field splines are defined, analogously to the variational characterization of classical spline functions, as the minimizers of a higher-order fairness energy subject to constraints. The use of higher-order energies enables us to integrate constraints that realize the desired modeling features to the variational problem. Local editing and interpolation constraints at single points can be enforced in the optimization and the higher-order energies ensure that the resulting minimizers smoothly transition from the edited to the constrained region and behave smoothly around the interpolation constraints.

To implement the concept of n-field splines, we developed several new techniques. We introduce a biharmonic energy and more generally m-harmonic energies for piecewise constant (face-based) n-vector fields on triangle surface meshes. By applying a principle for the design of quadratic fairness energies for direction fields proposed in [Knöppel et al. 2013] to our setting, we extend the m-harmonic energies for n-vector fields to m-harmonic energies for n-direction fields. Secondly, we integrate hard interpolation and alignment constraints, as well as constraints for placing singularities to the minimization of the higher-order energies for n-vector and n-direction fields. These approaches extend the weak alignment constraints for the globally optimal n-direction field approaches introduced in [Knöppel et al. 2013] and [Liu et al. 2016]. Thirdly, we propose an efficient approximation algorithm for n-field splines that allows for real-time modeling. The n-field splines are solutions of sparse linear systems, and, therefore, they can be robustly

Authors' address: Christopher Brandt; Leonardo Scandolo; Elmar Eisemann; Klaus Hildebrandt,
Delft University of Technology, Department of Intelligent Systems, Van Mourik Broekmanweg 6, Delft, 2628 XE, The Netherlands, {c.brandt,l.scandolo,e.eisemann, k.a.hildebrandt}@tudelft.nl.

computed. However, for many applications, in particular for the modeling of *n*-fields, real-time responses are desired or even necessary. The technical basis of our approximation algorithm is a Fourier-like representation that associates frequencies spectra and corresponding modes to *n*-fields. The algorithm computes *n*-fields satisfying the user-constraints in an appropriate subspace spanned by low-frequency modes. In this sense, the approximation algorithm computes an additionally low-pass filtered *n*-field spline. Since the quadratic fairness energies in the subspace are represented by diagonal matrices, the computation of the reduced solutions is very fast. After a preprocessing stage in which eigenfields of an *n*-field Laplacian are computed, we obtain computation times of few milliseconds for all meshes we tested and a speed-up of a factor up to 100 over solving the full linear system using sparse direct solvers.

To emphasize the applicability of the *n*-field splines and the real-time solver, we apply the resulting modeling tool to two graphics problems: real-time editing of hatchings of surfaces and interactive design of anisotropic BRDFs on surfaces. In addition to these applications, we think that the proposed techniques hold potential for quad meshing applications. For example, in [Ebke et al. 2016] the computational cost of vector field design is named as one of the factors that hinder interactive quad meshing via integer grid maps. Our scheme could potentially remove this barrier.

2 RELATED WORK

Tools for designing *n*-fields are important for various applications in computer graphics. They are used for texture generation [Chi et al. 2014; Knöppel et al. 2015; Praun et al. 2000; Turk 2001; Wei and Levoy 2001], non-photorealistic line art [Hertzmann and Zorin 2000] and painterly rendering [Zhang et al. 2007], image stylization [Yao et al. 2012], anisotropic shading [Mehta et al. 2012; Raymond et al. 2014], quad-remeshing [Bommes et al. 2009; Kälberer et al. 2007; Li et al. 2011; Ray et al. 2006; Tarini et al. 2011] or hexagonal parameterization [Nieser et al. 2012], and surface segmentation [Solomon et al. 2011; Zhuang et al. 2014] to name just a few examples. The processing of *n*-fields poses challenging problems and much work has been dedicated to establishing techniques that tackle these problems. In the following, we outline approaches closest related to the proposed work. For more background and further references, we refer to the recent surveys [de Goes et al. 2015; Vaxman et al. 2016].

Variational n-field design. In variational *n*-field design, *n*-fields are constructed as solutions to optimization problems, which aim for the smoothest *n*-fields that satisfy design goals specified by the users. The smoothness of a *n*-field is measured by a fairness energy, an objective that quantifies the variation of the field along the surface. To compare the *n*-vectors of a *n*-vector fields at nearby points (*e.g.* neighboring triangles or vertices) the *n* vectors at one point are parallel transported along the shortest geodesic to the tangent space of the other point. One way to quantify the difference between two *n*-vectors (once they are in the same tangent space) is to select an arbitrary vector from both *n*-vectors and to measure the oriented angle between the selected vectors. Multiplying this angle by *n*, yields a quantity that, up to a multiple of 2π, is independent of the choice of vectors. The cosine of this quantity agrees with the cosine of the smallest angle between pairs of vector from the two *n*-vectors and one minus this cosine can be used as a measure of the deviation of the *n*-vectors. Based on this idea, a fairness energy for *n*-direction fields was introduced in [Hertzmann and Zorin 2000]. This approach was extended by the concept of the representation vector [Palacios and Zhang 2007; Ray et al. 2006], which is an alternative representation of *n*-vector fields on meshes. Modulo π/n, the *n* vectors of a *n*-vector make the same angle to a fixed coordinate direction in the tangent plane, hence, multiplying the angles by *n* yields a unique representation vector. This concept allows to formulate design of *n*-vector or *n*-direction fields as an optimization of the representation vector fields [Ray et al. 2006]. In recent work [Knöppel et al. 2013; Liu et al. 2016], a *n*-vector field representation using complex numbers and fairness energies that are quadratic with respect to the complex representation are introduced. The benefit of this approach is that globally optimal solutions can be computed by solving sparse linear systems. The approach can be extended to the computation of optimal *n*-direction fields. This is achieved by imposing a constraint on the $\mathbf{L}^2$-norm of the *n*-vector field during optimization and pointwise normalization afterwards. For controlling the design, [Knöppel et al. 2013] use a weak alignment constraint to an input field, *e.g.*, the principal curvatures directions of the surfaces. In [Liu et al. 2016] this approach is extended by a stroke-based design metaphor, in which fields weakly align with strokes placed by users. Globally optimal fields that weakly align with the strokes and an alignment field are computed and runtimes of 1s for a mesh with 50k triangles are reported. Our work extends this approach. It allows for spline-like modeling of *n*-vector fields including features like modeling with interpolation constraints, local editing and real-time responses. Our biharmonic energy for *n*-vector fields is also a quadratic energy, but compared to the Dirichlet energy of [Knöppel et al. 2013], the corresponding Euler-Lagrange equation is of higher order, which is a prerequisite for modeling with interpolation constraints and local editing.

Mixed-integer problems. An alternative approach to using the representation vector is to introduce explicit assignments, so-called matchings, between the *n* vectors at neighboring triangles (or vertices depending on the discretization used). Once a matching is fixed, differences between *n*-vectors can be measured using common measures for comparing vectors. For *n*-vector and *n*-direction field design, all possible matchings are introduced as variables to the optimization with the goal to find the best possible *n*-field and matching [Bommes et al. 2009; Kälberer et al. 2007; Ray et al. 2008]. This means that mixed-integer problems need to be solved for *n*-field construction. Hard interpolation constraints have been used for mixed-integer based *n*-field design. However, since mixed-integer problems need to be solved for field construction, these methods do not provide real-time responses.

Real-time design. Recently, Jakob et al. [2015] introduced a method for real-time quadrilateral and hexagonal mesh generation. The scheme proceeds in two stages: *n*-field design and mesh generation based on the *n*-field. To obtain a real-time system, the *n*-field design is not done by solving a global optimization problem, instead a multiresolution hierarchy is set up and local optimization steps are performed on the different levels of the hierarchy from coarse to fine. For efficiency, the objective for the local optimization steps is an extrinsic fairness energy for n-fields that does not need parallel transport of vectors. As a consequence, the objective depends not just on intrinsic properties of the surface but also on its extrinsic curvatures. In [Huang and Ju 2016] the extrinsic fairness energy is further explored and the relations between extrinsic and intrinsic fairness energies are analyzed. Since [Jakob et al. 2015] is the only scheme that can construct n-fields at rates comparable to our scheme, we include a comparison to their method to Section 9.

Polyvectors. Polyvectors [Diamanti et al. 2014] extend the idea of a representation vector for rotational symmetric n-vectors to arbitrary (non-symmetric) n-vectors. The idea is to assign to every n-vector the complex polynomial that has the n vectors as its roots, where vectors in $\mathbb{R}^2$ are identified with complex numbers. The space of polynomials of degree n is a vector space and n-vector field design problems can be formulated using this representation. A harmonic energy for polyvectors was introduced in [Diamanti et al. 2014] and additional objectives for quad-remeshing in [Diamanti et al. 2015]. For a discussion of benefits and drawbacks of the different representations of n-vector fields, we refer to the survey [Vaxman et al. 2016].

Controlling topology. Singularities are salient features of n-fields. Methods for controlling and editing singularities of vector, n-vector and n-direction fields and for generating fields from given sets of singularities have been proposed [Crane et al. 2010; Fisher et al. 2007; Lai et al. 2010; Liu et al. 2016; Palacios and Zhang 2007; Ray et al. 2009; Zhang et al. 2006]. Our approach combines variational field design with the enforcement of singularities. Singularities can be placed on the surface and the higher-order fairness energies ensure smooth transition of the field around the singularities.

Modeling of vector fields. Vector fields (or 1-fields) are a special case of n-vector fields. The topological and geometric properties of vector fields and differential operators on spaces of vector fields are well studied, and discretizations of classical operators on triangle meshes have been established [Desbrun et al. 2005; Polthier and Preuß 2003; Tong et al. 2003; Wardetzky 2006]. In [Brandt et al. 2017; Fisher et al. 2007] quadratic fairness energies have been used for variational vector field design. In both papers, the benefits of using biharmonic energies over harmonic energies for vector field design and modeling are emphasized. Our approach extents these techniques to the modeling of n-vector and n-direction fields using biharmonic energies.

Spectral processing of vector fields. In the last decade, spectral methods, which use the spectrum and eigenfunctions of the Laplace–Beltrami operator, have been very successful for various tasks in shape analysis and processing. For an introduction to spectral mesh processing and an overview of applications, we refer to the surveys of Lévy and Zhang [2009] and Zhang et al. [2010]. For fluid simulation, subspaces of low frequency modes of the vector Laplacian have been used for speeding-up computations [De Witt et al. 2012]. Recently, this approach has been generalized to fluid simulation on surfaces, where the eigenmodes of the Hodge–Laplace operator are used [Liu et al. 2015]. In a parallel development, the eigenmodes of the Hodge–Laplace operator have been used for spectral processing of tangential vector fields on surfaces [Brandt et al. 2017]. In a different recent development, eigenfields of the Hodge–Laplace operator have been used for functional representation of linear operators on spaces of tangential vector fields [Azencot et al. 2015]. In this paper, we further develop these lines of work and compute the first spectral decomposition of tangential n-vector fields on surfaces.

Higher-order energies. Optimization problems involving higher-order energies, like the biharmonic, thin plate or Willmore energy, have been used for example for fairing [Hildebrandt and Polthier 2007; Kobbelt et al. 1998], variational surface modeling [Jacobson et al. 2010; Sorkine and Cohen-Or 2004], deformation-based mesh editing [Botsch and Kobbelt 2004; Sorkine et al. 2004] and the construction of skinning weights [Jacobson et al. 2011; Wang et al. 2015]. One example of a benefit provided by higher-order energies is more control over the boundary behavior. For example, biharmonic problems allow to prescribe positions and derivatives (in normal direction) at the boundary, which allow to create G^1-continuous transitions at boundaries of surface patches [Kobbelt et al. 1998]. In contrast, using harmonic problems only G^0-transitions can be obtained. A second example is that interpolation constraints can be imposed at single points. For example, the minimizers of the thin plate energy over a two-dimensional domain subject to interpolation constraints at single points exist and are uniquely defined [Löhndorf and Melenk 2017]. In contrast, minimizers of the harmonic energy over a two-dimensional domain subject to point constraints in general are discontinuous, see [Braess 2007, pp 50–51] for an example. For applications like surface modeling and deformation it is desirable to be able to impose constraints on single points. The biharmonic and m-harmonic energies we propose provide the benefits of higher-order energies for the modeling of n-vector and n-direction fields.

3 BACKGROUND: n-VECTOR FIELDS

A n-symmetry vector (short: n-vector) in $\mathbb{R}^2$ is a set $\{v_1, v_2, ..., v_n\}$ of n vectors with a $\frac{2\pi}{n}$-fold rotational symmetry, *i.e.*, rotations by $\frac{2\pi}{n}$ map the set to itself. For example, a 1-vector is an ordinary vector and a 2-vector is a pair $\{v, -v\}$, where $v \in \mathbb{R}^2$ is arbitrary.

Representation vector. Consider the map that rotates any vector such that its argument, *i.e.* the oriented angle with the x-axis, scales by a factor of n and the length is preserved. If this map is applied to a n-vector $\{v_1, v_2, ..., v_n\}$, all elements v_i have the same image u, which is called the representation vector of $\{v_1, v_2, ..., v_n\}$. Moreover, the map from n-vectors to representation vectors is a bijection, which means that every n-vector has a unique representation vector and every vector $\mathbb{R}^2$ is the representation vector of a n-vector. The representation vector provides additional structure that we will use to work with n-vectors. For example, we can add n-vectors by converting them to representation vectors, adding the representation vectors and converting them back to n-vectors. In combination with the natural scaling of n-vectors, we obtain a vector space structure on the set of n-vectors.

For later use, we want to remark that the representation vector changes when moving from one coordinate system to another. We consider a rotation of the coordinate system by an angle of φ. Then the coordinates of the n-vector in the rotated system are given by $\{R^{-\varphi}v_1, R^{-\varphi}v_2, ..., R^{-\varphi}v_n\}$, where $R^{-\varphi}$ denotes the rotation by $-\varphi$. The representation vector in the rotated coordinate system is $R^{-n\varphi}u$. This means, the representation vector u is rotated by $-n\varphi$ when the coordinate system is rotated by φ.

n-vector fields. We consider tangential n-vector fields on triangle meshes that are constant in every triangle. To work with such n-vector fields, we fix a coordinate system in the tangent plane of every triangle and consider the vector $\mathbf{u} = (u_1, u_2, ..., u_{|\mathcal{F}|}) \in \mathbb{R}^{2|\mathcal{F}|}$, where $u_i \in \mathbb{R}^2$ is the representation vector of the n-vector of triangle T_i (with respect to the coordinate system fixed in T_i) and $|\mathcal{F}|$ is the number of triangles of the mesh. Explicitly, we choose one oriented edge in every triangle and use this as the x-axis of the coordinate system. The vector $\mathbf{u}$ completely describes the n-field, *e.g.*, we can reconstruct the n-vectors of every triangle T_i from its representation vector u_i. We will perform all computations using the representation vectors. Only once we convert initial input, like an alignment field, to representation vectors, and, after the computation is done, we covert the result into a n-vector field for visualization or other applications like constructing a hatching or BRDF on the surface. To simplify the presentation, we will sometimes refer to $\mathbf{u}$ as the n-vector field and u_i as the n-vector and rely on the context to make the distinction between representation vector and n -vector.

