Functional Characterization of Deformation Fields

Etienne Corman, Maks Ovsjanikov

Functional Characterization of Deformation Fields

ETIENNE CORMAN, University of Toronto, Canada
MAKS OVSJANIKOV, LIX, École Polytechnique, CNRS, France

In this paper we present a novel representation for deformation fields of 3D shapes, by considering the induced changes in the underlying metric. In particular, our approach allows to represent a deformation field in a coordinate-free way as a linear operator acting on real-valued functions defined on the shape. Such a representation both provides a way to relate deformation fields to other classical functional operators and enables analysis and processing of deformation fields using standard linear-algebraic tools. This opens the door to a wide variety of applications such as explicitly adding extrinsic information into the computation of functional maps, intrinsic shape symmetrization, joint deformation design through precise control of metric distortion, and coordinate-free deformation transfer without requiring pointwise correspondences. Our method is applicable to both surface and volumetric shape representations and we guarantee the equivalence between the operator-based and standard deformation field representation under mild genericity conditions in the discrete setting. We demonstrate the utility of our approach by comparing it with existing techniques and show how our representation provides a powerful toolbox for a wide variety of challenging problems.

CCS Concepts: • **Mathematics of computing → Discretization**; • **Computing methodologies → Computer graphics**; **Shape analysis**;

Additional Key Words and Phrases: Shape exploration, functional maps

ACM Reference Format:
Etienne Corman and Maks Ovsjanikov. 2018. Functional Characterization of Deformation Fields. *ACM Trans. Graph.* 1, 1 (November 2018), 19 pages. https://doi.org/0000001.0000001_2

Fig. 1. Our deformation field representation is useful to generate symmetrically consistent quad-meshes from a triangle mesh. The standard quad-meshing technique (left) results in a mesh misaligned with the symmetric features. Instead we propose an intrinsic symmetrization approach (middle, see also the accompanying video), which leads to a more symmetric shape, from which we compute the quad mesh and pull it back to the initial shape (right), creating an anisotropic quad-mesh better aligned with the symmetry.

1 INTRODUCTION

Designing and analyzing shape deformations is a central problem in computer graphics and geometry processing, with applications in scenarios such as shape manipulation [47, 59], animation and deformation transfer [49], shape interpolation [24, 56], and even anisotropic meshing [39] among myriad others. Traditionally, shape deformation has been motivated by interactive applications in which the main goal is to design a deformation that satisfies some user-prescribed handle constraints while preserving the main structural properties of the shape. In other applications, such as shape interpolation and deformation transfer, that lack handle constraints, the goal is to design a global deformation field that would satisfy some structural properties as well as possible.

In both types of applications, most approaches are based on specifying a deformation energy and providing a method to optimize it. On the other hand, several works have demonstrated that by choosing an appropriate *representation* for shape deformations, many tasks can become significantly easier, and in particular can help to enforce certain properties of the deformation field, which are otherwise very difficult to access and optimize for. In addition to the classical per-vertex displacement vectors, such representations have included gradient-based deformations [59, 60], Laplacian-based approaches [29, 48] and Möbius transformations in the context of conformal deformations [15, 54] among others.

At the same time, a number of recent works have shown that many basic operations in geometry processing can be viewed as linear operators acting on real-valued functions defined on the shapes. This includes the functional representation of mappings or correspondences acting through composition [36, 42], representations of vector fields as derivations [2, 41] and formulation of shape distortion via shape difference operators [44]. One advantage of these representations is that linear operators can be naturally *composed,* which makes it easy to define, for example, the push-forward of a vector field with respect to a mapping, if both are represented as linear operators, or to solve for Killing vector fields, by composition between a derivation and the Laplacian operator. Moreover, by using a consistent functional representation these techniques often alleviate the need for point-wise correspondences, which can be difficult to obtain, as shown very recently for example in a work on joint cross-field design [3].

