Frame Fields: Anisotropic and Non-Orthogonal Cross Fields

Daniele Panozzo[1] Enrico Puppo[2] Marco Tarini[3,4] Olga Sorkine-Hornung[1]

[1]ETH Zurich [2] Università di Genova [3] Università dell'Insubria, Varese [4] ISTI-CNR Pisa

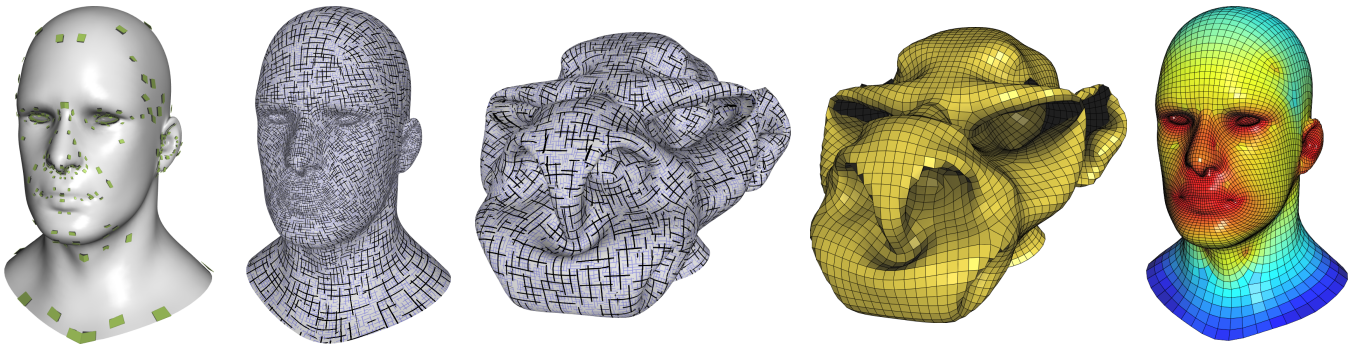


Figure 1: *Given a sparse set of constraints defined on the input surface, we interpolate a dense, non-uniform, anisotropic and non-orthogonal frame field. We then deform the surface to warp this frame field into a cross field, which we use to guide a uniform, isotropic and orthogonal quadrangulation of the deformed surface. Finally, we deform the resulting quad mesh back onto the original surface and obtain a non-uniform, anisotropic and non-orthogonal quadrangulation that follows the prescribed frame field (color scale represents element areas).*

Abstract

We introduce frame fields, which are a non-orthogonal and non-unit-length generalization of cross fields. Frame fields represent smoothly varying linear transformations on tangent spaces of a surface. We propose an algorithm to create discrete, dense frame fields that satisfy a sparse set of constraints. By computing a surface deformation that warps a frame field into a cross field, we generalize existing quadrangulation algorithms to generate anisotropic and non-uniform quad meshes whose elements shapes match the frame field. With this, our framework enables users to control not only the alignment but also the density and anisotropy of the elements' distribution, resulting in high-quality adaptive quad meshing.

CR Categories: I.3.5 [Computer Graphics]: Computational geometry and object modeling—Curve, surface, solid and object repres.

Keywords: frame field, n-RoSy field, anisotropic quad mesh

Links: DL PDF WEB VIDEO DATA

1 Introduction

Cross fields assign to every point of a surface a smoothly varying pair of orthogonal directions on the tangent plane. They are extensively used in Computer Graphics for generating quadrilateral meshes [Bommes et al. 2013a], for non-photorealistic rendering [Hertzmann and Zorin 2000; Palacios and Zhang 2007], texture synthesis [Lefebvre and Hoppe 2006; Li et al. 2011], and in architectural geometry [Liu et al. 2011; Panozzo et al. 2013].

The topology of a cross field is determined by singular points and separatrix lines connecting them: the singularities divert the flow of tangential directions, and the separatrices divide the surface into uniform patches. The arrangement of these topological features can be used to design cross fields that follow certain surface characteristics, such as curvature extrema and principal curvature lines, as well as to vary the field's density [Bommes et al. 2013a].

We introduce *frame fields*, which generalize cross fields by incorporating anisotropy, scaling and skewness while maintaining the topological structure of a cross field. A frame field is defined at each point of a surface by a pair of – possibly non-orthogonal and non-unit-length – tangent vectors, and their opposite vectors. It can be seen as a smoothly varying linear transformation on the tangent bundle of a smooth surface. Several natural quantities, such as the curvature tensor, the stress tensor, or a parameterization Jacobian, can be encoded using frame fields. We show that by decoupling the rotational component from the scaling and shearing, a frame field can be uniquely decomposed into a smooth cross field and a smooth symmetric tensor field on the tangent bundle.

We propose an interpolation algorithm that generates a dense frame field from a sparse set of constraints. Depending on the application, the constraints can be manually designed or automatically extracted from the surface geometry.

A frame field indicates the overall structure of an anisotropic and scale-varying quad meshing of the surface. In traditional parametrization-based quadrangulators [Bommes et al. 2013a], a given cross field describes the desired local alignment of mesh elements, whose ideal shape is assumed to be square. Frame fields remove this assumption: they additionally describe the *length* of mesh edges and their *skewness*, thus fully specifying the desired local shape of mesh elements.

We show how to warp a frame field into a cross field with a variational approach, by deforming the input surface in $\mathbb{R}^3$. The resulting surface is isotropically quadrangulated; by warping it back to the

original shape, we generate a non-homogeneous, anisotropic and skewed quad mesh, similar to the kind of meshes manually designed by professionals for animation or CAD purposes. The singularities, introduced during the isotropic quadrangulation to accommodate for the intrinsic curvature of the deformed shape, handle the change of tessellation density in the adaptive quad mesh (see Figure 1). In addition to quadrilateral meshing, frame fields can also be used to guide deformations of images and surfaces.

The contributions of this paper can be summarized as follows:

1. We define frame fields, highlighting their relations to cross fields and providing the mathematical foundations for their canonical representation over triangle meshes.

2. We present an algorithm to generate a frame field that interpolates a set of given constraints.

3. We apply frame fields to extend a popular class of isotropic mesh quadrangulation algorithms to generate meshes with anisotropic and non-uniform quad faces.

4. We present a cross-hatching technique for visualizing a frame field.

2 Related work

We review the literature on cross fields, which are orthogonal and unit-length frame fields, and we discuss existing anisotropic meshing approaches. We also briefly touch on the relation between interpolating linear transformations and frame fields.

Cross fields were initially proposed by Hertzmann and Zorin [2000] for non-photorealistic rendering by means of cross-hatching. Following the seminal work on vector field design [Zhang et al. 2006; Fisher et al. 2007], cross fields have been formally defined and studied a few years later, and also generalized to a wider set of symmetries [Palacios and Zhang 2007; Ray et al. 2008]. A cross field can be created by specifying its singularities [Ray et al. 2008; Crane et al. 2010], or by prescribing a sparse set of directional constraints and letting the topology emerge from a smoothing process [Bommes et al. 2009; Ray et al. 2009; Knöppel et al. 2013]. Cross fields do not encode scale or anisotropy, which restricts their application to tasks such as hatching-based NPR [Hertzmann and Zorin 2000; Palacios and Zhang 2007] or uniform, isotropic remeshing [Ray et al. 2006; Kälberer et al. 2007; Bommes et al. 2009; Lai et al. 2010; Pietroni et al. 2011]. Cross fields have been recently used to design thrust networks and self-supporting surfaces [Panozzo et al. 2013]. Conjugate fields [Liu et al. 2011] are non-orthogonal cross fields which can be used to create planar quad meshes.

Anisotropic quadrilateral meshing. In the context of remeshing, anisotropy allows increasing the approximation power of a discretization without increasing the element count [Alliez et al. 2003; Kovacs et al. 2010]. Anisotropic, quad-dominant meshes are also commonly used to define control grids for subdivision surfaces that align with shape features, in order to reduce the distortion of the limit surface during animation [DeRose et al. 1998].

Anisotropic, polygonal meshes can be generated by tracing curvature lines from a set of seed points whose density is proportional to the curvature values [Alliez et al. 2003; Marinov and Kobbelt 2004]. The resulting meshes are quad-dominant but often contain other polygonal elements as well, which makes them challenging to use for FEM or subdivision surfaces. Element alignment to principal curvature directions is not necessarily desired in applications where the user requires explicit control over the alignment, e.g. for animation purposes [Takayama et al. 2013].

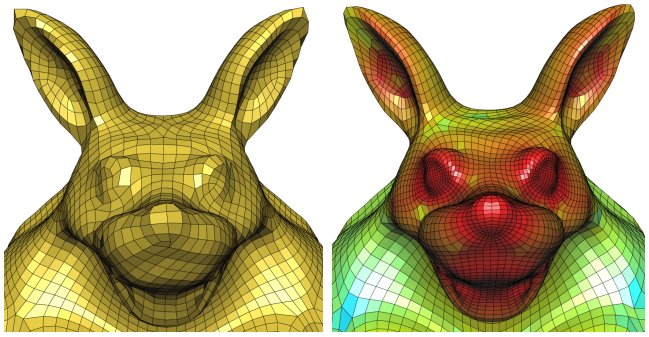


Figure 2: *Uniform quadrangulation (MIQ) vs. our anisotropic quadrangulation. A similar number of quads is used in both cases to cover the whole* Armadillo *(see also Figure 10 for a full view of our result). The same constraints are given as input in both cases; the nearest cross field to our frame field is used for the MIQ. The Haursdorff error w.r.t. the input surface is 20% lower with our variable-scale meshing: 1.5e-4 compared to 1.8e-4 for MIQ, measured w.r.t. the bounding box diagonal.*

A different approach to create quad meshes with variable density is proposed in [Huang et al. 2008; Zhang et al. 2010; Ling et al. 2011]. The quadrangulation is defined as the Morse-Smale complex of the solution of a wave equation, modeled with an anisotropic surface metric. Unfortunately, this type of methods provides only limited control over the placement of singularities and edge alignment, which are both highly desirable practical requirements.

Liu et al. [2011] generalize [Bommes et al. 2009] to conjugate direction fields, with the purpose of designing architectural structures made of flat quadrangular elements. The method specifically targets the architectural application and is computationally expensive.