The representation vectors, hence also the piecewise constant n-vector fields on a mesh, form a vector space. On this space, we consider the $\mathbf{L}^2$-scalar product

$$\left\langle \mathbf{u}, \mathbf{u}' \right\rangle_{\mathbf{L}^2} = \sum_i \text{area}(T_i) \left\langle u_i, u_i' \right\rangle$$

and the corresponding $\mathbf{L}^2$-norm

$$\|\mathbf{u}\|^2_{\mathbf{L}^2} = \langle \mathbf{u}, \mathbf{u} \rangle_{\mathbf{L}^2} ,$$

where $\mathbf{u}, \mathbf{u}' \in \mathbb{R}^{2|\mathcal{F}|}$.

n-direction fields. n-direction fields are n-vector fields consisting only of unit-length n-vectors. These fields are of particular interest as for many applications the magnitudes of the vectors are irrelevant.

Transport of n-vectors. The fairness measures, which will be introduced in the following sections, compare the n-vectors of a n-vector fields in adjacent triangles. To do so, one n-vector is parallel transported from its tangent plane to the tangent plane of the other n-vector. In this paragraph, we discuss the transport of a n-vector from a triangle T_i to an adjacent triangle T_j. We fix coordinate systems in both triangles and denote by $u = (u_x, u_y)$ the coordinates of the corresponding representation vector in T_i. If the x-axes of the coordinate systems in both triangles are aligned with the oriented edge e_{ij} that is common to both triangles, the transport is simply given by the identity matrix, *i.e.*, the coordinates of the transported vector in the triangle T_j agree with the coordinates u of the vector in T_i. In the general case, where the coordinate systems in the triangles T_i and T_j may not align to the common edge, we first transform to the e_{ij}-aligned coordinate system in T_i, then use the trivial transport to the e_{ij}-aligned coordinate system in T_j, and, finally, transform to the (non e_{ij}-aligned) coordinate system in T_j. Let φ_{ij} and φ_{ji} be the oriented angles between the edges chosen as x-axis in T_i and T_j and the common edge e_{ij}. Then, the representation vector transforms by multiplication with the rotation matrices $R^{-n\varphi_{ij}}$ and $R^{-n\varphi_{ji}}$. Altogether, the transport of the representation vector from T_i to T_j is given by a rotation

$$R_{ij} = R^{n(\varphi_{ji} - \varphi_{ij})}. \tag{1}$$

In practice, it is convenient to precompute the rotations R_{ij} for every pair of adjacent triangles.

We want to remark that the R_{ij}s form a discrete connection on the mesh, see [Crane et al. 2010; Knöppel et al. 2013] for more background on discrete connections. Since the R_{ij}s depend on n, the connections differ for the different types of n-fields. For n=1, the R_{ij}s agree with the usual parallel transport of vectors (discrete Levi-Civita connection) induced by the metric on ambient 3-space.

Singularities. The singularities of a n-vector field are of fractional degree $\frac{\iota}{n}$, where $\iota \in \mathbb{N}$. Since we are working with vector fields that are discontinuous, the classical notion of singularities cannot be applied, Therefore, the concept of discrete singularities has been developed, see [Crane et al. 2010; Ray et al. 2009] for more background. In our setting, the representation vectors and the discrete connection can be used to extract information about the n-field singularities. To every vertex, we associate an index which equals the number of full rotations of the vectors in the one ring around the vertex divided by n. Explicitly, let $T_0, ..., T_k$ be the oriented triangle 1-ring (ordered clockwise) around some vertex and $u_0, ..., u_k$ the corresponding n-vector representatives in each of those triangles. We then transport the vector u_0 into the next triangle T_1 and compute the signed angular difference $\psi_0 \in (-\pi, \pi]$ between the transported vector and u_1. $\psi_1, ..., \psi_k$ are being computed in the same fashion, where for ψ_k, we transport u_k into the triangle T_0. The sum of the angular differences will be an integer multiple of 2π and usually 0. If it is different from 0, we say the n-vector field has a singularity of index $\sum_j \psi_j/(2\pi n)$ at that vertex. The number singularities of a n-vector field is not arbitrary. The discrete Poincaré–Hopf theorem for n-vector fields, [Ray et al. 2008], states that the sum of the indices over all vertices equals $2 - 2g$, where g is the genus of the surface.

Polyvector fields. Instead of restricting to rotational symmetric n-vectors, one can consider other constraints or even n independent vectors. Such n-vectors can be described as the roots of a complex polynomial, cf. [Diamanti et al. 2014]. The coefficients of the polynomial then make up a representation analogous to the representation vector of rotational symmetric n-vectors. We want to remark that the constructions proposed in this paper can be carried over to this setting. Only the addition of representation vectors is replaced by addition of complex polynomials and the absolute value of the representation vector is replaced by a norm for complex polynomials of degree n.

4 HARMONIC ENERGY FOR n-VECTOR FIELDS

The basis of the *globally optimal n-direction fields* approach introduced in [Knöppel et al. 2013] are quadratic fairness energies for n-vector fields. The benefit of using quadratic fairness energies for n-vector field design over previous highly-nonlinear approaches is that the globally optimal solutions can be computed by solving linear systems. Whereas in [Knöppel et al. 2013] a space of vertex-based vector fields is used for discretization, we are considering piecewise constant face-based vector fields, a commonly used alternative setting, here. In [Knöppel et al. 2013], a one-parameter family of quadratic energies is studied. The harmonic energy we consider corresponds to the anti-holomorphic energy in their notation. In this section, we first introduce a novel harmonic energy for piecewise constant n-vector fields on meshes. Then we summarize how the harmonic energy can be used for n-vector and n-direction field design with weak alignment constraints following the approach of Knöppel et al. [2013].

Harmonic energy for piecewise constant n-vector fields. Since we are able to transport n-vectors from a triangle to its neighbors via (1), we can quantify the difference of vectors in neighboring triangles via $\|R_{ij}u_i - u_j\|^2$. These differences can be used to construct a harmonic energy for piecewise constant n-vector fields $\mathbf{u} = \left(u_1, \ldots, u_{|\mathcal{F}|}\right)$ on triangle meshes $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$:

$$E_H(\mathbf{u}) = \sum_{(i,j)\in\mathcal{E}} w_{ij}\|R_{ij}u_i - u_j\|^2 \qquad (2)$$

$$\text{where } \; w_{ij} = \frac{3l^2_{\mathbf{e}_{ij}}}{\text{area}(T_i \cup T_j)},$$

with $l_{\mathbf{e}_{ij}}$ being the length of the common edge between triangles T_i and T_j. This harmonic energy is a natural extension to n-vector fields of the harmonic energy for face-based, piecewise constant, tangential vector fields, see [Brandt et al. 2017]. A full derivation of the energy, and the weights w_{ij} in particular, can be found in the appendix. The energy (2) is quadratic in $\mathbf{u}$. Hence, there is a corresponding n-vector field Laplacian $\mathbf{\Delta}$ for piecewise constant n-fields, which is the self-adjoint operator[1] corresponding to the discrete harmonic energy, *i.e.*, it is defined via

$$\forall\, \mathbf{u} \;:\; \langle \mathbf{\Delta u}, \mathbf{u}\rangle_{\mathrm{L}^2} = E_H(\mathbf{u}). \qquad (3)$$

In [Diamanti et al. 2014], a harmonic energy (called Dirichlet energy in their paper) for piecewise constant n-vector fields was already introduced. In the following, we discuss the relation of the two energies. In our notation, the energy introduced in [Diamanti et al. 2014] is given by

$$\sum_{(i,j)\in\mathcal{E}} \|R_{ij}u_i - u_j\|^2.$$

The difference to the proposed harmonic energy is that no weights are used. In this sense, this is a *combinatorial* harmonic energy that does not account for the geometry of the triangulation. This leads to undesired results for meshes with irregular triangulations. An example is shown in Figure 1, where we compare the smoothest, 2-direction fields on a symmetric mesh using our harmonic energy and the combinatorial harmonic energy from [Diamanti et al. 2014]. For the combinatorial harmonic energy, the singularities appear in an unsymmetric pattern, *i.e.*, the resulting fields depend on the triangulation. In contrast, when using the proposed geometric harmonic energy instead, the four singularities appear in a symmetric pattern. This is the same pattern one obtains when using a regular triangulation of the same shape.

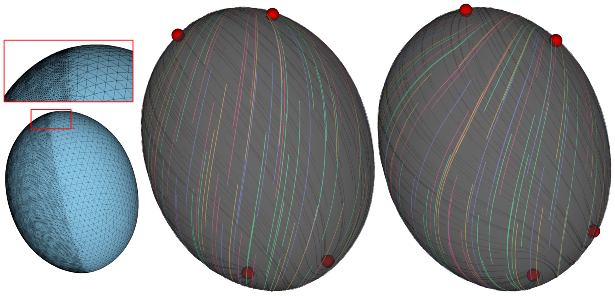

Fig. 1. Comparing the smoothest 2-fields on an ellipse mesh with irregular triangulation (the mesh is shown on the left) using our harmonic energy (middle) and the harmonic energy proposed in [Diamanti et al. 2014].

n-direction fields. The harmonic energy for n-vector fields cannot directly be used for n-direction field design. As shown in [Knöppel et al. 2013], minimizing the harmonic energy over n-vector fields with a unit-length constraints per vector is ill-posed. To get a well-posed problem, they propose optimizing over all rescalings of the field and adding a single $\mathbf{L}^2$-constraint on the field (*i.e.* $\|\mathbf{u}\|^2_{\mathbf{L}^2} = 1$) to prevent the solution $\mathbf{u} \equiv 0$. The vectors of the resulting vector field are then normalized to satisfy the pointwise unit norm constraint. Even if pointwise unit-length is not desired, but no other constraints or alignment is prescribed, the same $\mathbf{L}^2$-constraint would give a meaningful (non zero) solution to the problem of finding a smoothest n-vector field. The solution to the minimization problem

$$\min_{\|\mathbf{u}\|^2_{\mathrm{L}^2}=1} E_H(\mathbf{u})$$

can be readily acquired by finding the smallest eigenvalue to the eigenvalue problem

$$\mathbf{\Delta u} = \lambda \mathbf{u}, \qquad (4)$$

which can be shown using the method of Lagrange multipliers.

Field alignment. Often, a balance between smoothness and the alignment to a given n-vector or direction field $\mathbf{u}'$ is desired. For example, smooth 4-direction fields that align with estimated principal curvature directions, [Cohen-Steiner and Morvan 2003; Hildebrandt and Polthier 2011], are used in applications. For n-vector fields, the following energy can be used to obtain smoothest n-vector fields aligned with some input field $\mathbf{u}'$:

$$E_H(\mathbf{u}) + \mu \left\|\mathbf{u} - \mathbf{u}'\right\|^2_{\mathrm{L}^2}. \qquad (5)$$

In the case of n-direction fields, the norm of the difference of $\mathbf{u}$ and $\mathbf{u}'$ is no longer meaningful since during optimization, the vectors of $\mathbf{u}$ are being freely rescaled as described above. Thus, we want to compare the angular difference between the n-vectors of $\mathbf{u}$ and $\mathbf{u}'$, which can be achieved via the $\mathbf{L}^2$-product for n-vector field. In particular, the following energy will be minimized:

$$E_H(\mathbf{u}) - 2\mu\ \left\langle \mathbf{u}, \mathbf{u}'\right\rangle_{\mathrm{L}^2}, \qquad (6)$$

again subject to $\|\mathbf{u}\|^2_{\mathrm{L}^2} = 1$. Due to the constraint on the $\mathbf{L}^2$-norm, the scalar product of $\mathbf{u}$ and $\mathbf{u}'$ is bounded and minimizing the negative

[1] $\mathbf{\Delta}$ is self-adjoint means: $\langle \mathbf{\Delta u}, \mathbf{v}\rangle_{\mathrm{L}^2} = \langle \mathbf{u}, \mathbf{\Delta v}\rangle_{\mathrm{L}^2}\ \forall \mathbf{u}, \mathbf{v}$. This property is needed to uniquely determine the operator $\mathbf{\Delta}$ in eq. (3).

scalar product favors fields $\mathbf{u}$ that align with $\mathbf{u}'$. The norm of the vectors of the alignment field, *i.e.* $\|u_i'\|$, can be used to control the local weighting of alignment. In particular, in regions with $u_i' = 0$, only smoothness will be taken into account.

So far we have introduced harmonic energies for face-based, piecewise constant n-vector and n-direction fields, and described how smooth fields that align to user specified n-vector or n-direction fields can be acquired by solving sparse linear systems. However, field alignment only gives an approximate tool for designing n-fields as it is oftentimes hard to specify the right weight μ and alignment vectors u_i' such that the field follows the user input satisfactorily. In the following, we will introduce higher-order smoothness energies and describe how interpolation constraints on vectors and directions can be enforced.