While tangent vector fields are classically understood as operators (derivations) in differential geometry, extrinsic vector fields do not enjoy a similar property. Our main goal is to provide a coordinate-free representation of extrinsic vector fields (that we also call deformation fields) as functional operators, which will prove useful for analysis and design of shape deformations. As we demonstrate below our representation greatly simplifies certain tasks such as

Parts of this work were supported by a Competitive Research Grant CRG6 2017 3426 from KAUST, a Google Focused Research Award and the ERC Starting Grant No. 758800 (EXPROTEA).
Authors' addresses: Etienne Corman, University of Toronto, 40 St. George Street, ON, Toronto, M5S 2E4, Canada; Maks Ovsjanikov, LIX, École Polytechnique, CNRS, Route de Saclay, Palaiseau, 91128, France.

intrinsic symmetrization, the computation of mappings by composition with other operators, and joint deformation design without requiring point-wise mappings. Moreover, it provides an explicit link between deformation fields and the changes in intrinsic metric quantities, which can be useful in a variety of analysis and deformation processing tasks. For example, our intrinsic shape symmetrization built on our representation of deformation fields, allows to deform a given shape to make it more intrinsically symmetric. This can be useful, for example, to compute more symmetric quad meshes, starting from a triangle mesh, by first deforming it, remeshing the deformed shape and pulling back the quad mesh onto the original shape, as shown in Fig. 1 and Figures 11, 12, and 13.

2 RELATED WORK

Shape deformation is one of the oldest and best-researched topics in computer graphics and geometry processing. We therefore only mention works most directly related to ours and refer the interested reader to surveys including [11, 34] and [10] (Chapter 9).

A multitude of methods exists for surface deformation starting with the seminal work of [50], its early follow-ups including [13, 57] and the multi-scale variants, such as [22, 26, 63] among many others. Similarly to our approach, many of these techniques are based on optimizing the so-called elastic thin shell energy that measures stretching and bending, and which is often linearized for efficiency. In the majority of cases, deformations are represented explicitly as extrinsic vector fields defined on a surface, making deformation transfer difficult in the absence of precise pointwise correspondences.

A number of methods have proposed alternative representations for deformation fields, which greatly simplify certain tasks in design and analysis. This includes gradient-based techniques [59, 60] which consider the deformation field by aligning its gradient with a set of local per-triangle transformations. By working in gradient space, constraints can be posed independently on the triangles and then optimized globally by solving the Poisson equation. Similarly, Laplacian-based techniques [29, 35, 48] are based on defining shape deformations by manipulating per-vertex differential coordinates (Laplacians) in order to match some target Laplacian coordinates. Such differential coordinates enable direct editing of local shape properties, which can be especially beneficial for preserving and manipulating the high-frequency details of the surface. However, these coordinates are typically not rotationally invariant and additional steps are necessary to introduce invariance [29, 40, 48].

More recently, a number of methods have introduced representations for mesh deformations specifically geared towards particular shape manipulations, such as computing conformal transformations by designing special maps into the space of quaternions [15] or by using face-based compatible Möbius transformations [54]. These techniques are rotationally invariant and coordinate-free, while being restricted to special types of manipulations. Another technique, closely related to ours, designs shape deformations by constructing a continuous divergence-free vector field [55], and applying path line integration to obtain a deformed shape. We also consider the effect of the deformation on the metric, but both analyze the distortion of arbitrary extrinsic vector fields and show how they can be represented in coordinate-free way as linear functional operators.

Our use of spectral techniques and functional maps for representing deformation fields is also related to previous works in spectral shape processing, including the early approaches of Lévy and colleagues and their extensions [16, 28, 53] and more recent techniques such those based on coupled quasi-harmonic bases and functional maps [27, 58]. In these and related methods deformation fields are represented as triplets of functions, which encode displacement in each spatial coordinate. Although this representation is simple and naturally fits with the functional map framework, it suffers from several drawbacks. First, it is not rotationally invariant and induces artefacts if the shapes are not pre-aligned or are in different poses (see e.g., Figure 1). Perhaps more fundamentally, such a representation is not "shape-aware" since it does not reflect the change in the (e.g., metric) structure of the shapes induced by the deformation, which reduces its utility in deformation analysis and design. We demonstrate through extensive experiments, that by using our coordinate-free representation we can avoid these limitations and open the door to entirely novel design and analysis applications, such as intrinsic symmetrization (Section 8.2), which cannot be achieved using previous methods.

Our approach of considering the deformation via its induced metric distortion is also related to the work of [20] and [45] who manipulate shapes by explicitly editing their curvature properties. Moreover, our use of the strain tensor in characterizing metric distortion is closely related to the applications in various physically based deformation scenarios including [33, 51] among many others (see also the surveys on physically based elastic deformable models [34, 43]). Our approach is also related to the works that aim to design as-isometric-as-possible shape deformations [32, 46, 61]. Similarly to the latter work, our framework is general and allows an arbitrary prescribed distortion, although our method works directly on surface representations and moreover enables applications such as joint deformation design.