Anisotropy of elements can also emerge from techniques aimed at producing very coarse quad layouts [Tarini et al. 2011; Bommes et al. 2013b]. In this case, however, the anisotropy is a casual byproduct rather than a controlled result stemming from input requirements.

Kovacs et al. [2010] propose an adaptation of [Bommes et al. 2009] to generate curvature-aligned, anisotropic quad meshes. Their key idea is to embed the surface in a 6D space and use the Riemannian metric induced by the embedding to drive the parametrization and quad meshing. The method is restricted to anisotropy implied by curvature directions, and it is unable to insert singularities that are necessary to change the scale of the quads. A similar metric is also used in [Lévy and Bonneel 2012; Zhong et al. 2013] for anisotropic triangular remeshing.

As remarked in [Kovacs et al. 2010], higher-dimensional embeddings are effective to manage anisotropy, but they mask the relation between Gaussian curvature and field topology, which makes it difficult to place singularities based on the modified metric. In contrast, we rely on a 3D deformation that warps the anisotropic metric induced by our frame fields into a Euclidean metric. This way, both curvature-related and density-related irregular vertices of the final tessellation emerge from curvature-related singularities of the cross field on the deformed surface. An example result obtained with our method is compared in Figure 2 with a uniform quad mesh obtained with the popular Mixed-Integer Quadrangulation (MIQ) of Bommes et al. [2009]. The color scale depicts the area of mesh elements (red/smaller to blue/larger), highlighting the adaptivity of our result.

Interpolation of 2D transformations. Images and 2D shapes can be manipulated by interpolating a sparse set of linear transformations, see e.g. [Schaefer et al. 2006; Jacobson et al. 2011; Yücer et al.

2012]. Frame fields provide a natural way to interpolate linear transformations on the tangent bundle of a 2-manifold: they can interpolate 2D shear, scaling and rotations efficiently, as explained in Section 5. Several previous works discuss interpolation of linear and affine transformations in Euclidean spaces for deformation and animation purposes (see, e.g., [Alexa et al. 2000; Rossignac and Vinacua 2011]), while our method operates on surfaces.

3 Frame fields

Let $\mathcal{S}$ be a smooth, oriented surface embedded in $\mathbb{R}^3$, let p be a point on $\mathcal{S}$ and $\mathbf{T}_p\mathcal{S}$ the tangent plane at p.

A *frame* f_p at a point p of $\mathcal{S}$ is defined as a cyclically ordered set of four vectors $\langle \mathbf{v}, \mathbf{w}, -\mathbf{v}, -\mathbf{w} \rangle$ on $\mathbf{T}_p\mathcal{S}$ (hereafter, angular brackets represent cyclic order), such that $\mathbf{v}$ and $\mathbf{w}$ are linearly independent, and the angle of the counterclockwise rotation of $\mathbf{v}$ onto $\mathbf{w}$ is smaller than π.

Note that every pair of consecutive vectors in this cyclic order, i.e., $(\mathbf{v}, \mathbf{w}), (\mathbf{w}, -\mathbf{v}), (-\mathbf{v}, -\mathbf{w}), (-\mathbf{w}, \mathbf{v})$, together with the origin p define a right-handed basis (generally non-orthonormal and even non-orthogonal) for $\mathbf{T}_p\mathcal{S}$.

If $\mathbf{v}, \mathbf{w}$ are an orthonormal pair, i.e., $\mathbf{v} = \mathbf{u}$ and $\mathbf{w} = \mathbf{u}^\perp$, with $|\mathbf{u}| = 1$, then the resulting frame $\langle \mathbf{u}, \mathbf{u}^\perp, -\mathbf{u}, -\mathbf{u}^\perp \rangle$ is called a *cross*.

A *frame field* $\mathcal{F}$ associates to each point p of $\mathcal{S}$ a frame in the tangent plane $\mathbf{T}_p\mathcal{S}$. A *cross field* is a frame field where all frames are crosses; this is consistent with the definition in [Ray et al. 2008].

Given a frame f_p at p, let us put an arbitrary right-handed orthonormal basis $\mathbf{B}$ on $\mathbf{T}_p\mathcal{S}$ and let us express the vectors of frame f_p with respect to $\mathbf{B}$. We denote by $\mathbf{V} = [\mathbf{v}, \mathbf{w}]$ the 2×2 matrix having as columns the coordinates of $\mathbf{v}$ and $\mathbf{w}$ w.r.t. the basis $\mathbf{B}$. The determinant of $\mathbf{V}$ is always positive, since $\mathbf{v}$ and $\mathbf{w}$ are linearly independent and form an angle between 0 and π. If f_p is a cross, then $[\mathbf{u}, \mathbf{u}^\perp]$ is a rotation matrix. Conversely, every 2×2 matrix with a positive determinant defines a frame, and every 2×2 rotation matrix defines a cross.

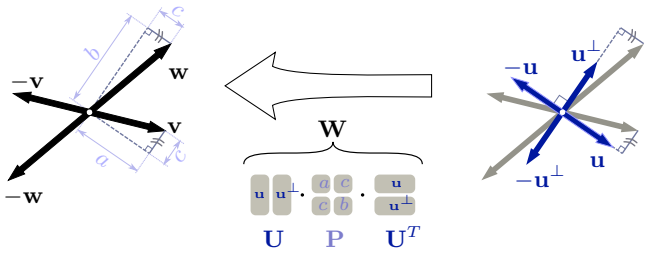


Figure 3: *Canonical decomposition of a frame (left) into a tensor* $\mathbf{W}$ *(middle) and a cross (right).* $\mathbf{W}$ *transforms the cross into the frame (right-to-left).*

Canonical decomposition. We can unambiguously represent a frame $f_p = \langle \mathbf{v}, \mathbf{w}, -\mathbf{v}, -\mathbf{w} \rangle$ by a combination of a cross $\mathbf{x} = \langle \mathbf{u}, \mathbf{u}^\perp, -\mathbf{u}, -\mathbf{u}^\perp \rangle$ and a linear, symmetric positive definite (SPD) map $\mathbf{W}$ on the tangent plane $\mathbf{T}_p\mathcal{S}$, such that

$$f_p = \mathbf{W}\mathbf{x} = \langle \mathbf{W}\mathbf{u}, \mathbf{W}\mathbf{u}^\perp, -\mathbf{W}\mathbf{u}, -\mathbf{W}\mathbf{u}^\perp \rangle. \tag{1}$$

To show this, let us set a local right-handed orthonormal basis on $\mathbf{T}_p\mathcal{S}$ and define $\mathbf{V} = [\mathbf{v}, \mathbf{w}]$ as before. Since $\mathbf{V}$ has full rank, it admits a unique polar decomposition $\mathbf{V} = \mathbf{U}\mathbf{P}$ where $\mathbf{U}$ is an orthogonal matrix and $\mathbf{P}$ is an SPD matrix. Moreover, since $\det \mathbf{V} > 0$, $\mathbf{U}$ is a rotation (in fact, it is the rotation nearest to the linear map $\mathbf{V}$) and $\mathbf{P}$ contains the stretch factors intrinsic to $\mathbf{V}$. We rewrite the polar decomposition as $\mathbf{V} = \mathbf{W}\mathbf{U}$, where

$$\mathbf{W} = \mathbf{U}\mathbf{P}\mathbf{U}^T. \tag{2}$$

Note that $\mathbf{W}$ is SPD (see Figure 3). Let $\mathbf{u}$ be the unit-length vector corresponding to the first column of $\mathbf{U}$, i.e., $\mathbf{U} = [\mathbf{u}, \mathbf{u}^\perp]$, then:

$$\mathbf{W}\,[\mathbf{u}, \mathbf{u}^\perp, -\mathbf{u}, -\mathbf{u}^\perp] = [\mathbf{v}, \mathbf{w}, -\mathbf{v}, -\mathbf{w}]. \tag{3}$$

Any cyclic permutation of the four columns of the matrix built from $\mathbf{u}$ represents the same cross field, and its transformation through $\mathbf{W}$ returns a corresponding cyclic permutation of the matrix built from $\mathbf{v}, \mathbf{w}$. Thus we can safely declare that $f_p = \mathbf{W}\mathbf{x}$:

Lemma 3.1. *Let* $f_p = \langle \mathbf{v}, \mathbf{w}, -\mathbf{v}, -\mathbf{w} \rangle$ *be a frame on* $\mathbf{T}_p\mathcal{S}$*. There exists a unique cross* $\mathbf{x} = \langle \mathbf{u}, \mathbf{u}^\perp, -\mathbf{u}, -\mathbf{u}^\perp \rangle$ *and a unique SPD linear map* $\mathbf{W}$ *such that* $f_p = \mathbf{W}\mathbf{x} = \langle \mathbf{W}\mathbf{u}, \mathbf{W}\mathbf{u}^\perp, -\mathbf{W}\mathbf{u}, -\mathbf{W}\mathbf{u}^\perp \rangle$.

In summary, a frame field $\mathcal{F}$ on $\mathcal{S}$ can be uniquely decomposed into a cross field $\mathcal{X}$ and an SPD tensor field $\mathcal{W}$. The decomposition is defined point-wise, therefore it cannot be directly used to characterise the differential properties of the frame field. In the following, we rely on the differential properties of the components to define the corresponding properties of the frame field.

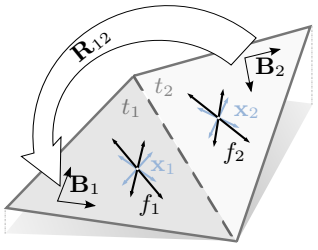
Definition 3.2. A frame field is said to be continuous/smooth if both $\mathcal{X}$ and $\mathcal{W}$ are continuous/smooth.

Note that while this can potentially restrict the space of continuous/smooth frame fields, we did not experience this problem in our experiments. As we will discuss in Section 5, we always generate a smooth frame field by optimizing for a smooth cross field and a smooth SPD tensor field.