5 n-FIELD SPLINES

Splines are fundamental for geometric modeling. The concept of n-field splines we introduce allows building spline-like modeling systems for n-fields. While classical splines are defined for functions over an interval, the variational characterization of splines allows for building counterparts to splines that are defined on more general domains. Examples are spline functions on meshes or tangential vector field splines on meshes. The splines are characterized as the minimizers of a fairness energy subject to constraints. To ensure fairness of the solution under constraints, the order of the fairness energy has to be high enough. For n-fields on surfaces, the order of the harmonic energy derived in the previous section is not high enough to support interpolation constraints. For functions on two-dimensional domains it can been shown that minimizing the harmonic energy subject to interpolation constraints at one (or more) points, yields non-continuous solutions, see [Braess 2007, pp 50–51]. This effect can also be observed in mesh deformation and parametrization, when harmonic energies (or other energies with a second-order Euler-Lagrange equation) and hard interpolation constraints at single points are combined; we refer to [Martinez Esturo et al. 2014] for examples. The same problem shows up in our experiments, an example of this kind is shown in Figure 2. Hence, to enable modeling of n-fields with interpolation constraints, a higher-order fairness energy is needed.

m-harmonic energies. The higher-order energies for n-vector fields, we introduce, are constructed using the Laplacian and the $\mathbf{L}^2$-product for n-vector fields. We define the m-harmonic energies as

$$E_m(\mathbf{u}) = \left\langle \Delta^m \mathbf{u}, \mathbf{u} \right\rangle_{\mathbf{L}^2}. \tag{7}$$

Of particular interest is the biharmonic energy, $m = 2$, which we denote by E_B. The case $m = 1$ yields the harmonic energy. The energies E_m are quadratic and positive definite by construction. The biharmonic energy of a field $\mathbf{u}$ equals the squared $\mathbf{L}^2$-norm of the n-vector field Laplacian Δ of $\mathbf{u}$

$$E_B(\mathbf{u}) = \left\langle \Delta^2 \mathbf{u}, \mathbf{u} \right\rangle_{\mathbf{L}^2} = \langle \Delta \mathbf{u}, \Delta \mathbf{u} \rangle_{\mathbf{L}^2} = \|\Delta \mathbf{u}\|^2_{\mathbf{L}^2},$$

which follows from the self-adjointness of the n-vector field Laplacian. We want to remark that starting from a discrete Laplace operator, m-harmonic energies for functions on meshes have been constructed analogously to our construction of m-harmonic energies for n-vector fields. These higher-order energies are used for example for surface modeling, fairing and deformation. For more background, we refer to the textbook by Botsch et al. [2010], in particular, to Appendix A1.

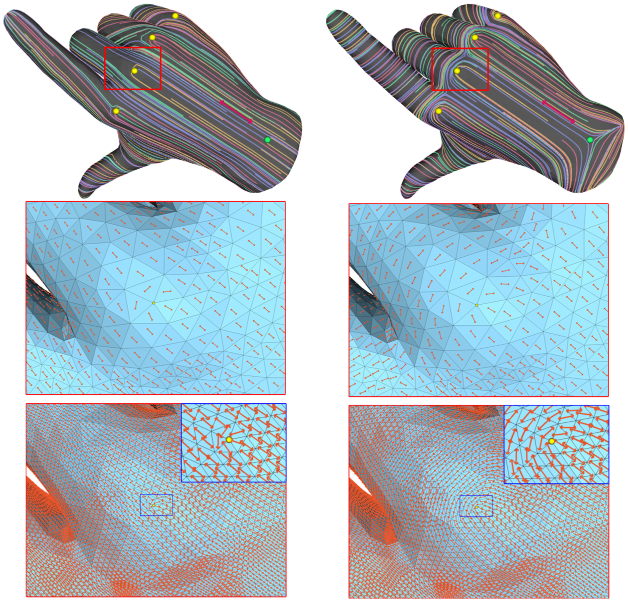

Fig. 2. Placing singularities in a 2-direction field spline by adding hard constraints as described in Section 5. On the left the harmonic energy is used and the hard-constraints lead to a discontinuous solution, which does not converge under refinement (see insets). Using our higher order energy (right) allows the smooth interpolation of such constraints, which remains consistent under refinement.

In the classical setting of spline functions over an interval, minimizers of the harmonic energy under hard constraints yield piecewise linear functions and minimizers of the biharmonic energy are cubic splines. One can also build fairness energies by combining m-harmonic energies for different m. For example, in the classical case, minimizers of the weighted sum of the biharmonic and the harmonic energy are called *splines in tension*. In the following, we will focus on splines defined as minimizers of the biharmonic energy, for simplicity of presentation. Other types of n-field splines can be constructed in the same manner, just the biharmonic energy needs to be replaced by some other m-harmonic energy or a weighted sum of m-harmonic energies.

For interactive modeling, our tool will mainly use interpolation constraints. However, it is often effective to additionally use weak alignment to an existing field as a starting point. Weak alignment constraints to some input field for the splines can be imposed in the same way as describe in the previous section, only E_H is replaced by E_B in equations (5) and (6).

n-vector field splines. For n-vector fields $\mathbf{u}$, an interpolation constraint in some triangle T_i can be enforced by a constraint of the form $u_i = d_i$, where d_i is a prescribed representation vector in T_i. To add or modify interpolation constraints in our interactive modeling system, the users do not work with representation vectors because this would be unintuitive. Instead they select a triangle and specify one vector in the triangle. The system automatically adds the missing $n-1$ vectors. Internally, the system converts the input vector into the corresponding representation vector d_i. However, the representation vectors are only used internally and not shown to the users.

The *n-vector field splines* are defined by the variational problem

$$\begin{aligned} \underset{\mathbf{u}}{\operatorname{argmin}}\ & E_B(\mathbf{u}) + \frac{\beta}{(\operatorname{area}(\mathcal{M}))^2}\|\mathbf{u}-\mathbf{u}'\|^2_{\mathbf{L}^2} \\ & \text{subject to } \ u_i = d_i \text{ for all } i \in I, \end{aligned} \tag{8}$$

where $\mathbf{u}'$ is an alignment field and I is the set of triangle indices for which an interpolation constraint has been specified. The squared area of $\mathcal{M}$ is used as a factor to make the energy invariant to rescaling of the surface. When no alignment field is given, the term $\beta/(\operatorname{area}(\mathcal{M}))^2\|\mathbf{u}-\mathbf{u}'\|^2_{\mathbf{L}^2}$ is removed from the problem. In addition to the interpolation constraints, other types of constraints can be imposed. We discuss singularity constraints below.

n-direction field splines. In the following, we extend the approach to the modeling of n-direction fields. As for the design of n-direction fields with weak constraints, see Section 4, the principle is to minimize over the set n-vector fields with a constraint on the $\mathbf{L}^2$-norm and obtain a n-direction field by pointwise normalization of the vectors of the minimizer. To impose interpolation constraints in this setting, we need to formulate the constraint in a way that only the direction, not the length of the vector, is prescribed. Let d_i be the representation vector of a unit n-vector prescribed as a interpolation constraint in triangle T_i. Then, we constrain u_i to be orthogonal to $J_i d_i$ (where J_i is the clockwise rotation by $\pi/2$ in the tangent plane of triangle T_i), *i.e.* $\langle u_i, J_i d_i\rangle = 0$. The constraint ensures that u_i is collinear with d_i. Since, we are working with representation vectors, we need to avoid the case $u_i = -t d_i$ for some $t > 0$. We prevent this case by using an alignment field $\mathbf{u}'$ with $u'_i = s_i d_i$ for some sufficiently large $s_i > 0$. One wants to choose the s_i as small as possible while still assuring that u_i points to the right direction. Since solving the system is inexpensive, it is valid to perform a binary search for best the parameters s_i. In practice, a large enough constant for all s_i was sufficient in our tests to achieve the correct alignment direction.

The n-direction field splines are defined by

$$\begin{aligned} \underset{\mathbf{u}}{\operatorname{argmin}}\ & E_B(\mathbf{u}) - 2\mu\ \langle \mathbf{u}', \mathbf{u}\rangle_{\mathbf{L}^2} \\ \text{subject to } & \|\mathbf{u}\|_{\mathbf{L}^2} = 1 \\ \text{and } & \langle u_i, J_i d_i\rangle = 0 \text{ for all } i \in I, \end{aligned} \tag{9}$$

where $\mathbf{u}'$ is an alignment field, with modified entries at the hard constraints as described above.

Constraints on singularities. In addition to the interpolation and the weak alignment constraints, further types of constraints can be imposed on the n-field splines. In the following, we describe how we enforce singularities at vertices. Let $T_0, \ldots, T_k$ be the oriented 1-ring of triangles around some vertex v_i. Then, in order to enforce that at v_i we get a singularity with index $m \in \left\{q \cdot \frac{1}{n} \mid q \in \mathbb{Z}\right\}$, we add $k-1$ hard constraints, that enforce the n-vector in T_{j+1} to be the n-vector in T_j, rotated by $\frac{2\pi m}{k}$, for $j = 0, \ldots, k-1$. When we express these constraints in terms of our n-vector field representation $\mathbf{u}$, they have the following form:

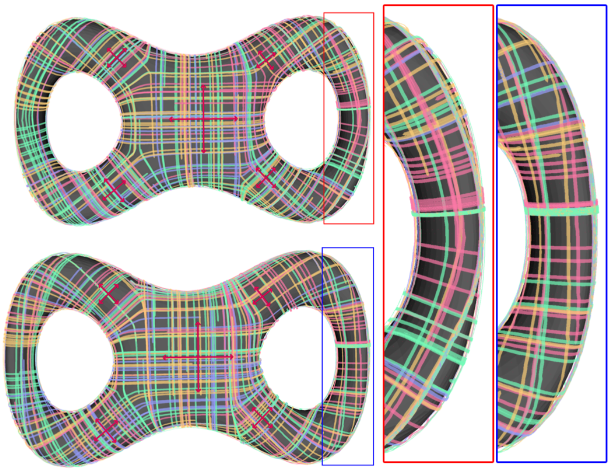

Fig. 3. Comparing the smoothest 4-direction fields on the double torus, in the presence of five hard constraints, when using 4-field splines (top) or when using the harmonic energy (bottom). As can be seen in the highlighted areas, the constraints are interpolated in a differentiable manner when using the biharmonic energy.

$$R_{j+1,j}u_{j+1} - R_{\frac{2\pi m}{k}}u_j = 0. \tag{10}$$

The constraints prescribe a precise rotation of the field around the singularity, and the only degree of freedom left is the magnitude and orientation of one of the n-vectors (which define the rest). This construction is a simple approach ensuring that singular vertices with the desired index are generated. However, it has some limitations. The rotation of the vectors around the singularity is evenly distributed and all u_i on the 1-ring around the singularity are constrained to have the same magnitude. This matches our desire to compute smooth fields but does not have to be optimal in general. On the other hand, imposing the singularity constraint in a general form[2] would be highly non-linear and therefore increase computation times. We want to highlight that imposing hard constraints on the singularities requires higher-order energies as introduced in this paper. Using the common harmonic energy in combination with the constraints (10) leads to degenerated solutions that do not converge to a smooth optimum under refinement. This will be made precise and shown by examples in Section 9.

With this type of singularity control we are able to enforce singularities at any vertex V_i of absolute degree smaller than $k/(2n)$, where k is the number of triangles around the vertex V_i. This limitation is due to the way we define singularities in piecewise constant n-vector fields, see Section 3. As described there, the sum of degrees of all singularities in a field is prescribed by the genus of the underlying mesh. Thus, inserting singularities will have global effects on the field. Alternatively singularity placement can be efficiently controlled by using hard constraints, see Figure 7 for an example.

[2] The most general constraint would force the sum of angular differences between the transported n-vectors to be equal to a value that depends on the chosen singularity index.

Completely defining the topology of a field is not a desired capability of our system since this is orthogonal to the goal of providing intuitive design tools through hard constraints and higher order smoothness.

Finally note, that this type of constraints to control the appearance of singularities is limited to n-fold rotational symmetric fields and has to be adapted individually for any other type of PolyVector field.

6 MATRIX REPRESENTATION

Computing n-field splines amounts to solving linear systems. Before we derive the linear systems, we introduce the matrices representing the m-harmonic energies, the $\mathbf{L}^2$-scalar product and the n-vector field Laplacian. First step is to fix a coordinate system in the plane of every triangle. The matrices act on the vectors $\mathbf{u} \in \mathbb{R}^{2|\mathcal{F}|}$ listing the coordinates of the representation vectors of all triangles. We denote the matrix representing the harmonic energy by $\mathbf{S}$. The matrix is defined as the symmetric matrix that satisfies

$$E_H(\mathbf{u}) = \mathbf{u}^T\mathbf{S}\mathbf{u}$$

for all $\mathbf{u} \in \mathbb{R}^{2|\mathcal{F}|}$. Explicitly, $\mathbf{S}$ is the $2|\mathcal{F}|\times 2|\mathcal{F}|$ matrix which consists of the following 2×2 blocks:

$$\begin{pmatrix} \mathrm{S}_{2i\;2j} & \mathrm{S}_{2i\;2j+1} \\ \mathrm{S}_{2i+1\;2j} & \mathrm{S}_{2i+1\;2j+1} \end{pmatrix} = -w_{ij}R_{ij} \text{ for } i \neq j$$

$$\begin{pmatrix} \mathrm{S}_{2i\;2i} & \mathrm{S}_{2i\;2i+1} \\ \mathrm{S}_{2i+1\;2i} & \mathrm{S}_{2i+1\;2i+1} \end{pmatrix} = \sum_{k\in\mathcal{N}(i)} w_{ik}\mathbf{I}.$$

The matrix $\mathbf{M}$ representing the $\mathbf{L}^2$-scalar product is defined as the matrix that satisfies $\langle \mathbf{u}, \mathbf{u}' \rangle_{\mathbf{L}^2} = \mathbf{u}^T\mathbf{M}\mathbf{u}'$ for all pairs $\mathbf{u}, \mathbf{u}' \in \mathbb{R}^{2|\mathcal{F}|}$. Explicitly, it is the diagonal $2|\mathcal{F}|\times 2|\mathcal{F}|$ matrix that repeats the area of each triangle twice. This matrix is often called the mass matrix. The matrices representing the Laplacian and the m-harmonic energies can be constructed as products of $\mathbf{S}$ and $\mathbf{M}$. The matrix $\mathbf{L}$ for the n-vector field Laplacian $\mathbf{\Delta}$ is given by $\mathbf{L} = \mathbf{M}^{-1}\mathbf{S}$. The matrix $\mathbf{B}$ representing the biharmonic energy is

$$\mathbf{B} = \mathbf{S}\mathbf{M}^{-1}\mathbf{S},$$

and, more generally, the matrices for the m-harmonic energies are: $\mathbf{S}(\mathbf{M}^{-1}\mathbf{S})^{m-1}$. The matrices representing the scalar product and the m-harmonic energies are positive definite.