Finally, our framework for joint design is related to the deformation transfer and interpolation techniques such as [5, 49] and [24] to name a few. Our approach is different in that we place special emphasis on relating deformations between shapes with only soft (or functional) correspondences, which are often much easier to obtain than detailed point matches. Moroever, rather than transporting Jacobian matrices associated with the deformation, which requires both a pre-alignment and an approximate triangle-to-triangle map (as done in [49]) we study and transport the change in the intrinsic metric structure directly. As we show below, this results in better joint deformation design especially given approximate functional maps, and shapes in arbitrary poses.

Thus, in contrast to the majority of existing techniques our goal is to devise a coordinate-free representation of extrinsic deformations as linear functional operators, by making an explicit connection between the extrinsic deformations and the change in intrinsic metric quantities. As such, our representation fits within the recent line of work that represents many operations in geometry processing as functional operators, including mappings or correspondences [36, 42], representations of vector fields as derivations [2, 41] and

the formulation of shape distortion via shape difference operators [44]. Therefore, although we build on classical constructions such as the infinitesimal strain tensor, we show how they can be exploited to create a functional representation of shape deformation, which can be used in conjunction with other operators. As we demonstrate below, our representation is particularly useful for analysing and manipulating the effect of the deformation on the shape structure and for relating deformations across shapes, with only soft correspondences between them. In particular, it enables applications such as intrinsic symmetrization, joint deformation design and allows to introduce extrinsic information in the computation of functional maps. Remarkably, we prove that together with the classical Laplace-Beltrami operator, our approach leads to a *complete* (up to rigid motion) *coordinate-free functional shape representation*, which opens the door to new shape processing applications.

3 OVERVIEW

The rest of the paper is organized as follows: first, we define the functional deformation field representation using the classical notions of the Levi-Civita connection and the strain tensor, and list the main properties of this representation (Section 4). We then provide a link between this definition and the previously proposed shape difference operators, by considering their infinitesimal extensions, introducing a new unified operator, and proving the equivalence between the two definitions (Section 5). In Sections 6 and 7 we provide a discretization of all of these notions, and show that they preserve the main properties of the continous counterparts. Finally, we illustrate the utility of our representation by describing several novel application scenarios, which range from functional map inference, to intrinsic symmetrization and deformation field design that all exploit the properties of our representation and its relation to other previously proposed linear operators (Section 8). Note that Sections 5 and 7 can be skipped by readers that are not interested in the connection to shape difference operators.

To summarize, our main contributions include:

- Introducing *functional deformation fields* as a way to represent extrinsic vector fields in a coordinate-free way as operators acting on functions, represented as matrices in the discrete setting.
- Providing a link between functional deformation fields and the previously proposed shape difference operators, which leads to both a new unified shape difference and alternative functional deformation fields, which can be made sensitive to specific (e.g., non-conformal) classes of distortions.
- Showing how functional deformation can be used to add extrinsic information (second fundamental form) into the computation and analysis of functional maps. We also prove that together with the Laplace-Beltrami operator, they provide a *complete* coordinate-free shape characterization up to rigid motions.
- Describing how this representation enables a number of novel applications including intrinsic shape symmetrization, useful for symmetric quad remeshing, deformation design and functional deformation transfer without pointwise correspondences.

4 EXTRINSIC VECTOR FIELDS AS OPERATORS

In this section we provide a coordinate-free representation of extrinsic vector fields by considering their action on the underlying shape metric. Throughout this section we assume that we are dealing with a smooth surface M without boundary embedded in $\mathbb{R}^3$. The appropriate discretization of all the concepts introduced in this section will be given in Section 6.

The Levi-Civita Covariant Derivative. We first need to introduce some fundamental notions from differential geometry. In particular, we will use the classical Levi-Civita connection to define derivatives on a surface. More precisely, given a *tangent vector* u at some point $p \in M$, and an extrinsic vector field V on M, consider an arbitrary curve $\gamma(t)$ on M such that $\gamma(0) = p$ and $\gamma'(0) = u$. Then, we let $\bar{\nabla}_u V = \left.\frac{\partial V(\gamma(t))}{\partial t}\right|_{t=0}$. Here $\bar{\nabla}_u V$ is the standard covariant derivative of the ambient space. Note that at a fixed point $p \in M$, $\bar{\nabla}_u V$ is a vector in $\mathbb{R}^3$. We can project the covariant derivative onto the tangent plane at p to obtain a vector in the tangent plane, which is denoted simply by $\nabla_u V$ where ∇ is the Levi-Civita connection on M extended naturally to extrinsic vector fields, ([17] p. 126). We also remark that for any vector x in the tangent space, $\langle \nabla_u V, x\rangle = \langle \bar{\nabla}_u V, x\rangle$, which we will use in our discretization.