Discrete frame fields. Let $\mathcal{M}$ be a triangle mesh approximating a smooth surface $\mathcal{S}$. Similarly to [Ray et al. 2008], we discretize the frame field $\mathcal{F}$ on $\mathcal{S}$ as a piecewise-constant field on $\mathcal{M}$, constant inside each triangle.

A frame f_t on triangle t is represented by the two canonical components (Lemma 3.1): a cross $\mathbf{x}$ and an SPD linear map $\mathbf{W}$. The cross $\mathbf{x}$ is represented by an angle w.r.t. a local orthonormal basis $\mathbf{B}$ on the plane of t, modulo rotations of $\pi/2$ [Ray et al. 2008]. The map $\mathbf{W}$ is represented as an SPD 2×2 matrix w.r.t. the same basis $\mathbf{B}$.

An advantage of this representation over a more explicit one, which directly encodes two of the four vectors of the frame, is that $\mathbf{W}$ is independent of the choice of the representative vectors. However, the representation is not globally consistent, since the basis $\mathbf{B}$ is different for every triangle. Note that, apart from special cases, it is impossible to choose a globally consistent (smooth) basis for the entire mesh due to the hairy ball theorem.



A discrete cross field is said to be *smooth* if it minimizes the smoothness energy proposed in [Ray et al. 2008] (see their Eq. 16), i.e., if the differences between adjacent crosses are small. To compare between *frames* on two adjacent triangles, we need to locally express the frame field in a common reference system. Let t_1 and t_2 be two triangles sharing an edge, and let $f_1 = \{\mathbf{x}_1, \mathbf{W}_1\}$, $f_2 = \{\mathbf{x}_2, \mathbf{W}_2\}$ be the two corresponding frames expressed in the bases of the triangles' planes, $\mathbf{B}_1$ and $\mathbf{B}_2$. The change of basis for the cross part of the frame field is the

rotation $\mathbf{R}_{12} = \mathbf{B}_1\mathbf{B}_2^{-1}$. The same matrix $\mathbf{R}_{12}$ is also used to transport the tensor part: let us decompose $\mathbf{W}_1 = \mathbf{U}_1\mathbf{P}_1\mathbf{U}_1^T$ and $\mathbf{W}_2 = \mathbf{U}_2\mathbf{P}_2\mathbf{U}_2^T$, where the $\mathbf{U}$'s and $\mathbf{P}$'s are the rotation and SPD matrices, respectively (as in the canonical decomposition). To change the reference system of $\mathbf{W}_2$ to be compatible with $\mathbf{W}_1$ we have to substitute $\mathbf{U}_2$ with $\mathbf{R}_{12}\mathbf{U}_2$. Consequently, $\mathbf{W}_2$ expressed in the same reference system of $\mathbf{W}_1$ becomes $\mathbf{R}_{12}\mathbf{W}_2\mathbf{R}_{12}^T$.

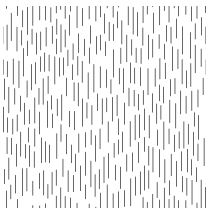
Alternatively, the same construction could be adapted to discretize frame fields on vertices and linearly interpolate them inside faces. In this case, the cross field would be defined using [Knöppel et al. 2013], and the linear part interpolated using barycentric coordinates. We opted for the piecewise-constant definition since it fits better with the following algorithmic part, where we generate a map whose Jacobian matches a piecewise-constant matrix defined per triangle.

4 Frame field visualization

A visualization of a given frame field on a mesh can be desirable in a variety of applications. The corresponding task for cross fields is elegantly solved in [Palacios and Zhang 2011] by using line integral convolution (LIC). A frame field has additional features to be conveyed, which makes its visualization more difficult: while *skewness* can be represented naturally with LIC, *scale* and *anisotropy* cannot. As noted in [Palacios and Zhang 2011], these attributes might be mapped over color, but the mapping would be arbitrary and unintuitive.

We rather resort to a cross-hatching technique: we superimpose two line-hatchings in the two directions specified by the frame field; the *spacing* between the lines in one direction indicates the length of the frame field component in the other direction. This cross-hatching tends to sketch rectangles with the size and shape indicated by the frame field, providing an intuitive visualization without the need to compute a full quadrilateral remeshing as in Section 7.

To make the resulting images readable even in the presence of strong scale variations, we superimpose two different cross-hatching patterns: a thick, black one at a given scale proportional to the frame field scale, and a thinner, light blue one, which is 0.2 times finer.



We implemented the above concept in a realtime, texture based renderer for frame fields defined over triangle meshes. First, we procedurally generate a pseudo-random, blue-noise, tileable texture, which features an irregular layout of vertical segments of varying lengths (see inset). A simple dart-throwing technique is employed to populate the texture with vertical strokes while ensuring that no two segments are closer than a given threshold. Next, a two-layered UV-mapping is propagated over the surface following the frame field, i.e., the parametrization of the vertices of a given triangle t has a Jacobian that is close to the frame f_t. Similarly to [Sorkine et al. 2002], we start at an arbitrarily chosen triangle and iteratively expand over neighbors, until each triangle is covered twice: exactly once for each of the two directions of the frame field. During the propagation, the UV coordinates computed by different triangles incident on the same vertex are averaged together if they are similar enough, otherwise texture seams are generated. In the final rendering, each fragment accesses the texture twice at the two UV locations, creating the two superimposed directions forming the cross-hatching (for each cross hatching pattern).

This simple approach is effective in hiding texture seams thanks to the stochastic nature of the texture, and because the seams of the two superimposed directions appear in different positions. Since no semantics is attached to the stroke length, seams that are almost *horizontal* in texture space are less evident; for this reason, during the UV propagation, we prioritize expansion over edges that are close to vertical in texture space. Similarly, texture distortions in the *vertical* texture direction do not disrupt the directions of the lines: when deciding whether to lump texture coordinates, we can tolerate more discrepancies in the V direction, thus reducing the number of seams.

The visualizations of the frame field appearing in various figures are realtime screenshots of this simple technique.

5 Frame field interpolation

Many natural quantities can be encoded as a frame field, such as the curvature tensor, or the stress tensor of a self-supporting surface [Vouga et al. 2012]. We provide a user-friendly way for designing frame fields, which can be employed in applications such as anisotropic meshing or user-defined image deformation. This method consists in generating a smooth frame field that satisfies a sparse set of given constraints $\hat{f}_1, \ldots, \hat{f}_k$ on triangles $t_1, \ldots, t_k$.

The canonical decomposition of Lemma 3.1 is applied to each constrained frame, obtaining a set of cross field constraints $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_k$ and a set of linear transformations $\hat{\mathbf{W}}_1, \ldots, \hat{\mathbf{W}}_k$. Note that both the crosses and the matrices are defined in the local reference systems at each triangle, as described in Section 3.

The discrete smooth cross field $\mathcal{X}$ that satisfies the constraints $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_k$ on the corresponding faces is computed using [Bommes et al. 2009]. We interpolate the coefficients of the symmetric tensor field $\mathcal{W}$ independently as harmonic scalar fields that satisfy the constraints. However, this cannot be done directly, since each constraint is expressed in a different reference system, and it is impossible to pick a consistent reference system for the entire surface. Following an approach similar to [Palacios and Zhang 2007], we define a face-based discrete Laplacian operator $\mathbf{L}^{\mathbf{B}}$ that encodes the changes of the reference systems and we use it to interpolate the coefficients of the symmetric tensors as follows:

$$\mathbf{L}^{\mathbf{B}}(\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_n^T)^T = \mathbf{0} \qquad (4)$$

$$\text{subject to} \quad \mathbf{w}_j = \hat{\mathbf{w}}_j, \quad j \in \mathcal{C}$$

where each $\mathbf{w}_i \in \mathbb{R}^3$ is a vector that contains the three entries of $\mathbf{W}_i$ (one coefficient is redundant since the matrix is symmetric), $\mathcal{C}$ is the set of the constrained faces and $\hat{\mathbf{w}}_j$ are the constraints. Note that the different components of the $\mathbf{w}_i$'s are not separable due to the presence of rotations that transform between adjacent bases; therefore, the sparse matrix $\mathbf{L}^{\mathbf{B}}$ has size $3n \times 3n$, where n is the number of faces of mesh $\mathcal{M}$. The entries of $\mathbf{L}^{\mathbf{B}}$ are described in Appendix A. The validity of the interpolation result is guaranteed by the following lemma, whose proof is provided in Appendix B.

Lemma 5.1. *All matrices $\mathbf{W}_i$ obtained by solving Eq. (4) are SPD, provided that all constraints $\hat{\mathbf{W}}_j$ are SPD.*

Note that $\mathcal{W}$ is a generic SPD tensor field, which cannot be directly interpolated as a 2-RoSy field, since the latter is equivalent to a traceless symmetric tensor (see Appendix of [Palacios and Zhang 2007]).

Alternative parametrization. The fact that our frame representation is unique and independent of the labeling of the four frame vectors as $\mathbf{v}$, $\mathbf{w}$, etc., eases the task of interpolating between frames whose vectors are labelled inconsistently, by simply interpolating between the three unique coefficients of $\mathbf{W}$ (regardless of labeling). For this reason, we prefer using this representation to alternatives, like using the matrix $\mathbf{P}$ (as done in [Alexa et al. 2000] for planar

mesh morphing), even though its coefficients a, b, c have the benefit of being more directly linked to intuitive quantities like scale and skeweness (see Figure 3). Direct interpolation of such geometric properties has been used, e.g., in [Pal et al. 2014] in a much simpler setting, to interpolate the scales of orthogonal reference systems induced by a singularity-free direction field. In our general setting, this type of interpolation requires a much more complex derivation of the Laplace operator to account for inconsistent labeling of frame vectors. Of course, the choice of field representation affects the interpolation results, but according to our experiments, the differences appear to be small in practice (see Figure 4).