For later usage, we also introduce the 2×2 matrix that performs a rotation by $\pi/2$ in the plane of a triangle. Since we use positively oriented orthonormal bases in the tangent planes, the rotations have the same matrix representation in all tangent planes:

$$J_i = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.$$

7 COMPUTING n-FIELD SPLINES

We now state the linear systems whose solutions are field-aligned, interpolating, higher-order n-vector or n-direction field splines. In the following, let $I = i_1, \ldots, i_m$ be the set of hard constrained vectors, $u_i = d_i$, let $\mathbf{u}'$ be the specified alignment field and μ the alignment weight. $\mathbf{d}$ is the vector that stacks real and imaginary parts of the constrained directions $d_{i_1}, \ldots, d_{i_m}$.

n-vector fields. Let $\mathbf{F}$ be the $2m\times 2|\mathcal{F}|$ matrix for which $\mathbf{Fu} = \mathbf{d}$ is equivalent to $u_i = d_i$ for $i \in I$. Then the optimization problem (8) is equivalent to solving

$$\begin{pmatrix} \mathbf{B}+\mathbf{M} & \mathbf{F}^T \\ \mathbf{F} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{M}\mathbf{u}' \\ \mathbf{d} \end{pmatrix} \tag{11}$$

where the vector $\boldsymbol{\lambda}$ stacks the $2m$ (rescaled) Lagrange multipliers.

n-direction fields. We first modify the alignment field in order to enforce the correct directions in the hard constrained faces, as described before, *i.e.* we set $u_i' = s_i d_i$ for $i \in I$, while leaving the rest of the alignment field as is. Let $\mathbf{D}$ be the $m\times 2|\mathcal{F}|$ matrix for which $\mathbf{Du} = 0$ is equivalent to $\langle u_i, J_i d_i \rangle = 0$ for all $i \in I$. Then a minimizer of (9) can be found by solving the linear system

$$\begin{pmatrix} \mathbf{B}-\lambda\mathbf{M} & \mathbf{D}^T \\ \mathbf{D} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ \boldsymbol{\gamma} \end{pmatrix} = \begin{pmatrix} \mathbf{M}\mathbf{u}' \\ 0 \end{pmatrix}, \tag{12}$$

where λ takes the role of the parameter μ and has to be chosen in the range $\left(-\infty, \hat{\lambda}_1^2\right)$, where smaller λ means higher alignment. $\hat{\lambda}_1$ is the smallest eigenvalue of the n-vector field bi-Laplacian $\mathbf{\Delta}^2$ restricted to fields which satisfy the hard constraints. This reformulation is akin to [Knöppel et al. 2013]. A proof and a more precise statement regarding $\hat{\lambda}_1$ is included to the appendix. The auxiliary variable $\boldsymbol{\gamma}$ stacks the m (rescaled) Lagrange multipliers. Once $\mathbf{w}$ is computed, the desired n-direction field spline $\mathbf{u}$, which interpolates the directions d_i in triangles T_i and aligns to the field $\mathbf{u}'$, is obtained by normalizing all vectors of $\mathbf{w}$: $u_i = \frac{w_i}{\|w_i\|}$. The validness of this system and the relationship between λ and μ is discussed in the appendix.

ALGORITHM 1: Computation of n-direction field splines

```
Input  : The mesh $(\mathcal{V}, \mathcal{E}, \mathcal{F})$, the rotational symmetry index $n$, an
         alignment field $\mathbf{u}'$, the alignment parameter $\lambda$, the list of hard
         constraints $d_i$ along with the indices of constrained triangles $I$
Output : $n$-direction field spline interpolating the hard constraints and
         aligning to $\mathbf{u}'$
First solve:
    (1) Modify the alignment field at the hard constraints:
        $\forall\ i \in I:\ u_i' \leftarrow s_i d_i$, for sufficiently large $s_i$.
    (2) Choose a basis for the tangent space in each triangle and compute
        the rotation matrices $R_{ij}$ as described in Section 3.
    (3) Set up the matrices $\mathbf{S}$, $\mathbf{M}$, $\mathbf{L}$ and $\mathbf{B}$ as described in Section 6 and the
        matrix $\mathbf{D}$ which stacks the constraints $\langle u_i, J_i d_i \rangle = 0$.
    (4) Factorize the matrix $\mathbf{V}_\lambda = \mathbf{B} - \lambda\mathbf{M}$.
    (5) Solve $\mathbf{D}\mathbf{V}_\lambda^{-1}\mathbf{D}^T\boldsymbol{\gamma} = \mathbf{D}\mathbf{V}_\lambda^{-1}\mathbf{M}\mathbf{u}'$ by using the factorization above to
        compute the matrix $\mathbf{V}_\lambda^{-1}\mathbf{D}^T$ and the vector $\mathbf{V}_\lambda^{-1}\mathbf{M}\mathbf{u}'$.
    (6) Use the same factorization and $\gamma$ from above to compute
        $\mathbf{w} = \mathbf{V}_\lambda^{-1}\left(\mathbf{M}\mathbf{u}' - \mathbf{D}^T\gamma\right)$.
    (7) The $n$-direction field spline $\mathbf{u}$ is now given by $u_i = \frac{w_i}{\|w_i\|}$.
Updated constraints:
    (1) Only compute the columns $\mathbf{V}_\lambda^{-1}\mathbf{D}^T$ for the rows of $\mathbf{D}$ that are new or
        updated.
    (2) Recompute the multiplications $\mathbf{D}\mathbf{V}_\lambda^{-1}\mathbf{D}^T$ and $\mathbf{D}\mathbf{V}_\lambda^{-1}\mathbf{u}'$ and solve the
        two systems from steps (5) and (6) above.
    (3) Output the $n$-direction field spline $u$ as given in (7) above.
```

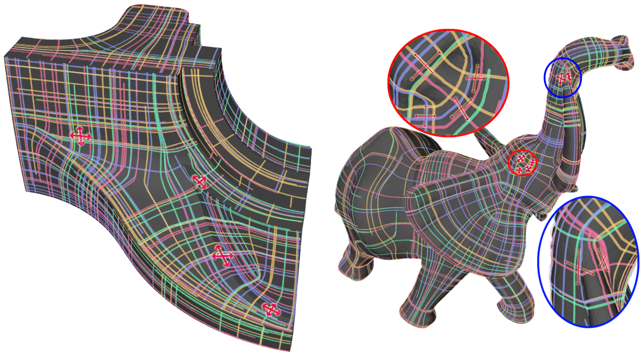

Fig. 4. Left: By adding four hard constraints, we are able to force the smoothest, curvature aligned ($\lambda = -0.5$) 4-field to align to features that are not following the principle curvature directions. Right: Hard constraints enable precise editing at small scale: the insets show how hard constraints can be used to align to small features on a high resolution mesh with fine details. Such precise control can not be achieved via least-squares alignment terms but requires hard constraints.

For both, n-vector and n-direction fields, the constraints (10) to enforce singularities can be readily appended to the matrices $\mathbf{D}$ and $\mathbf{F}$ to solve for smoothest constrained n-vector or n-direction field splines respectively.

Implementation. In our implementation the user can add and modify hard constraints by selecting a face and dragging an arrow to specify a desired direction. The optimal field is updated each frame depending on the current configuration of the constraints. This allows for instant feedback when trying to adjust hard constraints in the design of n-field splines. Note that when an existing constraint is modified, in the case of n-vector fields, only the right hand side changes, so the full system (11) can be factorized once and solved with the modified constraints. However, once new constraints are introduced, the matrix $\mathbf{F}$ changes (new rows have to be added) and the matrix has to be refactorized. In the case of n-direction fields, the hard constraints are encoded in the matrix $\mathbf{D}$, so changing and adding hard constraints both would lead to refactorization. To avoid factorizing the large matrices in both systems every time constraints are added or modified, we reorganize the equations. We will demonstrate the reformulation on system (12), the steps to reformulate system (11) are identical.
Let $\mathbf{V}_\lambda = \mathbf{B} - \lambda\mathbf{M}$ and multiply the first $2|\mathcal{F}|$ rows of the system by $\mathbf{D}\mathbf{V}_\lambda^{-1}$ from the left, which amounts to the system

$$\mathbf{D}\mathbf{w} + \mathbf{D}\mathbf{V}_\lambda^{-1}\mathbf{D}^T\boldsymbol{\gamma} = \mathbf{D}\mathbf{V}_\lambda^{-1}\mathbf{M}\mathbf{u}'$$
$$\mathbf{D}\mathbf{w} = 0$$

and by subtracting the second set of equations from the first we get

$$\mathbf{D}\mathbf{V}_\lambda^{-1}\mathbf{D}^T\boldsymbol{\gamma} = \mathbf{D}\mathbf{V}_\lambda^{-1}\mathbf{M}\mathbf{u}' \tag{13}$$

where the left-hand side is still symmetric. We solve (13) for $\boldsymbol{\gamma}$ and then recover $\mathbf{w}$ via

$$\mathbf{w} = \mathbf{V}_\lambda^{-1}(\mathbf{M}\mathbf{u}' - \mathbf{D}^T\boldsymbol{\gamma}). \tag{14}$$

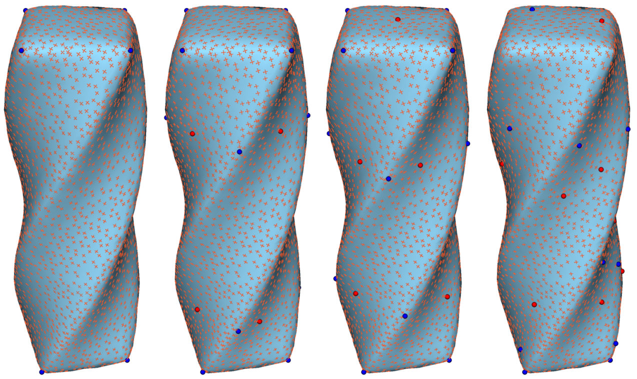

Fig. 5. Eigenvectors of the 4-field Laplacian. From left to right: 1st, 6th, 14th and 110th eigenfield. Singularities with indices $-\frac{1}{4}$ and $\frac{1}{4}$ are marked in red and blue respectively.

The advantage of this is that we only have to factorize $\mathbf{V}_\lambda$ once, and whenever new constraints are added or modified, we only have to solve the dense but very small ($2m \times 2m$) system (13). To set up the left hand side of (13) we also need to solve a linear system to obtain updates to those rows of $\mathbf{V}_\lambda^{-1}\mathbf{D}$, which have to be changed due to updated hard constraints. This can be done using the factorization of $\mathbf{V}_\lambda$. Note, that in practice hard constraints will not be edited simultaneously, such that only one column of $\mathbf{D}$ is being modified. Then, since we need to update the alignment field when we edit hard constraints, we need to compute the right hand side $\mathbf{D}\mathbf{V}_\lambda^{-1}\mathbf{u}'$. After solving (13), we solve for $\mathbf{w}$ in (14) using the same factorization. Timings when solving for a 4-direction field with 30 hard constraints can be taken from Table 1 (right-most column). Note that those timings were computed as only one hard constraint has been modified. They include setting up and solving the dense system (13) and then recovering $\mathbf{w}$ via (14). The time needed to factorize the matrix $\mathbf{V}_\lambda$ is listed separately since this is done at the preprocessing stage and only needs to be repeated when the parameter λ is changed. As can be seen, for small meshes this reformulation may be used in a real-time editing setting, however, for larger meshes (or in case the parameter λ needs to be changed interactively, in order to control the strength of alignment), we propose a change of basis, under which the matrix $\mathbf{V}_\lambda$ diagonalizes. This will be described in the following section.

Algorithm 1 summarizes the computation of n-direction field splines.

8 REAL-TIME EDITING

In this section we describe our approximation algorithm that allows for computing n-field splines in real-time, independently of the mesh resolution. We want to emphasize that this is an optional step as the full systems can alternatively be solved. The benefit of using the fast computation is that it ensures a fluid interaction and allows users to immediately get feedback on how the constraints they are placing affect their design. Screen captures comparing the user-interaction with and without using the approximation algorithm is shown in the supplementary video. As a trade-off to a faster computation

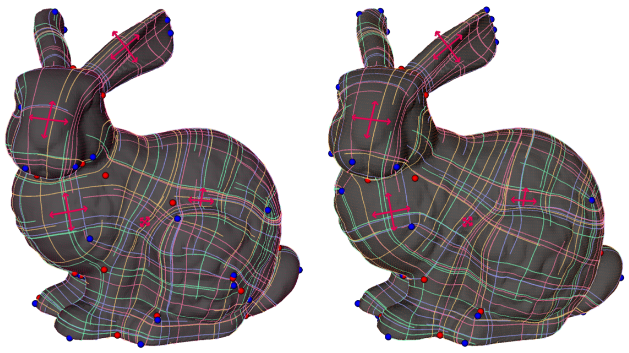

Fig. 6. Visual comparison of the smoothest, curvature aligned ($\lambda = -0.05$), 4-direction field on the bunny, subject to five hard constraints (pink crosses), when computing it using the full system (12) (left) and the reduced system (17) (right), using a basis of 500 eigenfields. The fields possess 72 and 68 singularities respectively.

during a modeling session, a pre-computation has to be performed, in which a basis for n-vector fields is computed. Timings for both pre-computation and the reduced/unreduced solves for the n-field spline system are listed in Table 1 and discussed in Section 9.

Using the eigenbasis of the n-vector field Laplacian $\mathbf{\Delta}$, we obtain a Fourier-type decomposition of face-based, piecewise constant n-vector fields that associates a frequency spectrum to a n-field. The eigenfields and spectrum of $\mathbf{\Delta}$ are the solutions of the generalized eigenproblem

$$\mathbf{S}\mathbf{u} = \lambda\mathbf{M}\mathbf{u}. \tag{15}$$

Since $\mathbf{\Delta}$ is self-adjoint, all eigenvalues are real the there exist an $\mathbf{L}^2$-orthonormal eigenbasis. The eigendecomposition enables the design of spectral processing tools, such as compression and spectral filtering, for n-fields. For example, projecting a n-field to the subspace spanned by the k eigenfields with the lowest eigenvalues, is a low-pass filter for the n-field. Here, we will not explore this direction—but use the low-frequency eigenfields to derive a reduced-order scheme for the fast approximation of n-field splines. The computation is split in an offline and an online stage. In the offline stage, the first k eigenfields are computed and the relevant matrices are constructed. In the online stage, the computations are restricted to the subspace spanned by the eigenfields. This results in a reduced computational burden in the online phase and enables real-time computation and interactive modeling of n-field splines for larger meshes.