The fundamental object that we consider below is the infinitesimal strain tensor, which can be understood as a bilinear form, acting on pairs of vectors x, y in the tangent plane of a point $p \in M$. Namely, given an extrinsic vector field V, the infinitesimal strain tensor $\mathcal{L}_V \mathbf{g}(x, y)$ is defined as:

$$\mathcal{L}_V \mathbf{g}(x, y) = \langle x, \nabla_y V\rangle + \langle \nabla_x V, y\rangle \tag{1}$$

The infinitesimal strain tensor $\mathcal{L}_V \mathbf{g}$ is also called the Lie derivative of the metric. It has the advantage of being linear in the vector field V, which makes it easy to handle for deformation and vector field design and therefore has been used in a wide variety of works in computer graphics [34].

Physically, it represents the infinitesimal stretch that the object undergoes at each point. Thus, the eigenvector associated to the largest eigenvalue of $\mathcal{L}_V \mathbf{g}$ (which can be thought of simply as a symmetric 2x2 matrix) at a point p, corresponds to the tangent vector x that represents the local direction of maximal stretch.

With these definitions in hand we propose to consider a linear functional operator E^V, which we will use to capture and manipulate a deformation field V. Both the input and the output of our operator are smooth real-valued functions defined on the surface. This operator is defined implicitly, in the same spirit as the shape difference operators introduced by Rustamov et al. [44] as follows: for every *pair* of real-valued functions f, g we require:

$$\int_M \langle \nabla g, \nabla E^V(f)\rangle \mathrm{d}\mu = \int_M \mathcal{L}_V \mathbf{g}(\nabla g, \nabla f)\mathrm{d}\mu. \tag{2}$$

The following proposition guarantees that E^V is well-defined.

Proposition 4.1. *For any extrinsic vector field V there is a unique linear functional operator E^V that satisfies Eq. (2) above. Moreover, this operator is linear in both the vector field V and function f.*

In the rest of the paper we call the linear functional operator E^V, a *functional deformation field representation of V*. Our main goal is

to design, manipulate and analyze extrinsic vector fields V through their associated linear functional operators E^V. This approach has already proved useful in the context of manipulating maps or correspondences [36], tangent vector fields [2] and shape distortions [44]. In particular, these works have helped to establish a general formalism of shape manipulation through the associated linear functional operators, which can "communicate" by composition. This allows, for example, to transfer tangent vector fields across shapes without assuming pointwise correspondences [2] or to design very efficient shape matching algorithms using the functional map representation [37]. Therefore, inspired by these works, we propose to extend this framework to also include *extrinsic* (or deformation) fields. As we show below, our representation naturally fits within the general functional operator formalism and enables a number of novel applications.

4.1 Key Properties of Functional Deformation Fields

Second-fundamental form representation. One interesting special case to consider is the interpretation of E^V when the deformation field is the normal field $V = n$. By using Eq. (1) it is possible to see ([17] p.128) that the covariant derivative of the normal yields the second fundamental form denoted by $\mathbf{h}_p : T_pM \times T_pM \to \mathbb{R}$, more precisely $\mathcal{L}_n \mathbf{g} = -2\mathbf{h}$. Therefore the operator E^n captures the action of curvature on functions, since:

$$\int_M \langle \nabla f, \nabla E^n(g) \rangle \mathrm{d}\mu = -2 \int_M \mathbf{h}(\nabla f, \nabla g) \mathrm{d}\mu.$$

From a theoretical point of view the knowledge of the Laplace-Beltrami operator gives access to the first fundamental form and E^n yields information about the second. Thus these two operators jointly provide a coordinate-free representation of the embedding.