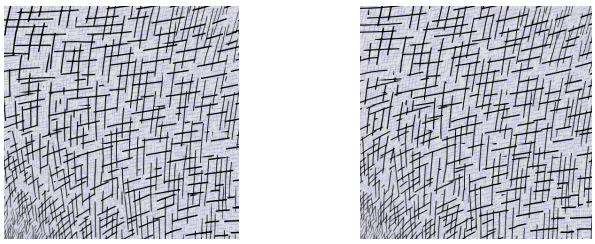


Figure 4: *Different parameterizations for interpolation. The frame field is constrained just at the four corners. Left: interpolation of the coefficients of* $\mathbf{W}$. *Right: direct interpolation of the magnitudes of* $\mathbf{v}$ *and* $\mathbf{w}$ *and the angle between them.*

6 Frame-driven deformation

A frame-driven deformation of a surface strives to warp a frame field defined over the surface into a cross field. In this way, we can apply the existing technology for processing cross fields to our frame fields, in particular, to the generation of anisotropic and non-uniform quadrilateral meshes (Section 7).

3D embedding. Let $(\mathcal{A}, g_{\mathcal{F}})$ be an abstract manifold, where $g_{\mathcal{F}} = \mathbf{W}^{-T}\mathbf{W}^{-1}$ is the metric implied by a smooth frame field $\mathcal{F}$ defined on $\mathcal{A}$. We want to compute an embedding $\mathcal{S}'$ of this abstract manifold such that the canonical metric on $\mathcal{S}'$ (induced by the ambient Euclidean space) is similar to $g_{\mathcal{F}}$, or equivalently, such that the frame field is warped into a cross field.

It follows from the Nash embedding theorem [Nash 1956] that, in general, a Euclidean embedding of $(\mathcal{A}, g_{\mathcal{F}})$ exists in higher dimensions; in particular, Lévy and Bonneel [2012] report that 10 dimensions are necessary to obtain a smooth embedding for a 2-manifold. However, for our applications, such a high-dimensional embedding would be impractical: it cannot be used for image and surface deformation, and it is unclear how it could be useful for anisotropic meshing. As noted by Kovacs et al. [2010], there is no direct connection between the intrinsic curvature of a surface embedded in a high-dimensional space and the actual singularities of a cross field. Moreover, our experimental results indicate that in practice, three dimensions are sufficient to express $g_{\mathcal{F}}$ sufficiently well. For these reasons, we obtain an approximation of the metric by computing an embedding in 3D, that is, by deforming the input surface $\mathcal{S}$.

Deformation energy. In our discretization, the frame field is defined on all triangles of a mesh $\mathcal{M}$: given a triangle t of $\mathcal{M}$, we denote by f_t the frame associated with t and by $\mathbf{W}_t$ the corresponding $\mathbf{W}$ matrix computed in the local coordinate system of t. The map $\mathbf{W}_t$ transforms the cross $\mathbf{x}_t$ in triangle t into the frame f_t. Therefore, the ideal deformation for the triangle t is $\mathbf{W}_t^{-1}$. As discussed before, it might be impossible to exactly deform all triangles to their ideal target shape and maintain a closed mesh; we thus minimize an energy that penalizes deviations from the ideal transformations:

$$\mathcal{E}(\mathbf{p}') = \sum_{t\in\mathcal{M}} \min_{\mathbf{Q}_t\in SO(3)} A_t \left\| \mathbf{J}_t(\mathbf{p}') - \mathbf{Q}_t\tilde{\mathbf{W}}_t^{-1} \right\|_F^2 \qquad (5)$$

where $\mathbf{p}'$ is the vector of unknown vertex positions of the deformed mesh, A_t is the area of t, $\mathbf{J}_t(\mathbf{p}')$ is the Jacobian of the deformation of triangle t, and $\tilde{\mathbf{W}}_t$ is the 3×3 version of $\mathbf{W}_t$, expressed in the global 3D coordinate system. The best-fitting (unknown) rotation matrix $\mathbf{Q}_t$ provides an additional degree of freedom that factors out triangle rotations, providing a richer space to minimize the energy.

Block coordinate descent. The energy (5) is similar to the ARAP energy proposed in [Sorkine and Alexa 2007; Liu et al. 2008]. This energy can be quickly and reliably minimized using block coordinate descent: we iteratively alternate between fixing $\mathbf{Q}$ and optimizing for $\mathbf{p}'$ (which amounts to solving a sparse linear system), and fixing $\mathbf{p}'$ and optimizing for $\mathbf{Q}$ (which amounts to a set of local Procrustes problems that can be solved using SVD). The gradient of the energy w.r.t. $\mathbf{p}'$, for fixed $\mathbf{Q}$, can be compactly written as:

$$\nabla\mathcal{E}(\mathbf{p}') = -4(\mathbf{L}\mathbf{p}' - \mathbf{b}) \qquad (6)$$

where $\mathbf{L}$ is the standard cotan Laplacian matrix of $\mathcal{M}$ and $\mathbf{b}$ is defined row-wise as:

$$\mathbf{b}_i = \sum_{j\in\mathcal{N}(i)} \frac{1}{2}\big(\cot\theta_{ij}\mathbf{Q}_{t(i,j)}\tilde{\mathbf{W}}^{-1}_{t(i,j)} + \cot\theta_{ji}\mathbf{Q}_{t(j,i)}\tilde{\mathbf{W}}^{-1}_{t(j,i)}\big)(p_j - p_i), \qquad (7)$$

where p_i and p_j are vertex positions of the input mesh, $\mathcal{N}(i)$ denotes the indices of the vertices adjacent to vertex i, $t(i,j)$ is the triangle to the left of the half-edge $(p_j - p_i)$, and θ_{ij} is the angle of $t(i,j)$ opposite of this half-edge. For the derivation of the gradient see Appendix C.

Initialization. To initialize the optimization, we use the vertex positions in the original mesh as $\mathbf{p}'$ and we randomly generate rotations by very small angles to initialize $\mathbf{Q}$. Such small rotations are not necessary (the identity matrix is actually the most natural candidate) but we found that they help the solver to avoid local minima in areas with zero mean curvature, while not affecting the optimization anywhere else. Note that self-intersections commonly appear during deformation (see the ears in Figure 1) and do not require any special treatment.

7 Frame-field aligned quadrangulation

Since the deformation problem in the previous section is over-constrained, the deformed frame field is not exactly a cross field. For each triangle t' that is a deformed version of triangle t, we substitute the deformed frame $f_{t'} = \mathbf{J}_t f_t$ (where $\mathbf{J}_t$ is the Jacobian of the deformation) with its nearest cross, which is obtained through polar decomposition (see Canonical decomposition in Section 3).

Next, we smooth the cross field and use it to parametrize the deformed mesh with [Bommes et al. 2009]. We then remesh the deformed surface with [Ebke et al. 2013], producing a uniform quad remeshing. Note that any other method capable of smoothing a standard cross field and computing a quad mesh aligned with it can be employed instead. In this sense, our framework can be seen as a way to generalize any such system to deal with anisotropy, scale variation and non-orthogonality.

In the last step, we apply the reverse deformation to the resulting quad mesh: the final result is a remeshing of the original mesh $\mathcal{M}$ that is compliant with the original frame field. The reverse

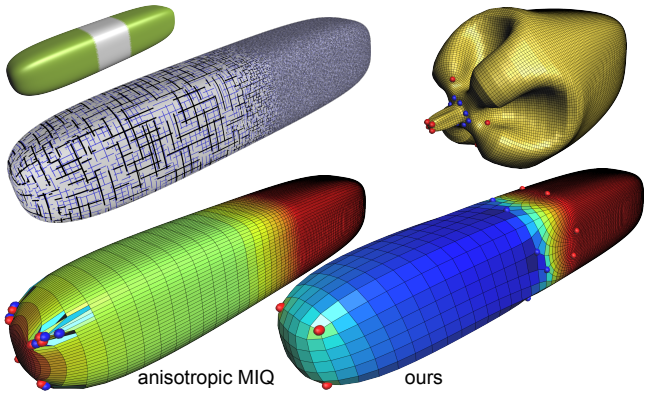


Figure 5: *Frame-field aligned quadrangulation vs. Anisotropic MIQ [Bommes et al. 2009]: a frame field imposing a strong transition of scale along the Cigar dataset (top left); the corresponding frame-driven deformation (top right); a remeshing obtained with Anisotropic MIQ (bottom left) and our remeshing (bottom right). Green areas in the top left picture represent constrained parts: the ratio between the scales imposed on the two parts is 10.*

Dataset	# Tris	E_d	Time (sec.)				# Quads
			Int.	Deform.	Parametr.	Remesh.	
Legs	29k	0.033	3.05	3.07	12.66	0.84	8.6k
Hand	35k	0.186	7.11	4.58	20.59	1.35	19.5k
Cube	40k	0.018	12.67	17.85	39.36	7.72	5.5k
Cigar	49k	0.020	22.22	23.59	4.36	1.77	28.8k
Face	52k	0.058	7.92	12.81	49.23	1.32	6.2k
Bunny	56k	0.045	19.49	8.38	40.13	1.62	14.4k
Horse	77k	0.050	11.23	10.08	58.77	2.46	9.3k
Armadillo	155k	0.072	25.82	22.33	802.56	3.92	18.2k
Ptex	46k	0.108	10.14	23.89	12.62	1.20	5.4k
Fertility	46k	0.416	8.55	22.40	3.47	1.15	18.5k

Table 1: *Statistics on our datasets: input size (*# Tris*), metric error of the field-driven deformation (E_d), processing time in seconds and output size (*# Quads*). We measured the processing time for all steps of our method, i.e. frame field interpolation (*Int.*, Section 5), surface deformation (*Deform.*, Section 6), cross field smoothing and surface parametrization (*Parametr.*, Section 7) and remeshing (*Remesh.*). For cross field smoothing and parametrization we use MIQ [Bommes et al. 2009] and for quad remeshing we use QEx [Ebke et al. 2013].*

deformation is computed trivially using barycentric coordinates, since the underlying triangle meshes (deformed and original) share the same connectivity, and the quad remeshing algorithm generates vertices that lie *on* the deformed mesh.

If the input frame field was obtained through interpolation of constraints, the same constraints are maintained during the cross field smoothing phase on the deformed surface. The cross field in all other triangles is interpolated as in standard MIQ. In this way, new singularities may arise in the smoothed cross field, reflecting the intrinsic (Gaussian) curvature of the deformed surface [Bommes et al. 2013a]. Intuitively, these singularities include the ones necessary to accommodate for changes in tessellation density in the final result (see Figure 5 and other figures throughout this paper for examples). In other words, after the deformation, the singularities needed for adaptive tessellation become equivalent to the singularities needed for a uniform tessellation and can be dealt with as such.