In the following, we discuss the reduction of equations (11) and (12), which yields fast approximation algorithms for n-vector and n-direction field splines. Let $\mathbf{U}$ be the $2|\mathcal{F}| \times k$ matrix which stacks the first k eigenvectors as its columns and let λ_i denote the first k eigenvalues. Any n-field $\mathbf{w_U}$ that is in the subspace spanned by $\mathbf{U}$ can be describe by reduced coordinates $\mathbf{w}$, which are given by $\mathbf{w_U} = \mathbf{U}^T\mathbf{w}$. Since the eigenfields are $\mathbf{L}^2$-orthonormal, $\mathbf{U}^T\mathbf{M}\mathbf{U} = \mathbf{I}$, we have

$$\mathbf{U}^T\mathbf{V}_\lambda\mathbf{U} = \mathbf{U}^T\left(\mathbf{B} - \lambda\mathbf{M}\right)\mathbf{U} = \operatorname{diag}\left(\lambda_i^2 - \lambda\right) =: \mathbf{\Lambda}_\lambda. \tag{16}$$

This means that solving the sparse $2|\mathcal{F}| \times 2|\mathcal{F}|$ system (14) to obtain a smooth n-vector field in the unreduced case (which has to be done at least three times when constraints are updated) is being replaced by the diagonal system $\mathbf{\Lambda}_\lambda\mathbf{x_U} = \mathbf{w_U}$, which can be solved by k multiplications with precomputed numbers. The reduced version of (13) is

$$\mathbf{D_U}\mathbf{\Lambda}_\lambda^{-1}\mathbf{D}_\mathbf{U}^T\boldsymbol{\gamma} = \mathbf{D_U}\mathbf{\Lambda}_\lambda^{-1}\mathbf{M_U}\mathbf{u}'_\mathbf{U}, \tag{17}$$

where $\mathbf{u}'_\mathbf{U} = \mathbf{U}^T\mathbf{u}'$, $\mathbf{D_U} = \mathbf{U}^T\mathbf{D}\mathbf{U}$ and $\mathbf{M_U} = \mathbf{U}^T\mathbf{M}\mathbf{U}$, which can be updated efficiently when $\mathbf{D}$ changes since it is extremely sparse. Finally, $\mathbf{w_U}$ can be recovered via

$$\mathbf{w_U} = \mathbf{\Lambda}_\lambda^{-1}\left(\mathbf{M_U}\mathbf{u}'_\mathbf{U} + \mathbf{D}_\mathbf{U}^T\gamma\right). \tag{18}$$

Using this reduction, the cost for solving for n-field splines is independent of the resolution of the meshes, aside from multiplications with $\mathbf{U}$. By storing $\mathbf{U}$ on a GPU and computing multiplications there, we computed n-field splines in real time for meshes with 300k triangles in our experiments (see Table 1).

9 EXPERIMENTS

Real-time n-field spline editor. Using the higher-order energies, together with the reformulations of the involved linear system and spectral reduction described in the previous sections, we implemented a tool for real-time editing of n-field splines. It allows users to click and drag on a mesh to insert hard constraints and specify their direction, while the field and its visualization are updated in real-time. As a visualization we chose to draw lines along the surface that follow one of the n directions along the n-field. To keep the visualization consistent, we initialize seed points for those lines once and re-integrate them whenever the n-field spline has changed. All examples that are shown in this paper have been created using this tool. We also show several editing sessions and features of the tool in the supplementary video. In our experience, the precise control offered via hard constraints, combined with the instant feedback due to the reductions offer an intuitive, effective and persistent way of n-field editing that has many advantages over previous methods. A feature of the propose technique is that the output depends continuously on the constraints, which allows users to slightly perturb constraints in order to optimize features of the field on a small scale in an intuitive way. Below we discuss various aspects of the editing framework described in this paper along with comparisons to previous work.

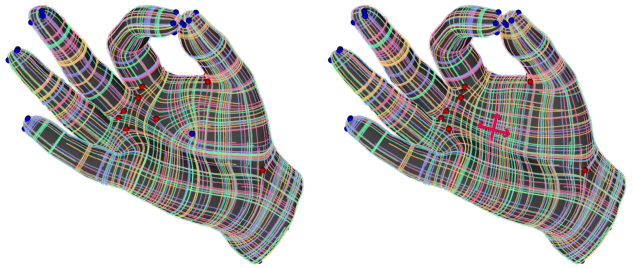

Fig. 7. The smoothest, curvature aligned ($\lambda = -0.2$), 4-direction field on the hand (left) is intuitively modified by merging two singularities using a hard constraint (right).

Basic examples. In Figures 7, left, 6 and 12, top row, we show curvature aligned, smoothest 4-direction fields. Singularities, which are automatically optimized and not user input or achieved via integer variables, appear in desirable places, such as cube-like corners (right, upper part of the rocker arm in Figure 12) or at the end of extremities, in groups of four (see Figure 7 at the end of the fingers), which allows for the flow lines to follow the features until forming a plateau at the end. The results strongly resemble those obtained from the formulation of Knöppel et al. [2013], which is formulated for vertex-based fields. Different from their approach, we allow for higher order energies and hard constraints, *i.e.* *n*-field splines. In Figure 4, right, we show how this can be used to edit small scale details on a high resolution mesh with exact control over the alignment. When using soft constraints (*i.e.* only an alignment field) to control the layout of the field, we were unable to enforce alignment satisfactorily, especially in areas where conflicting constraints are close to each other: either one of the conflicting constraints dominates the shape of the field in that region, or the field averages the influence of the nearby constraints. When using hard constraints, we can easily force the field to point in various directions (even for directly adjacent triangles) and still obtain a smooth solution for the rest of the field. In Figure 4, left we edit the smoothest, curvature aligned 4-field by adding three hard constraints which force the flow lines to follow two features that do **not** follow the principal curvature directions. It is intuitive to move, merge or produce new singularities by placing appropriate hard constraints, a simple example for this is shown in Figure 7. Direct control over the placement of singularities at desired vertices is also possible and is treated in Section 5 and demonstrated in an experiment further below.

In Figure 5 we show some of the eigenfields of the 4-field Laplacian, Δ, which are used as a reduced basis for interactive editing. It is remarkable how the singularities are laid out symmetrically and appear consistently (for the first 108 eigenfields) at the corners of the twisted, rounded bar. This indicates that a process that forces the field not to place singularities at the corners will result in less smooth fields. Therefore, we think that the pure number of singularities of field is not a reliable indicator for the smoothness of a field.

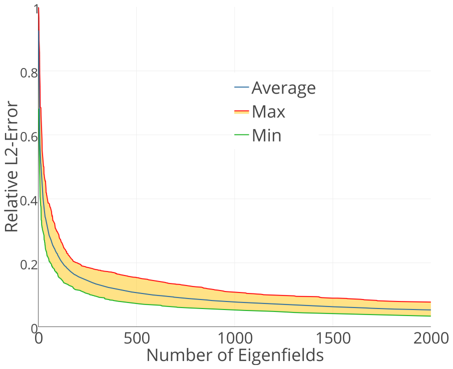

Fig. 8. Plots of the average, minimal and maximal relative $\mathbf{L}^2$-error when comparing the reduced solution (12) to the full solution (17) of a 4-direction field on the bunny mesh from ten randomized constraints (we generated twenty tests) using a rising number of eigenfields.

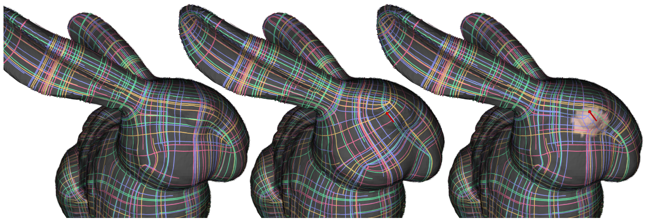

Fig. 9. Local editing on the bunny mesh: to the smoothest (biharmonic energy), curvature aligned, 4-direction field (left) we want to add a hard constraint to achieve diagonal quads around the eye. However, the constraint has an undesired non-local effect (middle). This can be remedied by using our local editing scheme, which allows for local changes to the field, while still yielding differentiable results (right).

The effects of using a reduced basis are shown in Figure 6 where the smoothest, curvature aligned 4-field, altered by adding five hard constraints, is shown as an optimal solution of the unreduced (left) and reduced (right) system. In the reduced case a basis of the 500 lowest frequency eigenfields was constructed. While the general layout of the field remains the same, some singularities moved and merged, which is due to the fact that the hard constraints and curvature alignment introduced high-frequency features, which are not contained in the reduced basis.

As a numerical comparison, we generated twenty different 4-direction fields from a random set of twenty hard constraints on the bunny mesh using the unreduced and reduced systems and kept track of the relative $\mathbf{L}^2$ distance between both solutions when using between one and 2000 eigenfields in the reduced case. In Figure 8 we plot the minimal, average and maximal $\mathbf{L}^2$ distances between each reduced and unreduced field generated this way. The number of eigenfields that should be used depends on the application. In order to keep the relative distance to a full solution consistently below 10%, at least 1000 eigenfields should be used. We found, however, that about 500 eigenfields are enough to achieve fields that are visually hard to distinguish from the unreduced solution. Thus the proposed model reduction is well-suited as a technique that enables interactive modeling of *n*-fields on larger meshes.

n-field splines. By introducing higher order smoothness energies for *n*-fields, we get the guarantee of continuous differentiability at constraints. To highlight this, we show an artificial example in Figure 10, where we constrain a complete region of the 4-vector field (lower part, in blue), while solving for the smoothest field under these constraints using the harmonic energy E_H, left, and the biharmonic energy E_B, right. As can be seen, using the harmonic energy results in a constant field on the upper half. The tendency of the field in the lower half to grow in magnitude and the clockwise rotation are not continued, which corresponds to a discontinuity in the derivative in the continuous setting. In fact, changing the blue part of the field does not have any effect on the variable, upper part, as long as the vectors in the boundary triangles remain the same.

Using the biharmonic energy smoothly continues the tendencies of the lower field.

This enables us to perform tasks such as local editing, see Figure 9: since constraints are interpolated in a differentiable manner, it is valid to constrain the field on the whole mesh minus a small selected region. This is relevant in practice since hard and soft constraints have a global influence on the layout of the field, which is undesirable when editing small features in isolated regions. For computational efficiency, one can reduce the size of the system to be solved by discarding vectors that are constrained and do not affect the unconstrained vectors. Explicitly, the vectors of triangles that are not in the two-ring of any unconstrained triangle need not be included. Here two-ring refers to the dual graph in which the triangles are the nodes and nodes are connected if triangles share an edge. In this sense, the two-ring around the unconstrained triangles specifies boundary conditions which determine the solution in the unconstrained area.

In Figure 2 we show n-direction field splines where we prescribed the placement of four index $\frac{1}{2}$ (yellow) and one index $-\frac{1}{2}$ singularities in the field as described in Section 5. The main observation is that these types of hard constraints are not well posed under the harmonic energy. The constraints only affect the vectors in the one-rings around the placed singularities. Since a one-ring becomes smaller and smaller under refinement, the solutions converges to a discontinuous field. This is illustrated in the insets in the lower row, where the same hard constraints were used in a low and high resolution version of the hand mesh. Using the biharmonic energy leads to a smooth solution, where the singularity constraints have a global effect and the solution remains consistent under refinement. This is another example for the importance of higher order smoothness energies when posing hard constraints.

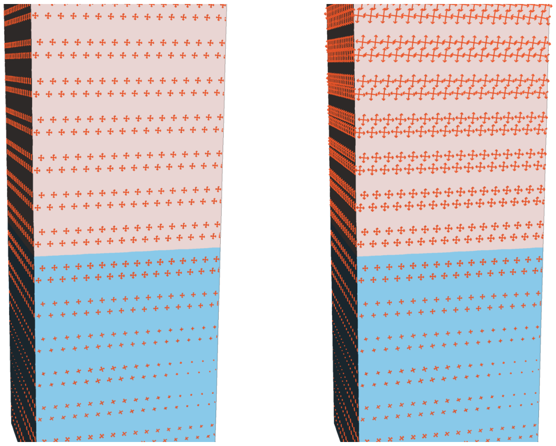

Fig. 10. Comparing the harmonic (left) and biharmonic (right) energies in presence of boundary constraints: the blue part of the field was fixed and the rest of the field computed as smooth as possible, as defined through the different energies. As can be seen, the higher order energy continues the clockwise rotation and increase in size (*i.e.* the first derivative remains continuous), while the harmonic energy only ensures continuity of the field and thus remains constant.

The tendency to interpolate constraints in a differentiable manner when using n-field splines can also be seen in Figure 3, where the curvature of the flow lines near the constraints is continued in the regions on the side, away from any constraints, which is not the case when using the common harmonic energy. Note that in this example we did not use curvature alignment.

Timings. In Table 1 we list the timings taken to solve the reduced and unreduced systems to compute smoothest n-fields subject to 30 hard constraints, using the harmonic energy and biharmonic energy. We constructed 4-direction fields, and used the 500 lowest frequency eigenfields of the corresponding Laplacian as a reduced basis. The timings to construct those bases is listed as well. When measuring the computation time, only one of the 30 hard constraints was assumed to have changed in the last frame, such that only specific parts of the system have to be set up and updated. The reduction leads to computation timings that are neither influenced by the resolution of the mesh, nor by the order of the energy. The field can be updated about 200 times per second, such that the visualization of the field and display of the mesh become the bottle-neck for real-time n-vector field editing. In the unreduced case, we obtain computation times of more than 100 ms for meshes with 70k triangles in our experiments. Times for adding or removing new constraints are even longer. The trade-off for the fast computation times in the reduced scheme is that the eigenfields need to be precomputed, which takes about 3 minutes for meshes with 70k triangles in our experiments.

Comparison to Instant Field Aligned Meshes. Jakob et al. [2015] introduced a scheme for computing smooth n-fields via a local averaging scheme on a multiresolution hierarchy and parallelization in very low computation times. Since this is the only technique that results in comparable timings, we want to put our framework into contrast.

The techniques used in their approach are fundamentally different from the methods we propose. Our fields are globally optimal

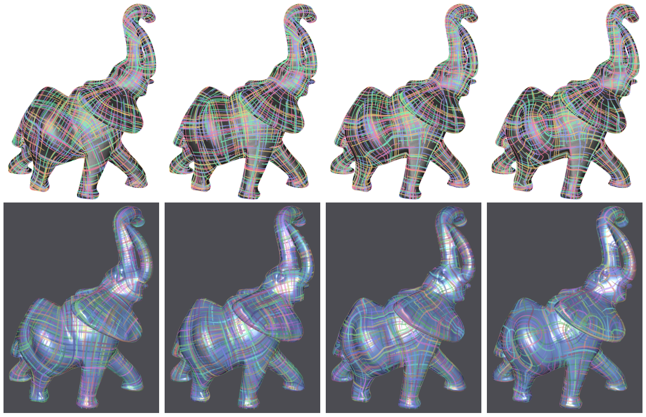

Fig. 11. Curvature alignment using our framework (top row) and [Jakob et al. 2015] (bottom row). We used the bihamornic energy and, from left to right, $\lambda = 0.0001, 0, -0.005, -0.1$. Bottom row, left, shows the result when using the extrinsic energy, and then, from left to right, alignment via local averaging using interpolation weights of 0.01, 0.1 and 0.5.