Anisotropic MIQ. We compare our approach to Anisotropic MIQ proposed in [Bommes et al. 2009] for aligning meshes to curvature fields. In Anisotropic MIQ, anisotropic scales associated with curvature are used as gradients during the parametrization phase to scale mesh elements accordingly. Since this is done after the cross field has been fixed, its topology cannot be altered, so no new singularities may arise to accommodate for the changes in scale. In the example shown in Figure 5 we use an orthogonal frame field with a large change of scale. The Anisotropic MIQ is able to handle the change of scale along the longitudinal direction, which does not require transitions in the connectivity. The size of elements in the transversal direction is maintained constant. This is a compromise between the scales enforced at the two halves of the *Cigar*; many irregular vertices are crowded together near the four singularities of the field to compensate for an abrupt change of scale towards the extremities.

8 Results and applications

The statistics on our experiments are reported in Table 1. We assess the quality of our frame-driven deformation by measuring how well it succeeds in warping the frame field into a cross field. Let $\mathbf{W}'_t$ be the $\mathbf{W}$ matrix of the warped field $\mathbf{J}_t f_t$ at triangle t, where $\mathbf{J}_t$ is the Jacobian of the deformation. We measure how much $\mathbf{W}'_t$ differs from the identity and we integrate this over the whole mesh:

$$E_d = \frac{1}{A_{\mathcal{M}}} \sum_{t \in \mathcal{M}} A_t \left\| \mathbf{W}'_t - \mathbf{I} \right\|_F ,$$

where $A_{\mathcal{M}}$ is the total area of mesh $\mathcal{M}$ and A_t is the area of a triangle t. The deformation error is moderate in all experiments, suggesting that the computed 3D embedding is a good approximation of the metric implied by the frame field (Table 1).

Depending on the provided constraints, the deformation obtained by minimizing Eq. (5) might produce meshes that are not sufficiently smooth to be processed with MIQ. In this case, we add a regularization term to the deformation energy as follows:

$$\mathcal{E}_r(\mathbf{p}') \;=\; (1-\lambda)\,\mathcal{E}(\mathbf{p}') + \lambda \left\| \mathbf{L}(\mathbf{p}' - \mathbf{p}) \right\|^2 . \qquad (8)$$

A fixed $\lambda = 0.1$ was used for all experiments.

Isotropic scale. Figure 6 illustrates an example of large variation of *isotropic* scale on the *Cube*. One side of the surface is constrained to have an orthogonal, isotropic frame field aligned to principal curvature directions. A similar frame field, scaled down six times, is the constraint on the opposite side. The region in the middle

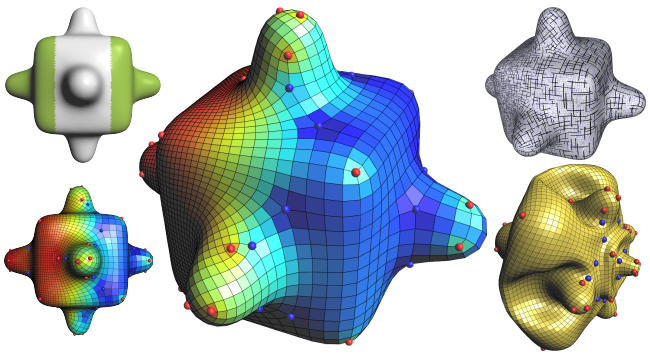


Figure 6: *Large change of scale, isotropic: The singularities necessary to warrant transition of scale naturally arise when smoothing the cross field on the deformed mesh.*

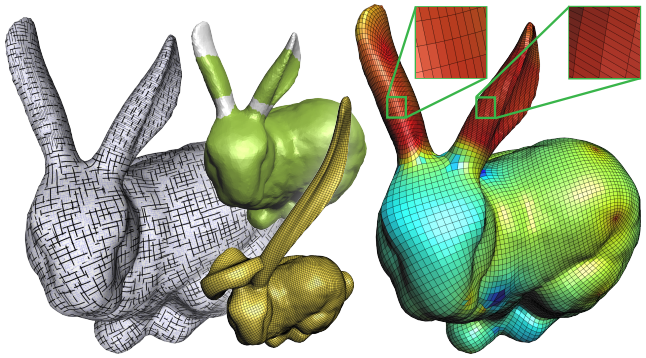


Figure 7: *Large change of scale, anisotropic: Both ears of the Bunny are assigned an anisotropic frame field, one stretched in the longitudinal direction and the other stretched in the transversal direction.*

is unconstrained to allow for a smooth scale transition. Figure 5 illustrates another such example, where the scales imposed on the two halves of the *Cigar* differ by a factor of 10, and the transition region towards the middle of the *Cigar* is relatively narrow. In both cases, the alignment in the constrained regions is preserved, and the singularities necessary to accommodate for the change of scale are automatically introduced. Note that they are always placed in regions where the deformed mesh has a high Gaussian curvature.

Anisotropic scale. Figure 7 illustrates an example of large anisotropic scaling. The frame field is isotropic, at unit scale and aligned with the principal curvature directions on the whole *Bunny* except at its ears, where anisotropic scaling is added by imposing an opposite stretch direction on each ear. Our method remeshes the entire model while satisfying the constraints even on the left ear, where the anisotropy is perpendicular to the mesh features.

Skewness. Figure 8 shows an example of large skewness on the *Legs* dataset. The frame-driven deformation greatly stretches the legs to compensate for the high shearing, and the singularities necessary to accommodate the transition from diagonal to longitudinal orientation are automatically introduced.

Sparse constraints. Given a set of manually placed sparse constraints, our method can generate high-quality meshes that are similar to those manually generated by an artist. Figures 1 and 9 show

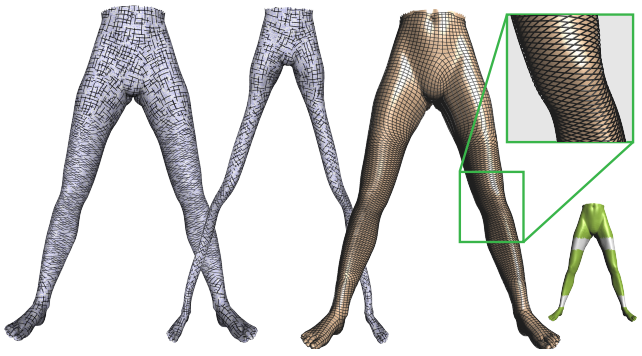


Figure 8: *A highly skewed frame field is imposed in the middle part of the legs to obtain a diagonal alignment of the "fishnet stockings", while an orthogonal, longitudinal alignment is imposed at the feet and the hips. The singularities necessary to change the orientation of the meshing are inserted by our algorithm.*

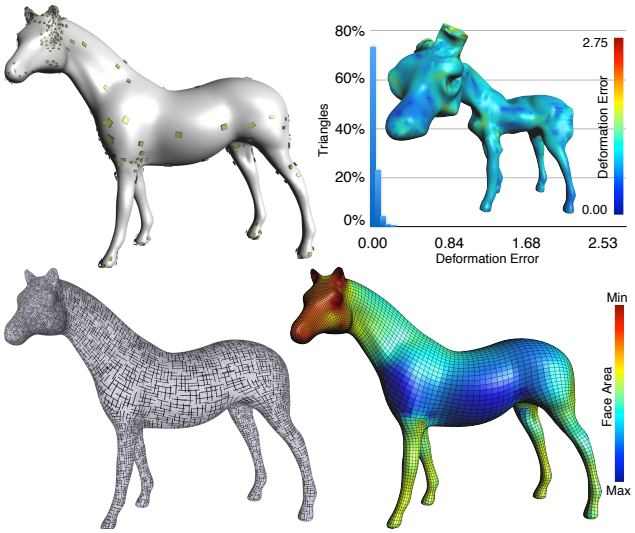


Figure 9: *Field design and quad remeshing from sparse constraints. A sparse set of constraints (green quads) is specified on the input surface to orient and size the frame field; a dense frame field is obtained by interpolation, and the surface is deformed to obtain a cross field. Top right: Color represents deformation error on a square root scale, see Table 1. The histogram represents the distribution of the per-triangle deformation error. Bottom right: The final quad mesh adheres to the input constraints and closely follows the interpolated frame field (color represents face area).*

two examples on the *Head* and *Horse* datasets, respectively. The green quads placed on the input meshes represent the frame field constraints. A point on the original mesh that projects onto a point on a constraint-quad is constrained to have a frame field that is formed as follows: we take the "horizontal" and "vertical" barycentric coordinates of the projected point (corresponding to bilinear interpolation on the quad) and use them to interpolate the two horizontal and vertical edges of the quad, respectively. The two resulting vectors form the constraint frame.

Dense input. In Figure 10 an artist specified the frame field on the hands and feet of the *Armadillo* by drawing a dense quad mesh. The frame field is then interpolated over the rest of the surface, and a pure quad mesh is generated. The singularities forced by the user constraints are preserved, and new ones are added to smoothly fill the unconstrained region. Note that our algorithm is not able to exactly preserve the connectivity provided by the artist.

Curvature-aligned. Figure 11 shows an example of curvature-aligned meshing on the *Hand* dataset. The frame field is defined by principal curvatures (including both directions and magnitudes) where the curvature anisotropy is high enough, and it is smoothly interpolated on the remaining faces. Note how the frame-driven deformation tends to compress the fingers along their length and squeeze the flattest region of the palm. The resulting quad mesh correctly follows the anatomy of the hand and exhibits anisotropic elements that are consistent with the curvature field.

Ptex conversion. Our algorithm can be used to convert a standard UV mapping to a Ptex mesh [Burley and Lacewell 2008]. The frame field is defined as the Jacobian of the UV mapping on faces that are far enough from the parametrization seams, and it is interpolated on the other faces. As shown in Figure 12, the quad mesh generated by our algorithm can be directly used for Ptex, since it perfectly aligns with the UV parametrization and gracefully handles the mismatches of scale and anisotropy at the seams.

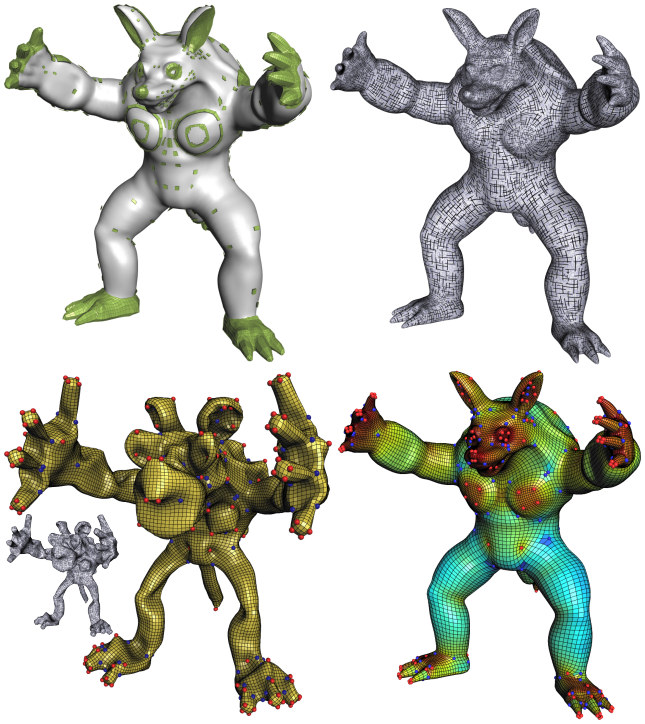


Figure 10: *An artist manually drawn a quad mesh on hands and feet plus a sparse set of constraints (green quads). Our method produces a complete quad mesh which preserves the singularities specified by the artist and add new ones to ensure a smooth interpolation of the constraints. The layout is analogous to Figure 1; red and blue bullets depict irregular vertices.*

Image warping and surface deformation. Figure 13 presents two simple examples of image and surface deformation. Note that differently from other deformation methods, our deformation is entirely intrinsic. We believe that the use of frame fields can complement traditional methods for image warping and deformation. However, a more thorough study and experimentation is necessary to make this approach practicable.

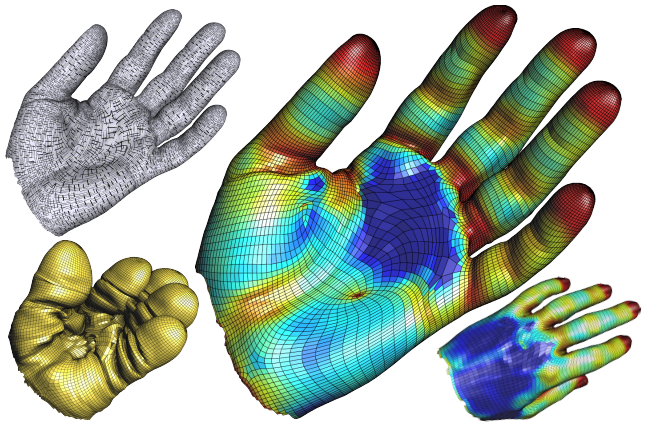


Figure 11: *An orthogonal but highly non-uniform and anisotropic frame field is interpolated from principal curvatures; the resulting quad mesh closely follows the anatomy of the hand.*

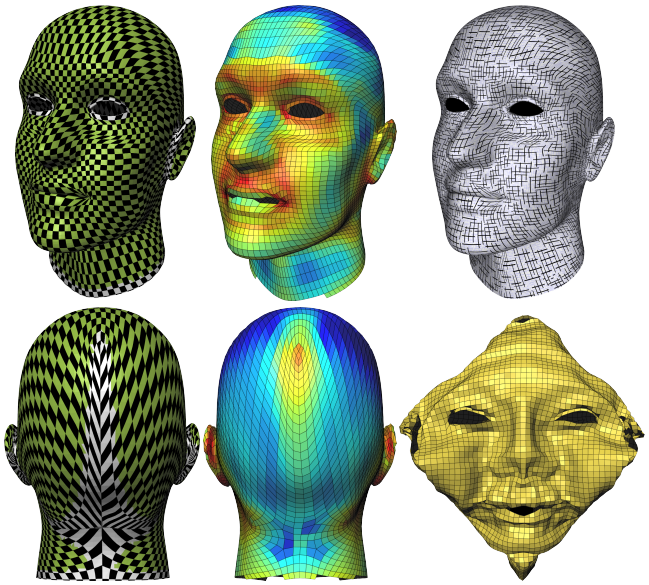


Figure 12: *We convert a standard UV mapping to a Ptex mesh [Burley and Lacewell 2008] by using the parametrization's Jacobian as a frame field. The checkerboard texture depicts the input UV parametrization; the frame field is constrained in green areas and it is interpolated in white areas.*

9 Limitations and concluding remarks

Frame fields represent a smoothly varying linear transformation over the tangent space of a surface and are a natural extension of cross fields enriched with scale, anisotropy and skewness. We have defined frame fields formally, discretized them over triangle meshes and proposed a method to generate fields that satisfy a set of user-provided constraints. We have demonstrated how they can be applied to solve practical geometry processing problems such as the design of anisotropic and non-homogeneous quad meshes and image or surface deformation.

A practical limitation of our approach is the fact that the frame-driven deformation (Section 6) may deteriorate the quality of the triangulation, especially when the constraints imply a large difference in scale.

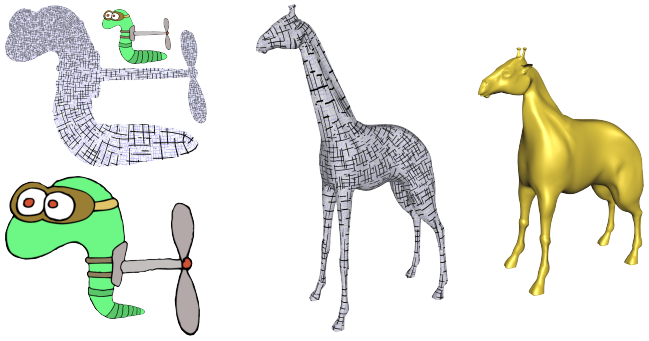


Figure 13: *Image and surface deformation. The Worm is scaled progressively along its spine, its airscrew is made bigger and the post shorter, by placing a few frame constraints, interpolating them over the entire domain and then performing frame-driven deformation. The Giraffe is deformed into a horse-like animal by placing anisotropic scaling constraints on its neck and legs.*

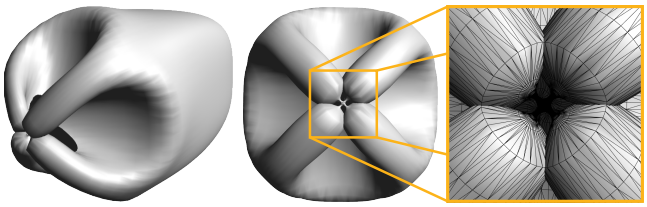


Figure 14: *An experiment similar to Figure 5, but where we try to impose a difference of a factor 100 in the scales. Our implementation of MIQ is not able to generate a pure isotropic quad mesh due to the low element quality on the deformed model.*

Triangle size differences can introduce numerical problems in the MIQ method [Bommes et al. 2009] which we use to quadrangulate the deformed surface. In practice, we did not notice any problems as long as the ratio between the largest and the smallest scales of frames was lower than 10, which is sufficient for many practical scenarios. We show in Figure 14 a case where our implementation of MIQ failed to generate a pure quad mesh due to the presence of more than a thousand flipped faces in the parametrization step. A possible way to alleviate this problem is to perform isotropic triangular remeshing after the deformation. This is not an intrinsic limitation of our method, because any technique capable of producing uniform quadrangulations over a given surface can be employed on our deformed mesh in lieu of [Bommes et al. 2009]. Another limitation is the lack of guarantees that our Euclidean embedding process terminates with low metric error, although we only noticed this problem for almost degenerate fields, as shown in Figure 15.

Frame fields open interesting possibilities for future work, both in terms of generation techniques and novel applications. A custom user interface could be designed to allow artists to intuitively and interactively specify the scaling and anisotropy constraints. This interface could take advantage of the possibility to handle orientation constraints separately from sizing constraints, and enable to warm-start each optimization step, improving the results incrementally as additional constraints are added. It would also be interesting to compute the surface embedding directly from the sparse set of frame constraints, without generating an intermediate frame field that is currently used to guide the deformation algorithm. By combining the two steps, it might be possible to improve the locations and the number of field singularities.

We believe that this work may have a potentially high impact in many geometry processing applications, such as the design of anisotropic and skewed planar quadrilateral meshes and in the generation of brick tessellations for structural geometry. A generalization of 6-symmetry fields with non-uniform lengths and non-uniform rotation angles could be used to create anisotropic, semi-regular triangle meshes. Extending frame fields to 3D could provide the foundation for anisotropic and adaptive hexahedral meshing.

Acknowledgements

The authors wish to thank: David Bommes for providing his Mixed-Integer solver; Nico Pietroni for his help in implementing the MIQ global parametrization; Hans-Christin Ebke for providing the code of the libQEx quadrangulator; Olga Diamanti for her ARAP implementation; Nikolas De Giorgis for the code to compute principal curvature directions; the repository of Aim@Shape, Alec Jacobson, Ladislav Kavan, Kenshi Takayama for providing datasets; Emily Whiting for narrating the accompanying video. This work was supported in part by the ERC Starting Grant iModel (StG-2012-306877).

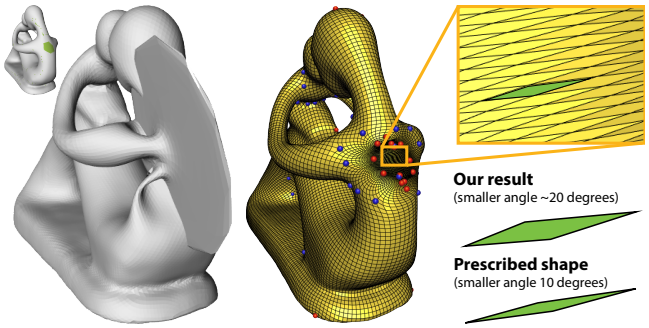


Figure 15: *We prescribe an extreme skewness in the green region in the back of the Fertility model. Our algorithm generates a pure quad mesh that does not precisely satisfy the provided constraints: We prescribed a minimal angle of 10 degrees, but our mesh has a minimal angle of around 20 degrees in the constrained region.*

References

ALEXA, M., COHEN-OR, D., AND LEVIN, D. 2000. As-rigid-as-possible shape interpolation. In *derACM SIGGRAPH*, 157–164.