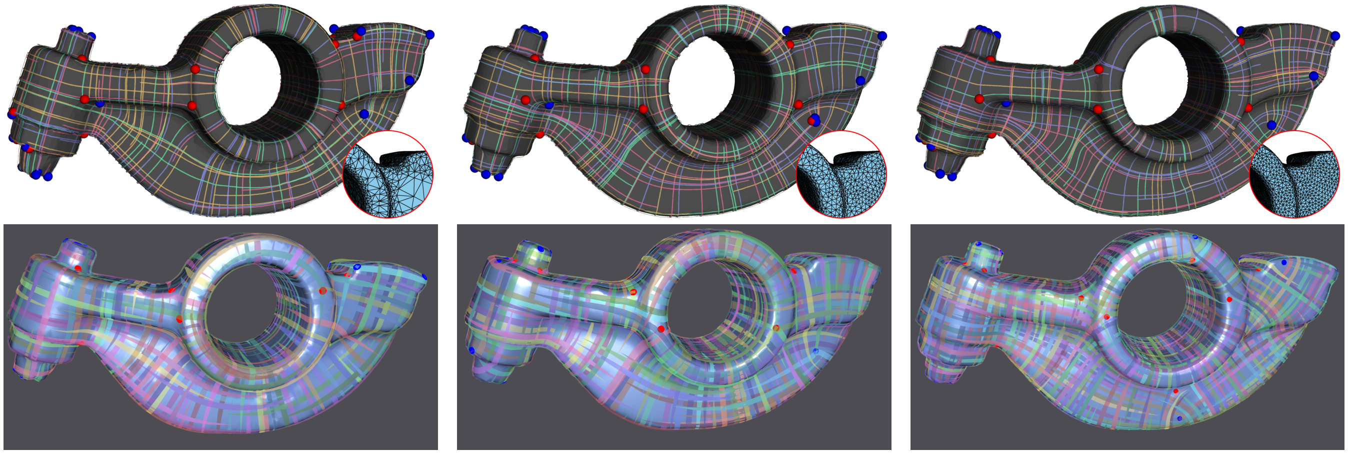

Fig. 12. Top row: The smoothest curvature aligned 4-fields on three different triangulations, computed using our system. The fields possess 51, 40 and 42 singularities, respectively, different numbers mostly resulting from merged groups of nearby singularities. Bottom row: The smoothest 4-fields, using the extrinsic energy and the system proposed by Jakob et al. [2015]. The fields possess 22, 30 and 42 singularities, respectively, being arranged in inconsistent structures.

and are solutions of linear systems, while [Jakob et al. 2015] use parallelization, a multiresolution hierarchy and local operations to iteratively optimize the fields. Their approach does not iterate until a minimum of an objective is found, but terminates after a number of local smoothing steps on each level of the hierarchy have been performed. In our experiments with their approach, we detected multiple drawbacks resulting from this strategy when compared to our approach. Their results are heavily affected by a changing the triangulation of a surface: Figure 12 shows the smoothest, curvature aligned 4-fields produced with our technique (top row) and the smoothest fields produced by Jakob et al. on three different triangulations of the rocker arm mesh. While our singularity layout remains consistent (only some pairs of very close singularities merge and cancel each other), the fields produced by Jakob et al. possess inconsistent and globally different singularity layouts.

In [Jakob et al. 2015], alignment to a globally defined field, such as principal curvature directions, is handled via linearly interpolating the current n-vectors with the vectors of the alignment field in every optimization step. Since this has to be done on every level of the multiresolution hierarchy, the constraint vectors have to be propagated along the hierarchy as well, by iteratively merging the constraint vectors. It is not clear whether such an optimization scheme minimizes a smoothness/alignment energy such as (9). As an alternative to achieve curvature alignment, Jakob et al. propose to minimize an extrinsic smoothness energy by averaging the n-vectors in 3D world coordinates. However, this approach does not allow for a weight between smoothness and alignment. In Figure 11, we show a comparison between curvature aligned fields using our framework with the biharmonic energy and different weights for λ and fields using the framework of [Jakob et al. 2015], where we show both the extrinsically smooth field and fields produced by using the alignment method described above, using different interpolation weights. While the extrinsically smooth field has comparable quality to our field for $\lambda = 0.0001$, it is not aligned to many features of the elephant, such as the eye, or the right part of the visible ear. There is no way to enhance the alignment when using the extrinsic energy. Using the local averaging scheme to align to the principal curvature directions produces visibly unsmooth fields, when the interpolation weights are high enough to guarantee alignment to the aforementioned features. Our technique maintains global smoothness while aligning to more features when using higher values of λ.

Table 1. Timings for the computation of smoothest, 4-direction fields without and with using the proposed spectral reduction. In all examples we added 30 random hard constraints. We state timings for both the harmonic and biharmonic smoothness energy. In the reduced case, we construct a basis from the 500 lowest frequency eigenfields.

Model Name	#faces	Bases setup	Reduced Setup & Solve harmonic \biharmonic	Factorization of Δ \ Δ^2	Unreduced Solve harmonic \biharmonic
Twisted Bar	3276	5s	5ms \ 5ms	40ms \ 83ms	6ms \ 6ms
Hand	12184	24s	5ms \ 5ms	161ms \ 393ms	16ms \ 23ms
Rocker Arm	20088	44s	5ms \ 5ms	267ms \ 654ms	28ms \ 37ms
Bunny	69666	170s	6ms \ 6ms	1318ms \ 4413ms	138ms \ 194ms
Elephant	79946	181s	6ms \ 6ms	1249ms \ 3112ms	155ms \ 182ms
Armadillo	331904	15m30s	8ms \ 8ms	8155ms \ 28s	598ms \ 801ms

Finally, we want to remark that since there are no guarantees on the optimality of solutions and because of the necessity to merge n-vectors when navigating the multi-resolution hierarchy in [Jakob et al. 2015], one cannot expect a continuous dependence of the solutions on the constraints. This is a key feature of a modeling tool. In our framework, slightly changing the constraints leads to small changes in the field.

10 APPLICATIONS

In the following we will describe two applications that strongly benefit from our ability to compute n-field splines in real-time in order to aid with artistic tasks, which require immediate feedback and smooth alignment to user defined hard constraints.

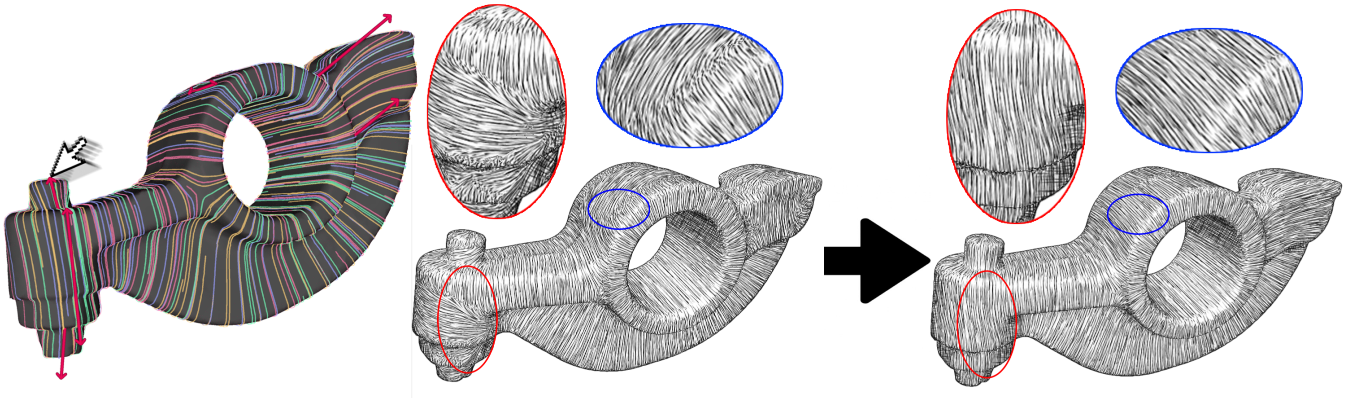

Fig. 13. **Left:** Screenshot of a modeling session of a 2-field spline using hard constraints. The constraints are edited via dragging the arrows and the field is updated and visualized at 60fps. **Middle:** Non-photorealistic rendering of the rocker arm mesh when using the *unedited* smoothest 2-field aligned to the maximal principal curvature. **Right:** Hatching rendering of the *edited* 2-field spline subject to six hard constraints.

Hatching. Rendering a surface mesh to resemble a line drawing, often called hatching, is a popular technique for stylizing 3D scenes, cf. [Kalogerakis et al. 2012; Kim et al. 2008; Praun et al. 2001; Suarez et al. 2016; Umenhoffer et al. 2011]. In order to perform such a rendering, directions on the surface have to be chosen, along which the lines can be drawn. Often, as in [Kim et al. 2008; Praun et al. 2001], these directions are somehow extracted from the principal curvature directions. Principal curvature directions are not unique in flat or umbilic regions, so directions in such regions need to be post-processed. Treating principal curvature directions as direction fields, allows to smooth the directions and to impaint directions in umbilic regions. For such smoothing processes, the principal curvature directions can be treated as 4-direction fields, see [Palacios and Zhang 2007]. However, for guiding stroke directions, we opt for 2-direction fields, since it is the natural choice for the regions in which strokes are being drawn in only one direction and not orthogonally.

Using the principal curvature directions as the stroke directions is just one of many choices and often it is desirable to design stroke directions from scratch or modify existing directions, for example to hide singularities in highly lit or occluded places. This leads to a direct application of our real-time framework for the design maximal curvature aligned 2-direction field splines, which can be enhanced via hard constraints. To illustrate this we implemented a real-time hatching system based on [Kim et al. 2008], which uses tonal art maps to texture surfaces according to light intensity and direction information. We enable the user to design a 2-direction field using our standard set of tools and directly update the directional information for the hatching application in real-time, which provides the user with an instant result. We show a comparison between the 2-direction field aligned to the maximal principal curvature directions and an edited 2-direction field spline subject to six hard constraints. It can be seen how alignment to curvature directions is not suitable in regions where many singularities are present or the maximal/minimal directions suddenly exchange their roles. We provide the ability to intuitively edit the field in such regions in order to enhance the quality of the rendering. We refer to the supplementary video for several examples of this application.

Anisotropic BRDF. In physically-based rendering systems one describes the reflection behavior of an object by using a *bidirectional reflectance distribution function* (BRDF) [Nicodemus 1977]. The BRDF at a given point on a surface depends on the direction towards the light and the direction towards the viewer and returns the color and intensity of the perceived light. Many such functions can be designed and they result in different perceptions of an objects material. Often, materials can be considered isotropic, meaning that their value does only depend on the angle between viewing/light direction and the surface normal. However, for anisotropic materials, like brushed metals, parameters have to be defined across the surface which define the orientation dependent response of the material to light. In Ashikhmin et al. [2000] a BRDF model is proposed that takes (amongst others) two parameters n_u and n_v, which describe the anisotropy of the light in the local coordinate frame (u, v). By designing a 2-direction field, one can specify the two directions of anisotropy by setting them to the direction of the field and its orthogonal direction respectively, thus describing the local coordinate frames up to the sign (to which the reflectance is indifferent).

Thus, using our framework for designing 2-direction field splines, provides control over the reflectance properties of such materials. Again, the real time response to newly imposed or modified hard constraints enables a direct feedback for the user, which is important for performing artistic tasks. A screenshot from an anisotropic BRDF design session using our framework is shown in Figure 14. There, we show a simple example to visualize what kind of effect we are aiming at. BRDF manipulation on a more complicated mesh can be seen in the supplementary video.

Alternatively one can directly control the shape of the highlights under a fixed viewing and light direction, as proposed in Raymond et al. [2014]. There, a relation between the BRDFs and the shape of the highlights is established, such that desired tangent directions of potential highlights can be specified by the user and then the BRDFs

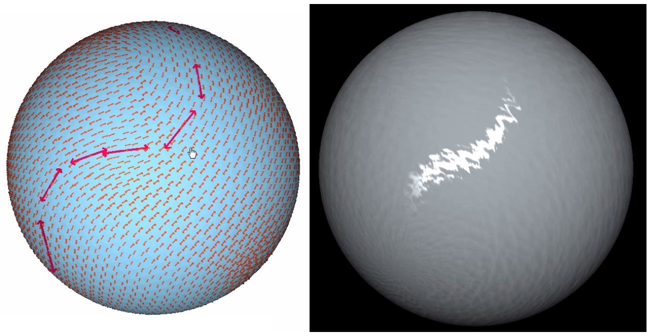

Fig. 14. Editing BRDFs to manipulate highlights on a golf-ball. The BRDF parameters are consistently and smoothly defined over the whole mesh as they are extracted from the 2-field spline, and so, highlights will be consistent and smooth even when view and light directions change.

are optimized to fit these highlights. In [Raymond et al. 2014] several tools to design a field with these prescribed tangent directions are offered. They are based on direct manipulation of a potential highlight, thus do not take into account the geometrical structure of such 2-direction fields and neglect the global structure of the field. Our tools offer an intuitive alternative that allows for the design of globally smooth tangent directions of potential highlights.

11 CONCLUSION

In this paper, we introduce n-field splines: an approach for modeling tangential n-vector and n-direction fields on surfaces. The approach enables modeling of n-fields with hard interpolation and soft alignment constraints, placing singularities and local editing. New fairness energies for n-fields, a biharmonic energy, forms the basis of our approach. The energies are convex and quadratic so that n-vector as well as n-direction field splines can be computed by solving sparse linear systems. Based on a spectral decomposition of n-vector fields, we derive a reduced optimization scheme for computing n-field splines in real-time. We apply our approach to controlling and editing of stroke directions in line-art renderings and the modeling of anisotropic BRDFs on surfaces.