ALLIEZ, P., COHEN-STEINER, D., DEVILLERS, O., LÉVY, B., AND DESBRUN, M. 2003. Anisotropic polygonal remeshing. *ACM Trans. Graph. 22*, 3, 485–493.

BOMMES, D., ZIMMER, H., AND KOBBELT, L. 2009. Mixed-integer quadrangulation. *ACM Trans. Graph. 28*, 3, 77:1–77:10.

BOMMES, D., LÉVY, B., PIETRONI, N., PUPPO, E., SILVA, C., TARINI, M., AND ZORIN, D. 2013. Quad-mesh generation and processing: A survey. *Comput. Graph. Forum 32*, 51–76.

BOMMES, D., CAMPEN, M., EBKE, H.-C., ALLIEZ, P., AND KOBBELT, L. 2013. Integer-grid maps for reliable quad meshing. *ACM Trans. Graph. 32*, 4, 98:1–98:12.

BURLEY, B., AND LACEWELL, D. 2008. Ptex: Per-face texture mapping for production rendering. In *Proc. of Eurographics Conference on Rendering*, 1155–1164.

CRANE, K., DESBRUN, M., AND SCHRÖDER, P. 2010. Trivial connections on discrete surfaces. *Comput. Graph. Forum 29*, 5, 1525–1533.

DEROSE, T., KASS, M., AND TRUONG, T. 1998. Subdivision surfaces in character animation. In *ACM SIGGRAPH*, 85–94.

EBKE, H.-C., BOMMES, D., CAMPEN, M., AND KOBBELT, L. 2013. Qex: Robust quad mesh extraction. *ACM Trans. Graph. 32*, 6, 168:1–168:10.

FISHER, M., SCHRÖDER, P., DESBRUN, M., AND HOPPE, H. 2007. Design of tangent vector fields. *ACM Trans. Graph. 26*, 3, 56:1–56:9.

HERTZMANN, A., AND ZORIN, D. 2000. Illustrating smooth surfaces. In *Proc. ACM SIGGRAPH*, 517–526.

HUANG, J., ZHANG, M., MA, J., LIU, X., KOBBELT, L., AND BAO, H. 2008. Spectral quadrangulation with orientation and alignment control. *ACM Trans. Graph. 27*, 5, 147:1–147:9.

JACOBSON, A., BARAN, I., POPOVIĆ, J., AND SORKINE, O. 2011. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph. 30*, 4, 78:1–78:8.

KÄLBERER, F., NIESER, M., AND POLTHIER, K. 2007. QuadCover – surface parameterization using branched coverings. *Comput. Graph. Forum 26*, 3, 375–384.

KNÖPPEL, F., CRANE, K., PINKALL, U., AND SCHRÖDER, P. 2013. Globally optimal direction fields. *ACM Trans. Graph. 32*, 4, 59:1–59:10.

KOVACS, D., MYLES, A., AND ZORIN, D. 2010. Anisotropic quadrangulation. In *Proc. ACM Symposium on Solid and Physical Modeling*, 137–146.

LAI, Y.-K., JIN, M., XIE, X., HE, Y., PALACIOS, J., ZHANG, E., HU, S.-M., AND GU, X. 2010. Metric-driven RoSy field design and remeshing. *IEEE Trans. Vis. Comput. Graph. 16*, 1, 95–108.

LEFEBVRE, S., AND HOPPE, H. 2006. Appearance-space texture synthesis. *ACM Trans. Graph. 25*, 3, 541–548.

LÉVY, B., AND BONNEEL, N. 2012. Variational anisotropic surface meshing with Voronoi parallel linear enumeration. In *Proc. 21st Int. Meshing Roundtable*, 349–366.

LI, Y., BAO, F., ZHANG, E., KOBAYASHI, Y., AND WONKA, P. 2011. Geometry synthesis on surfaces using field-guided shape grammars. *IEEE Trans. Vis. Comput. Graph. 17*, 2, 231–243.

LING, R., HUANG, J., SUN, F., JUTTLER, B., BAO, H., AND WANG, W. 2011. Spectral quadrangulation with boundary conformation. Tech. rep., TR-2011-13, The University of Hong Kong.

LIU, L., ZHANG, L., XU, Y., GOTSMAN, C., AND GORTLER, S. J. 2008. A local/global approach to mesh parameterization. *Comput. Graph. Forum 27*, 5, 1495–1504.

LIU, Y., XU, W., WANG, J., ZHU, L., GUO, B., CHEN, F., AND WANG, G. 2011. General planar quadrilateral mesh design using conjugate direction field. *ACM Trans. Graph. 30*, 6.

MARINOV, M., AND KOBBELT, L. 2004. Direct anisotropic quad-dominant remeshing. In *Proc. 12th Pacific Graphics*, 207–216.

NASH, J. 1956. The imbedding problem for Riemannian manifolds. *Annals of Mathematics 63*, 1, 20–63.

PAL, K., SCHÜLLER, C., PANOZZO, D., SORKINE-HORNUNG, O., AND WEYRICH, T. 2014. Interactive restoration of historical documents. *Comput. Graph. Forum (Proc. Eurographics) 33*, 2.

PALACIOS, J., AND ZHANG, E. 2007. Rotational symmetry field design on surfaces. *ACM Trans. Graph. 26*, 3.

PALACIOS, J., AND ZHANG, E. 2011. Interactive visualization of rotational symmetry fields on surfaces. *IEEE Trans. Vis. Comput. Graph. 17*, 7, 947–955.

PANOZZO, D., BLOCK, P., AND SORKINE-HORNUNG, O. 2013. Designing unreinforced masonry models. *ACM Trans. Graph. 32*, 4, 91:1–91:12.

PIETRONI, N., TARINI, M., SORKINE, O., AND ZORIN, D. 2011. Global parametrization of range image sets. *ACM Trans. Graph. 30*, 6.

PINKALL, U., JUNI, S. D., AND POLTHIER, K. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics 2*, 1, 15–36.

RAY, N., LI, W. C., LÉVY, B., SHEFFER, A., AND ALLIEZ, P. 2006. Periodic global parameterization. *ACM Trans. Graph. 25*, 1460–1485.

RAY, N., VALLET, B., LI, W. C., AND LÉVY, B. 2008. N-symmetry direction field design. *ACM Trans. Graph. 27*, 2, 10:1–10:13.

RAY, N., VALLET, B., ALONSO, L., AND LEVY, B. 2009. Geometry-aware direction field processing. *ACM Trans. Graph. 29*, 1:1–1:11.

ROSSIGNAC, J., AND VINACUA, A. 2011. Steady affine motions and morphs. *ACM Trans. Graph. 30*, 5, 116:1–116:16.

SCHAEFER, S., MCPHAIL, T., AND WARREN, J. D. 2006. Image deformation using moving least squares. *ACM Trans. Graph. 25*, 3, 533–540.

SORKINE, O., AND ALEXA, M. 2007. As-rigid-as-possible surface modeling. In *Proc. Symp. Geometry Processing*, 109–116.

SORKINE, O., COHEN-OR, D., GOLDENTHAL, R., AND LISCHINSKI, D. 2002. Bounded-distortion piecewise mesh parameterization. In *IEEE Visualization*, 355–362.

TAKAYAMA, K., PANOZZO, D., SORKINE-HORNUNG, A., AND SORKINE-HORNUNG, O. 2013. Sketch-based generation and editing of quad meshes. *ACM Trans. Graph. 32*, 4, 97:1–97:8.

TARINI, M., PUPPO, E., PANOZZO, D., PIETRONI, N., AND CIGNONI, P. 2011. Simple quad domains for field aligned mesh parametrization. *ACM Trans. Graph. 30*, 6.

VOUGA, E., HÖBINGER, M., WALLNER, J., AND POTTMANN, H. 2012. Design of self-supporting surfaces. *ACM Trans. Graph. 31*, 87:1–87:11.

WARDETZKY, M., MATHUR, S., KÄLBERER, F., AND GRINSPUN, E. 2007. Discrete Laplace operators: No free lunch. In *Proc. Symp. Geometry Processing*, 33–37.

YÜCER, K., JACOBSON, A., HORNUNG, A., AND SORKINE, O. 2012. Transfusive image manipulation. *ACM Trans. Graph. 31*, 6, 176:1–176:9.

ZHANG, E., MISCHAIKOW, K., AND TURK, G. 2006. Vector field design on surfaces. *ACM Trans. Graph. 25*, 4, 1294–1326.

ZHANG, M., HUANG, J., LIU, X., AND BAO, H. 2010. A wave-based anisotropic quadrangulation method. *ACM Trans. Graph. 29*, 118:1–118:8.

ZHONG, Z., GUO, X., WANG, W., LÉVY, B., SUN, F., LIU, Y., AND MAO, W. 2013. Particle-based anisotropic surface meshing. *ACM Trans. Graph. 32*, 4, 99:1–99:14.