Limitations and challenges. The discrete harmonic energy we introduce has one conceptual limitation, which it shares with the discrete harmonic energies for n-vector fields introduced in previous work. Only the trivial field is in its kernel. For surfaces of non-trivial genus g, there should be a $2g$-dimensional kernel of "discrete harmonic" n-vector fields. For piecewise constant vector fields (1-fields) such structure-preserving discretizations are known, however, it remains a challenge to find such a structure-preserving discretizations for n-vector fields.

A limitation of our current implementation of the n-vector field modeling tool is that the local editing is not integrated with the global editing. We use the local editing as a last step in the modeling pipeline and cannot switch back to global editing after local edits have been performed (without constraining the whole local region). Various ways to address this problem are possible. However, it is not clear which one is the simplest and most effective.

Another interesting direction of future work is to use the n-vector field splines for quadrilateral or hexagonal meshing.

ACKNOWLEDGMENTS

We would like to thank Wenzel Jakob, Marco Tarini, Daniele Panozzo, and Olga Sorkine-Hornung for sharing the source code of their Instant Field-Aligned Meshes approach.

REFERENCES

Michael Ashikhmin and Peter Shirley. 2000. An anisotropic phong BRDF model. *Journal of Graphics Tools* 5, 2 (2000), 25–32.

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

David Bommes, Henrik Zimmer, and Leif Kobbelt. 2009. Mixed-integer Quadrangulation. *ACM Trans. Graph.* 28, 3 (2009), 77:1–77:10.

Mario Botsch and Leif Kobbelt. 2004. An Intuitive Framework for Real-time Freeform Modeling. *ACM Trans. Graph.* 23, 3 (2004), 630–634.

Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, and Bruno Levy. 2010. *Polygon Mesh Processing.* AK Peters.

Dietrich Braess. 2007. *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics.* Springer.

Christopher Brandt, Leonardo Scandolo, Elmar Eisemann, and Klaus Hildebrandt. 2017. Spectral Processing of Tangential Vector Fields. *Comput. Graph. Forum* 36, 6 (2017), 338–353.

Ming-Te Chi, Chih-Yuan Yao, Eugene Zhang, and Tong-Yee Lee. 2014. Optical illusion shape texturing using repeated asymmetric patterns. *The Visual Computer* 30, 6-8 (2014).

David Cohen-Steiner and Jean-Marie Morvan. 2003. Restricted Delaunay triangulations and normal cycles. *ACM Sympos. Comput. Geom.* (2003), 237–246.

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

Fernando de Goes, Mathieu Desbrun, and Yiying Tong. 2015. Vector Field Processing on Triangle Meshes. In *SIGGRAPH Asia 2015 Courses.* 17:1–17:48.

Tyler De Witt, Christian Lessig, and Eugene Fiume. 2012. Fluid Simulation Using Laplacian Eigenfunctions. *ACM Trans. Graph.* 31, 1 (2012), 10:1–10:11.

Mathieu Desbrun, Anil N. Hirani, Melvin Leok, and Jerrold E. Marsden. 2005. Discrete Exterior Calculus. (2005). preprint, arXiv:math.DG/0508341.

Mathieu Desbrun, Eva Kanso, and Yiying Tong. 2008. *Discrete Differential Forms for Computational Modeling.* Birkhäuser, 287–324.

Olga Diamanti, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. 2014. Designing N-PolyVector Fields with Complex Polynomials. *Comput. Graph. Forum* 33, 5 (2014), 1–11.

Olga Diamanti, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. 2015. Integrable PolyVector Fields. *ACM Trans. Graph.* 34, 4 (2015), 38:1–38:12.

Hans-Christian Ebke, Patrick Schmidt, Marcel Campen, and Leif Kobbelt. 2016. Interactively Controlled Quad Remeshing of High Resolution 3D Models. *ACM Trans. Graph.* 35, 6 (2016).

Robert Eymard, Thierry Gallouët, and Raphaèle Herbin. 2000. Finite volume methods. In *Techniques of Scientific Computing, Part III,*, P. G. Ciarlet and J.-L. Lions (Eds.). North-Holland, 713–1020.

Matthew Fisher, Peter Schröder, Mathieu Desbrun, and Hugues Hoppe. 2007. Design of Tangent Vector Fields. *ACM Trans. Graph.* 26, 3 (2007), 56:1–56:10.

Aaron Hertzmann and Denis Zorin. 2000. Illustrating smooth surfaces. In *Proc. ACM SIGGRAPH.*

Klaus Hildebrandt and Konrad Polthier. 2007. Constraint-based fairing of surface meshes. In *Eurographics/ACM SIGGRAPH Symposium on Geometry Processing.* 203–212.

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

Zhiyang Huang and Tao Ju. 2016. Extrinsically Smooth Direction Fields. *Computers & Graphics* 58 (2016), 109–117.

Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. 2011. Bounded Biharmonic Weights for Real-Time Deformation. *ACM Trans. Graph.* 30, 4 (2011), 78:1–78:8.

Alec Jacobson, Elif Tosun, Olga Sorkine, and Denis Zorin. 2010. Mixed Finite Elements for Variational Surface Modeling. *Comput. Graph. Forum* 29, 5 (2010), 1565–1574.

Wenzel Jakob, Marco Tarini, Daniele Panozzo, and Olga Sorkine-Hornung. 2015. Instant Field-Aligned Meshes. *ACM Trans. Graph.* 34, 4 (2015), 189:1–189:15.

Felix Kälberer, Matthias Nieser, and Konrad Polthier. 2007. QuadCover - Surface Parameterization using Branched Coverings. *Comput. Graph. Forum* 26, 3 (2007).

Evangelos Kalogerakis, Derek Nowrouzezahrai, Simon Breslav, and Aaron Hertzmann. 2012. Learning hatching for pen-and-ink illustration of surfaces. *ACM Trans. Graph.* 31, 1 (2012), 1.

Yongjin Kim, Jingyi Yu, Xuan Yu, and Seungyong Lee. 2008. Line-art Illustration of Dynamic and Specular Surfaces. *ACM Trans. Graph.* 27, 5 (2008), 156:1–156:10.

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2013. Globally optimal direction fields. *ACM Trans. Graph.* 32, 4 (2013), 59:1–59:10.

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2015. Stripe Patterns on Surfaces. *ACM Trans. Graph.* 34, 4 (2015).

Leif Kobbelt, Swen Campagna, Jens Vorsatz, and Hans-Peter Seidel. 1998. Interactive Multi-resolution Modeling on Arbitrary Meshes. In *Proc. of ACM SIGGRAPH.* 105–114.

Yu-Kun Lai, Miao Jin, Xuexiang Xie, Ying He, Jonathan Palacios, Eugene Zhang, Shi-Min Hu, and Xianfeng Gu. 2010. Metric-Driven RoSy Field Design and Remeshing. *IEEE Trans. Vis. Comput. Graph.* 16, 1 (2010).

Bruno Lévy and Hao Zhang. 2009. Spectral mesh processing. In *ACM SIGGRAPH ASIA Courses.* 1–47.

Er Li, Bruno Lévy, Xiaopeng Zhang, WuJun Che, Weiming Dong, and Jean-Claude Paul. 2011. Meshless quadrangulation by global parameterization. *Computers & Graphics* (2011).

Beibei Liu, Gemma Mason, Julian Hodgson, Yiying Tong, and Mathieu Desbrun. 2015. Model-reduced Variational Fluid Simulation. *ACM Trans. Graph.* 34, 6 (2015), 244:1–244:12.

Beibei Liu, Yiying Tong, Fernando De Goes, and Mathieu Desbrun. 2016. Discrete Connection and Covariant Derivative for Vector Field Analysis and Design. *ACM Trans. Graph.* 35, 3 (2016), 23:1–23:17.

Maike Löhndorf and Jens Markus Melenk. 2017. On Thin Plate Spline Interpolation. In *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2016*, Marco L. Bittencourt, Ney A. Dumont, and Jan S. Hesthaven (Eds.). Springer, 451–466.

Janick Martinez Esturo, Christian Rössl, and Holger Theisel. 2014. Smoothed Quadratic Energies on Meshes. *ACM Trans. Graph.* 34, 1 (2014), 2.

Soham Uday Mehta, Ravi Ramamoorthi, Mark Meyer, and Christophe Hery. 2012. Analytic Tangent Irradiance Environment Maps for Anisotropic Surfaces. *Comput. Graph. Forum* 31, 4 (2012).

Fred Edwin Nicodemus. 1977. *Geometrical considerations and nomenclature for reflectance.* Vol. 160. US Department of Commerce, National Bureau of Standards, USA.

Matthias Nieser, Jonathan Palacios, Konrad Polthier, and Eugene Zhang. 2012. Hexagonal Global Parameterization of Arbitrary Surfaces. *IEEE Trans. Vis. and Comp. Graph.* 18, 6 (2012), 865–878.

Jonathan Palacios and Eugene Zhang. 2007. Rotational Symmetry Field Design on Surfaces. *ACM Trans. Graph.* 26, 3 (2007).

Konrad Polthier and Eike Preuß. 2000. Variational Approach to Vector Field Decomposition. In *Symposium on Data Visualization.* Springer, 147–155.

Konrad Polthier and Eike Preuß. 2003. Identifying Vector Field Singularities Using a Discrete Hodge Decomposition. In *Visualization and Mathematics III.* Springer, 113–134.

Emil Praun, Adam Finkelstein, and Hugues Hoppe. 2000. Lapped textures. In *Proc. ACM SIGGRAPH.* 465–470.

Emil Praun, Hugues Hoppe, Matthew Webb, and Adam Finkelstein. 2001. Real-time Hatching. In *Proc. ACM SIGGRAPH.*

Nicolas Ray, Wan Chiu Li, Bruno Lévy, Alla Sheffer, and Pierre Alliez. 2006. Periodic Global Parameterization. *ACM Trans. Graph.* 25, 4 (2006), 1460–1485.

Nicolas Ray, Bruno Vallet, Laurent Alonso, and Bruno Lévy. 2009. Geometry-aware Direction Field Processing. *ACM Trans. Graph.* 29, 1 (2009).

Nicolas Ray, Bruno Vallet, Wan Chiu Li, and Bruno Lévy. 2008. N-symmetry Direction Field Design. *ACM Trans. Graph.* 27, 2 (2008).

Boris Raymond, Gaël Guennebaud, Pascal Barla, Romain Pacanowski, and Xavier Granier. 2014. Optimizing BRDF Orientations for the Manipulation of Anisotropic Highlights. *Comput. Graph. Forum* 33, 2 (2014), 313–321.

Justin Solomon, Mirela Ben-Chen, Adrian Butscher, and Leonidas Guibas. 2011. Discovery of intrinsic primitives on triangle meshes. *Comput. Graph. Forum* 30, 2, 365–374.

Olga Sorkine and Daniel Cohen-Or. 2004. Least-squares Meshes. In *Proceedings of Shape Modeling International.* IEEE Computer Society Press, 191–199.

Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and Hans-Peter Seidel. 2004. Laplacian Surface Editing. In *Eurographics/ACM SIGGRAPH Symposium on Geometry Processing.* 179–188.

Jordane Suarez, Farès Belhadj, and Vincent Boyer. 2016. Real-time 3D rendering with hatching. *The Visual Computer* (2016), 1–16.

Marco Tarini, Enrico Puppo, Daniele Panozzo, Nico Pietroni, and Paolo Cignoni. 2011. Simple Quad Domains for Field Aligned Mesh Parametrization. *Proc. SIGGRAPH Asia 2011* 30, 6 (2011).

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

Greg Turk. 2001. Texture synthesis on surfaces. In *Proc. ACM SIGGRAPH.* 347–354.

Tamás Umenhoffer, László Szécsi, and László Szirmay-Kalos. 2011. Hatching for Motion Picture Production. *Comput. Graph. Forum* 30, 2 (2011), 533–542.

Amir Vaxman, Marcel Campen, Olga Diamanti, Daniele Panozzo, David Bommes, Klaus Hildebrandt, and Mirela Ben-Chen. 2016. Directional Field Synthesis, Design, and Processing. *Comput. Graph. Forum* 35, 2 (2016), 545–572.

Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. 2015. Linear Subspace Design for Real-time Shape Deformation. *ACM Trans. Graph.* 34, 4 (2015), 57:1–57:11.

Max Wardetzky. 2006. *Discrete Differential Operators on Polyhedral Surfaces–Convergence and Approximation.* Ph.D. Dissertation. Freie Universität Berlin.

Li-Yi Wei and Marc Levoy. 2001. Texture Synthesis over Arbitrary Manifold Surfaces. In *Proc ACM SIGGRAPH.*

Chih-Yuan Yao, Ming-Te Chi, Tong-Yee Lee, and Tao Ju. 2012. Region-Based Line Field Design Using Harmonic Functions. *IEEE Trans. Vis. Comp. Graph.* 18, 6 (2012).

Eugene Zhang, James Hays, and Greg Turk. 2007. Interactive Tensor Field Design and Visualization on Surfaces. *IEEE Trans. Vis. Comp. Graph.* 13, 1 (2007), 94–107.

Eugene Zhang, Konstantin Mischaikow, and Greg Turk. 2006. Vector field design on surfaces. *ACM Trans. Graph.* 25, 4 (2006), 1294–1326.

Hao Zhang, Oliver van Kaick, and Ramsay Dyer. 2010. Spectral Mesh Processing. *Comput. Graph. Forum* 29, 6 (2010), 1865–1894.

Yixin Zhuang, Ming Zou, Nathan Carr, and Tao Ju. 2014. Anisotropic geodesics for live-wire mesh segmentation. *Comput. Graph. Forum* 33, 7 (2014).

A DERIVATION OF THE DISCRETE HARMONIC ENERGY

The discrete harmonic energy for n-vector fields is defined as

$$E_H(\mathbf{u}) = \sum_{(i,j)\in\mathcal{E}} w_{ij} \left\| R_{ij} u_i - u_j \right\|^2 \tag{19}$$

$$\text{where} \;\; w_{ij} = \frac{3 l_{e_{ij}}^2}{\operatorname{area}(T_i \cup T_j)}.$$

In the following, we will justify the weights w_{ij}, as they emerge naturally in a face-based discretization of the harmonic energy for vector fields. The harmonic energy of a smooth vector field $\mathbf{v}$ is given by $E_H^{\text{smooth}}(\mathbf{v}) = \langle \mathbf{\Delta v}, \mathbf{v} \rangle_{\mathbf{L}^2}$, where $\mathbf{\Delta}$ is the Hodge–Laplace operator

$$\mathbf{\Delta} = -(\text{grad div} + \mathbf{J} \text{ grad curl.})$$

Rearranging terms, we get

$$\begin{aligned} E_H^{\text{smooth}}(\mathbf{v}) &= \langle \mathbf{\Delta v}, \mathbf{v} \rangle_{\mathbf{L}^2} \\ &= \langle -(\text{grad div} + \mathbf{J} \text{ grad curl})\mathbf{v}, \mathbf{v} \rangle_{\mathbf{L}^2} \\ &= -\langle \text{grad div } \mathbf{v}, \mathbf{v} \rangle_{\mathbf{L}^2} - \langle \mathbf{J} \text{ grad curl } \mathbf{v}, \mathbf{v} \rangle_{\mathbf{L}^2} \\ &= \langle \text{div } \mathbf{v}, \text{div } \mathbf{v} \rangle_{\mathbf{L}^2} + \langle \text{curl } \mathbf{v}, \text{curl } \mathbf{v} \rangle_{\mathbf{L}^2} \end{aligned}$$

where we used the equations $\langle \text{div } \mathbf{v}, f \rangle_{\mathbf{L}^2} = -\langle \mathbf{v}, \text{grad } f \rangle_{\mathbf{L}^2}$, $\text{curl } \mathbf{v} = -\text{div } \mathbf{Jv}$ and $\langle \mathbf{Jv}, \mathbf{v} \rangle_{\mathbf{L}^2} = -\langle \mathbf{v}, \mathbf{Jv} \rangle_{\mathbf{L}^2}$, which hold for all smooth vector fields $\mathbf{v}$ and square integrable functions f. For more background, we refer to [Brandt et al. 2017]. In [Fisher et al. 2007] and [Desbrun et al. 2008], a DEC-based discretization of this energy is used for vector field design. We are dealing with piecewise constant vector fields that jump at the edges. In this setting, discrete divergence and curl operators can be defined by testing a weak form of the divergence and curl with test functions. As test functions, we are using Crouzeix–Raviart finite elements, which are functions on the mesh that are linear polynomials in every triangle and edge-midpoint continuous. The nodes of the Crouzeix–Raviart elements are located at the midpoints of the edges of the mesh. The (non-conforming discrete) divergence and curl map piecewise constant vector fields to such edge-based functions. For more background on the discrete divergence and curl, we refer to [Brandt et al. 2017; Polthier and

Preuß 2000]. The values at the edge-midpoints are given by

$$\text{div}^* \, \mathbf{v}(m_{i,j}) = \frac{3}{A_{T_i \cup T_j}} \left(\langle v_i, J_i \, e_{ij} \rangle - \langle v_j, J_j \, e_{ij} \rangle \right), \quad (20)$$

$$\text{curl}^* \, \mathbf{v}(m_{i,j}) = \frac{3}{A_{T_i \cup T_j}} \left(\langle v_j, e_{ij} \rangle - \langle v_i, e_{ij} \rangle \right), \quad (21)$$

where $\left(v_1, ..., v_{|\mathcal{F}|}\right)$ are the piecewise constant vectors of the field per face, $m_{i,j}$ is the midpoint of the common edge between two adjacent triangles T_i and T_j, J_i is the operator that rotates a vector in the tangent plane of T_i by $\pi/2$ (following the orientation of the surface), $A_{T_i \cup T_j}$ is the combined area of T_i and T_j, and e_{ij} is the **non**-normalized directed (following the orientation of T_i) common edge between triangles T_i and T_j.

With these operators, one can discretize the harmonic energy:

$$\begin{aligned} E_H(\mathbf{v}) &= \langle \text{div}^* \, \mathbf{v}, \text{div}^* \, \mathbf{v} \rangle_{\mathbf{L}^2} + \langle \text{curl}^* \, \mathbf{v}, \text{curl}^* \, \mathbf{v} \rangle_{\mathbf{L}^2} \\ &= \sum_{(i,j)\in\mathcal{E}} \frac{3}{A_{T_i \cup T_j}} \Big(\left(\langle v_i, J_i \, e_{ij} \rangle - \langle v_j, J_j \, e_{ij} \rangle \right)^2 \\ &\qquad + \left(\langle v_j, e_{ij} \rangle - \langle v_i, e_{ij} \rangle \right)^2 \Big) \quad (22) \\ &= \sum_{(i,j)\in\mathcal{E}} \frac{3 l_{e_{ij}}^2}{A_{T_i \cup T_j}} \left(\left\langle R_{ij} v_i - v_j, J_j \, \frac{e_{ij}}{l_{e_{ij}}} \right\rangle^2 \right. \\ &\qquad \left. + \left\langle v_j - R_{ij} v_i, \, \frac{e_{ij}}{l_{e_{ij}}} \right\rangle^2 \right) \quad (23) \\ &= \sum_{(i,j)\in\mathcal{E}} \frac{3 l_{e_{ij}}^2}{A_{T_i \cup T_j}} \left\| R_{ij} v_i - v_j \right\|^2 \quad (24) \end{aligned}$$

In (22), we simply inserted the definition of the operators and canceled the squared area factors in div* and curl* with the area factors coming from the sum that computes the L^2 scalar product for the Crouzeix–Raviart elements. In (23), we used that $\langle v_i, e_{ij} \rangle = \langle R_{ij}(v_i), e_{ij} \rangle$. ($R_{ij}$ is the connection for 1-fields) and the linearity of the scalar product. Finally, in (24), we used that $\langle v, w \rangle^2 + \langle v, \mathbf{J} w \rangle^2 = \|v\|^2$ for any pair of an arbitrary vector v and a unit vector w. Then, since for $n = 1$, when using the same tangent space bases for the vectors and their n-vector representations, the vectors v_i coincide with the n-vectors u_i, so that (24) coincides with (2).

Since we end up with a weighted sum of finite differences, we can extend this energy to arbitrary n by taking the differences of the u_i representing the n-vectors by transporting them into a common tangent space. This leads to the generalized harmonic energy (19).

As discussed in Section 11, the discrete harmonic energy does not have a kernel of "discrete harmonic" n-vector fields for surfaces of non-trivial genus. For vector fields such a discretization can be obtained by combining conforming and non-conforming finite elements, see [Brandt et al. 2017]. However, it remains an open question how this construction can be carried over from 1-vector fields to n-vector fields.

Finally, we want to remark that in addition to the harmonic energy and the Hodge–Laplace operator, one can consider the Bochner–Laplace operator and the corresponding quadratic energy. For more background, we refer to [Knöppel et al. 2013; Liu et al. 2016]. In the notation used in these papers, the harmonic energy is called the anti-holomophic energy. The construction of a discrete Bochner Laplace operator for piecewise constant vector fields on triangle meshes remains an interesting open problem.

Relation to Finite Volumes. We want to remark that in addition to the derivation described above, the discrete harmonic energy (19) can be interpreted as a finite volume discretization. The control volumes are the triangles and each summand $w_{ij} \left\| R_{ij} u_i - u_j \right\|^2$ of the discrete harmonic energy measures the diffusive flux through the edge (i, j) between adjacent triangles. The weights w_{ij} are the transmissibilities.

For more background on finite volume methods, we refer to [Eymard et al. 2000]. The discrete harmonic energy (19) is analogous to the discrete harmonic energy (or squared discrete Sobolev H_0^1 semi norm) for piecewise constant functions on triangulations of compact domains in $\mathbb{R}^2$, compare [Eymard et al. 2000, Chapter 9, eq. (9.12)]. In contrast to their setting, we are working with vector fields and curved surface meshes, this is why the transport of vector fields R_{ij} is needed for evaluating the fluxes. Based on this analogy, we could use the weights that are commonly used for finite volume discretization of the harmonic energy with piecewise constant functions for our purposes. These are

$$\frac{2}{\cot \alpha_{ij} + \cot b_{ij}},$$

where α_{ij} and b_{ij} are the angles opposite of the edge (i, j) in the two adjacent triangles. The problem with these weights is that they may be negative, which can be avoided by requiring the triangulation to be Delaunay. The weights we propose are positive for any triangulation by construction. To the best of our knowledge, the weights we are proposing have not been used in the context of finite volume methods. It is an interesting task to further explore the properties of these weights and their use for finite volume methods.

B A LINEAR SYSTEM FOR n-DIRECTION FIELD SPLINES

In this section, we derive the linear system we solve to compute n-vector field splines and discuss the relationship between μ and λ. Our proof re-uses arguments from [Knöppel et al. 2013], however, since we use hard constraints and different objectives and discretizations, some modifications need to be made, and we will verify correctness for our specific system.

We will show that $\mathbf{u} = \frac{\mathbf{w}}{\|\mathbf{w}\|_{\mathbf{L}^2}}$, for $\mathbf{w}$ being a solution of the linear system from Section 7

$$\begin{pmatrix} \mathbf{B} - \lambda \mathbf{M} & \mathbf{D}^T \\ \mathbf{D} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ \boldsymbol{\gamma} \end{pmatrix} = \begin{pmatrix} \mathbf{M}\mathbf{u}' \\ 0 \end{pmatrix}, \quad (25)$$

is a minimizer of the previously stated n-vector field spline problem

$$\begin{aligned} \min_{\mathbf{u}} \; & E_B(\mathbf{u}) - 2\mu \langle \mathbf{u}, \mathbf{u}' \rangle_{\mathbf{L}^2} \quad (26) \\ \text{subject to } & \|\mathbf{u}\|_{\mathbf{L}^2} = 1 \\ \text{and } & \langle u_i, J_i d_i \rangle = 0 \;\forall\; i \in I \end{aligned}$$

To this end note that (25) can be rewritten as

$$\begin{aligned} \mathbf{B}\mathbf{w} - \mu \|\mathbf{w}\|_{\mathbf{L}^2} \mathbf{M}\mathbf{u}' &= \lambda \mathbf{M}\mathbf{w} - \mathbf{D}^T \boldsymbol{\gamma} \quad (27) \\ \text{and } \mathbf{D}\mathbf{w} &= 0, \end{aligned}$$

where $\mu = \frac{1}{\|\mathbf{w}\|_{\mathrm{L}^2}}$. Multiplying (27) by $\frac{1}{\|\mathbf{w}\|_{\mathrm{L}^2}}$, letting $\tilde{\boldsymbol{\gamma}} = \frac{\boldsymbol{\gamma}}{\|\mathbf{w}\|_{\mathrm{L}^2}}$ and plugging in $\mathbf{u} = \frac{\mathbf{w}}{\|\mathbf{w}\|_{\mathrm{L}^2}}$ we get

$$\begin{aligned} \mathbf{B}\mathbf{u} - \mu\mathbf{M}\mathbf{u}' &= \lambda\mathbf{M}\mathbf{u} - \mathbf{D}^T\tilde{\boldsymbol{\gamma}} \\ \text{and } \ \|\mathbf{u}\|_{\mathrm{L}^2} &= 1 \\ \text{and } \ \langle u_i, J_i d_i \rangle &= 0 \ \forall\ i \in I \end{aligned} \tag{28}$$

since $\langle w_i, J_i d_i \rangle = 0 \Leftrightarrow \langle u_i, J_i d_i \rangle = 0$. These are the necessary conditions for the constrained optimization problem (26), where the Lagrange multipliers are 2λ and $2\tilde{\gamma}$. Since the minimized functional is quadratic in $\mathbf{u}$, this is sufficient to show that $\mathbf{u}$ is a feasible minimizer.

To determine the relationship between λ and μ, we first make a change of basis $\mathbf{w} = \mathbf{C}\hat{\mathbf{w}}$, where the image of $\mathbf{C} \in M(2|\mathcal{F}|\times 2|\mathcal{F}|-m)$ is the vector space of all fields that satisfy the hard constraints, *i.e.* $\mathbf{D}\mathbf{w} = 0 \ \Leftrightarrow \ \mathbf{w} \in$ Image $\mathbf{C}$ and $\mathbf{C}^T\mathbf{C} = \mathbf{I}$. Then (25) is equivalent to solving

$$\mathbf{C}^T\left(\mathbf{B} - \lambda\mathbf{M}\right)\mathbf{C}\hat{\mathbf{w}} = \mathbf{C}^T\mathbf{M}\mathbf{u}', \tag{29}$$

since $\mathbf{u}'$ satisfies all hard constraints by construction and so $\mathbf{u}' \in$ Image $\mathbf{C}$. Let $\hat{\mathbf{U}}$ be a full eigenbasis of the constrained bi-Laplacian $\hat{\mathbf{V}} := \mathbf{C}^T\left(\mathbf{L}^T\mathbf{M}\mathbf{L}\right)\mathbf{C}$ which is mass-orthonormal, *i.e.* $\hat{\mathbf{U}}^T\mathbf{C}^T\mathbf{M}\mathbf{C}\hat{\mathbf{U}} = \mathbf{I}$. Then (29) is equivalent to

$$\left(\mathbf{\Lambda} - \lambda\mathbf{I}\right)\hat{\mathbf{w}}_{\hat{\mathbf{U}}} = \hat{\mathbf{U}}^T\mathbf{C}^T\mathbf{M}\mathbf{u}', \tag{30}$$

where $\hat{\mathbf{w}}_{\hat{\mathbf{U}}} = \hat{\mathbf{U}}^T\hat{\mathbf{w}}$ and $\mathbf{\Lambda}$ stacks the eigenvalues $\hat{\lambda}_i$ of $\hat{\mathbf{V}}$ ordered by magnitude (note that the eigenvalues are all positive as the frequency of hard constrained fields can only be higher than those of an unconstrained field). Now note that $\left(\hat{\mathbf{w}}_{\hat{\mathbf{U}}}\right)_i = \left(\hat{\mathbf{U}}^T\mathbf{C}^T\mathbf{M}\mathbf{u}'\right)_i\left(\hat{\lambda}_i - \lambda\right)^{-1}$. So, when $\lambda \longrightarrow \hat{\lambda}_1$, $\left(\hat{\mathbf{w}}_{\hat{\mathbf{U}}}\right)_1$ goes to (plus or minus) infinity. This implies $\|\hat{\mathbf{w}}\| \longrightarrow \infty$, and so $\mu = \frac{1}{\|\mathbf{w}\|} \longrightarrow 0$. Conversely, when $\lambda \longrightarrow -\infty$ we have that $\left(\hat{\mathbf{w}}_{\hat{\mathbf{U}}}\right)_i \longrightarrow 0$ for all i and so $\mu \longrightarrow \infty$.

Received October 2016; revised November 2017; accepted December 2017