A Derivation of $\mathbf{L^B}$

Let t_i, t_j be two adjacent triangles of $\mathcal{M}$, let $\mathbf{W}_i$, $\mathbf{W}_j$ be their $\mathbf{W}$ matrices, expressed in the respective local bases $\mathbf{B}_i, \mathbf{B}_j$, and let $\mathbf{R}_{ij}$ be the change of basis between t_i and t_j. The matrices $\mathbf{R}_{ij}$ are fixed per given mesh and choice of local bases on its faces. Let us denote

$$\mathbf{W}_i = \begin{bmatrix} w_{ia} & w_{ib} \\ w_{ib} & w_{ic} \end{bmatrix} \qquad \mathbf{R}_{ij} = \begin{bmatrix} r_{ija} & r_{ijb} \\ r_{ijc} & r_{ijd} \end{bmatrix}. \tag{9}$$

A uniform Laplacian for scalar functions on $\mathcal{M}$ that are constant on the faces is an $n \times n$ matrix defined as

$$\mathbf{L} = \sum_{(i,j)\in\mathcal{H}} \mathbf{L}_{ij}, \text{ where } \mathbf{L}_{ij}[i,i] = -1,\ \mathbf{L}_{ij}[i,j] = 1. \tag{10}$$

$\mathbf{L}_{ij}$ are $n \times n$ matrices whose entries are zeros except entries $[i,i]$ and $[i,j]$ as specified above; $\mathcal{H}$ denotes the set of half-edges of $\mathcal{M}$. We define a corresponding Laplacian matrix for our problem by taking into account that each matrix $\mathbf{W}_i$ contains three unknowns, and that a change of basis is necessary to compare the matrices of adjacent triangles. The matrix $\mathbf{W}_j$ expressed in the same basis of $\mathbf{W}_i$ becomes $\mathbf{W}_{ij} = \mathbf{R}_{ij}\mathbf{W}_j\mathbf{R}_{ij}^T$, where

$$\begin{aligned} w_{ija} &= r_{ija}^2 w_{ja} + 2r_{ija}r_{ijb}w_{jb} + r_{ijb}^2 w_{jc}\,, \\ w_{ijb} &= r_{ija}r_{ijc}w_{ja} + \\ &\qquad +(r_{ijb}r_{ijc} + r_{ija}r_{ijd})w_{jb} + r_{ijb}r_{ijd}w_{jc}\,, \\ w_{ijc} &= r_{ijc}^2 w_{ja} + 2r_{ijc}r_{ijd}w_{jb} + r_{ijd}^2 w_{jc}\,. \end{aligned}$$

Thus we want a matrix operator that computes the differences between w_{ia} and w_{ija}, w_{ib} and w_{ijb}, and w_{ic} and w_{ijc}. We represent each $\mathbf{W}_i$ by vector $\mathbf{w}_i = [w_{ia}, w_{ib}, w_{ic}]^T$ and we build a $3n \times 3n$ matrix $\mathbf{L^B}$ defined as

$$\mathbf{L^B} = \sum_{(i,j)\in\mathcal{H}} \mathbf{L}^{\mathbf{B}}_{ija} + \mathbf{L}^{\mathbf{B}}_{ijb} + \mathbf{L}^{\mathbf{B}}_{ijc} \tag{11}$$

where

$$\begin{aligned} &\mathbf{L}^{\mathbf{B}}_{ija}[3i,3i] = -1\,, && \mathbf{L}^{\mathbf{B}}_{ija}[3i,3j] = r_{ija}^2\,, \\ &\mathbf{L}^{\mathbf{B}}_{ija}[3i,3j+1] = 2r_{ija}r_{ijb}\,, && \mathbf{L}^{\mathbf{B}}_{ija}[3i,3j+2] = r_{ijb}^2\,, \\ &\mathbf{L}^{\mathbf{B}}_{ijb}[3i+1,3i+1] = -1\,, && \mathbf{L}^{\mathbf{B}}_{ijb}[3i+1,3j] = r_{ija}r_{ijc}\,, \\ &\mathbf{L}^{\mathbf{B}}_{ijb}[3i+1,3j+1] = r_{ijb}r_{ijc} + r_{ija}r_{ijd}\,, \\ &\mathbf{L}^{\mathbf{B}}_{ijb}[3i+1,3j+2] = r_{ijb}r_{ijd}\,, \\ &\mathbf{L}^{\mathbf{B}}_{ijc}[3i+2,3i+2] = -1\,, && \mathbf{L}^{\mathbf{B}}_{ijc}[3i+2,3j] = r_{ijc}^2\,, \\ &\mathbf{L}^{\mathbf{B}}_{ijc}[3i+2,3j+1] = 2r_{ijc}r_{ijd}\,, \\ &\mathbf{L}^{\mathbf{B}}_{ijc}[3i+2,3j+2] = r_{ijd}^2\,, \end{aligned}$$

and all other entries of the matrices are zero. Here we assume the indices of matrix entries start at zero.

B Proof of Lemma 5.1

We provide the proof in the plane. On a generic mesh $\mathcal{M}$ the proof is similar, but it involves further technicalities related to changes of local coordinate systems, which make the Laplace equation not separable for the three components of each $\mathbf{w}_i$.

In the plane, we can express all matrices $\mathbf{W}_i$ in a unique basis, so no changes of coordinates are necessary. In this case, we can rewrite Eq. (4) as three equations

$$\mathbf{L}^{\mathcal{M}}\mathbf{w}_a = 0, \quad \mathbf{L}^{\mathcal{M}}\mathbf{w}_b = 0, \quad \mathbf{L}^{\mathcal{M}}\mathbf{w}_c = 0 \tag{12}$$

subject to constraints, respectively,

$$w_{ja} = \hat{w}_{ja}, \quad w_{jb} = \hat{w}_{jb}, \quad w_{jc} = \hat{w}_{jc}, \quad j \in \mathcal{C}, \tag{13}$$

where $\mathbf{w}_a, \mathbf{w}_b, \mathbf{w}_c$ collect the w_{ia}, w_{ib}, w_{ic} coefficients of the $\mathbf{W}_i$ matrices, respectively, while $\mathbf{L}^{\mathcal{M}}$ is a standard $n \times n$ uniform Laplacian matrix depending just on the connectivity of $\mathcal{M}$. The solution of each constrained equation can be obtained in a standard way by removing the rows and columns of $\mathbf{L}^{\mathcal{M}}$ corresponding to the constraints, and then building a right-hand term that depends only on the constraints. Therefore, each constrained equation becomes a linear system of the type $\mathbf{L}_f^{\mathcal{M}}\mathbf{w}_z = \mathbf{b}_z$, for $z = a, b, c$, where $\mathbf{L}_f^{\mathcal{M}}$ is the Laplacian matrix reduced to the $n-k$ free terms of the Laplace equation and $\mathbf{b}_z$ is a linear combination of the $\hat{w}_{jz}$ coefficients. Since the matrix $\mathbf{L}_f^{\mathcal{M}}$ is the same in all three systems, the solution is of the type

$$\mathbf{W}_i = \sum_{j\in\mathcal{C}} \alpha_{ij}\hat{\mathbf{W}}_j, \tag{14}$$

where the α_{ij} coefficients depend only on $L_f^{\mathcal{M}}$. Since we solve a discrete Laplace equation with positive weights, the solution must satisfy the discrete maximum principle [Wardetzky et al. 2007], i.e.,

$$\min_{j\in\mathcal{C}} \hat{w}_{jz} \le w_{iz} \le \max_{j\in\mathcal{C}} \hat{w}_{jz}. \tag{15}$$

This implies that all the α_{ij} coefficients are non-negative and they cannot be all zero (actually, it can be proven that they sum to 1): if for a given j we had $\alpha_{ij} < 0$, then by setting the constraint $\hat{w}_{jz} > 0$ and all other $\hat{w}_{hz} = 0$, we would obtain $w_{iz} < 0$, which violates the maximum principle; similarly, if all α_{ij} were zero, the maximum principle would be violated by setting all $\hat{w}_{jz}$ strictly positive.

To conclude, let $\mathbf{v}$ be a non-zero vector. We have

$$\mathbf{v}^T\mathbf{W}_i\mathbf{v} = \mathbf{v}^T\left(\sum_{j\in\mathcal{C}} \alpha_{ij}\hat{\mathbf{W}}_j\right)\mathbf{v} = \sum_{j\in\mathcal{C}} \alpha_{ij}\mathbf{v}^T\hat{\mathbf{W}}_j\mathbf{v}, \tag{16}$$

and we know that all $\hat{\mathbf{W}}_j$ are SPD, so we must have $\mathbf{v}^T\hat{\mathbf{W}}_j\mathbf{v} > 0$ for all j, and since all α_{ij} are non-negative and at least one of them is positive, we must have $\mathbf{v}^T\mathbf{W}_i\mathbf{v} > 0$. □

C Gradient of $\mathcal{E}$

Following [Pinkall et al. 1993; Liu et al. 2008], the energy in Eq. (5) can be rewritten in terms of the half-edges of $\mathcal{M}$ as follows:

$$\mathcal{E} = \sum_{(i,j)\in\mathcal{H}} \min_{\mathbf{Q}_{t(i,j)}\in SO(3)} \cot\theta_{ij}\|(p'_i - p'_j) - \mathbf{Q}_{t(i,j)}\tilde{\mathbf{W}}^{-1}_{t(i,j)}(p_i - p_j)\|^2.$$

If we fix the rotations $\mathbf{Q}$, the partial derivative of $\mathcal{E}$ w.r.t. p'_i is

$$\begin{aligned} \frac{\partial\mathcal{E}}{\partial p'_i} &= 2\sum_{j\in\mathcal{N}(i)} (\cot\theta_{ij} + \cot\theta_{ji})(p'_i - p'_j) - \\ &- \left(\cot\theta_{ij}\mathbf{Q}_{t(i,j)}\tilde{\mathbf{W}}^{-1}_{t(i,j)} + \cot\theta_{ji}\mathbf{Q}_{t(j,i)}\tilde{\mathbf{W}}^{-1}_{t(j,i)}\right)(p_i - p_j) \end{aligned} \tag{17}$$

which can be rewritten in terms of the standard vertex-based cotan Laplacian matrix $\mathbf{L}$ of $\mathcal{M}$ as

$$\sum_{j\in\mathcal{N}(i)} 2\left(\cot\theta_{ij}\mathbf{Q}_{t(i,j)}\tilde{\mathbf{W}}^{-1}_{t(i,j)} + \cot\theta_{ji}\mathbf{Q}_{t(j,i)}\tilde{\mathbf{W}}^{-1}_{t(j,i)}\right)(p_j - p_i) \;-\; 4(\mathbf{L}\mathbf{p}')_i.$$

By substituting with $\mathbf{b}_i$ (Eq. (7)) in the above equation and collecting all terms, we obtain the gradient in Eq. (6).