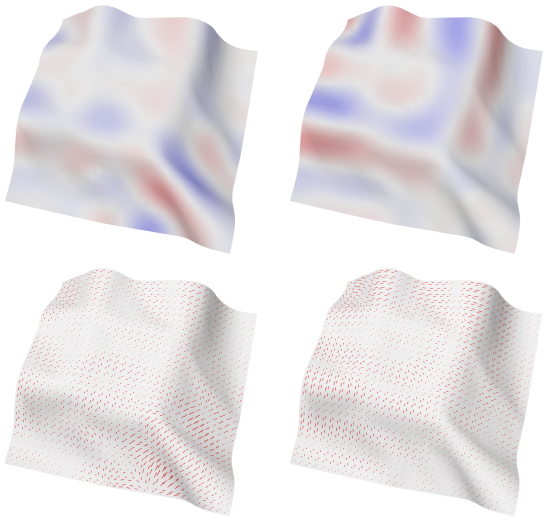
The operator E^n can be used to obtain a multi-scale representation of curvature information on the triangle mesh, as shown in Figure 2. In particular, the eigenfunctions corresponding to the largest eigenvalues of E^n, are those that align the best with the maximal principal curvature direction, and can be obtained even if E^n is represented in a reduced functional basis, making the computation less sensitive to noise in the triangulation. Moreover, as we demonstrate in Section 8.1, the operator E^n can be used to inject extrinsic information into the computation of functional maps.

We remark that the connection between the metric distortion along the normal fields and surface properties has been used heavily in Discrete Differential Geometry, in particular to establish a curvature theory for discrete surfaces, e.g. in [6] among others. These and related works exploit the relation between curvature and changes in area, given by the classical Steiner formulas to discretize curvatures on general polygonal meshes. Unlike such approaches, we concentrate on *representing an arbitrary deformation field* through its induced metric distortion, and consider the full Lie derivative of the metric (rather than, for example, only considering area changes), in the case of triangle meshes. Nevertheless, we leave the extension of our construction to arbitrary polygonal meshes and expanding the connection with recent results in this area, including [23] as interesting future work.

Composition with mappings. In many applications we are interested in the relation between deformations on multiple surfaces related by a mapping. In particular given a deformation field U_M of shape M and a diffeomorphism $\varphi : M \to N$ with the associated functional map (pullback) C_φ of functions from N to M, one can define a deformation field V_N of shape N that produces the same metric distortion. Instead of looking directly at the deformation of the metric, which might require a mapping between individual triangles [49], we account for the action of the metric on functions:

$$E^{U_M} C_\varphi(f) = C_\varphi E^{V_N}(f) \; \forall f \in C^\infty(N)$$

In other words, V_N can be obtained by considering an extrinsic vector field, whose operator representation has the same effect on functions when composed with the functional map C_φ as E^{U_M}. This property allows us to relate deformation fields without requiring point-to-point correspondences between shapes, by simply considering the commutativity of the operators C_φ and E. We illustrate this in Figure 5 and use it in Section 8.5 for deformation transfer and deformation symmetrization on meshes with different connectivities with only a functional map known between them. Furthermore, this approach is applicable to design deformations jointly on two shapes, such that they are consistent with the functional map C_φ and even as a regularizer in map computation.



1st eigenfunction 2nd eigenfunction

Fig. 2. Two eigenfunctions associated with the largest eigenvalues of the functional deformation E^n for the normal field n. The gradients of these functions represent the direction of maximal curvature (bottom row).

Vector field representation. In general the operator E^V does not uniquely define an extrinsic vector field. From Def. 2 it can be shown that the kernel of $V \mapsto E^V$ coincides with the vector fields satisfying $\mathcal{L}_V \mathbf{g} = 0$. In case of a volumetric manifold (i.e. $M \subset \mathbb{R}^3$) the kernel of our operator is restricted to infinitesimal rigid motions (see Theorem 1.7-3 in [14]) and thus provides a complete representation of extrinsic vector fields. In the case of a surface embedded in $\mathbb{R}^3$ the kernel of E^V includes infinitesimal isometries such as Killing vector fields but also local normal fields in planar areas. No rigidity result seems to be known for smooth surfaces. However, as we demonstrate below, in the discrete case of shapes represented as triangle meshes, it can be shown that for almost all surfaces the

kernel of $V \mapsto E^V$ consists only of rigid deformations (Prop. 6.2). Note that we place *no restriction on the magnitude of the deformation fields*. Thus, although our construction is based on the infinitesimal strain tensor, the extrinsic vector fields themselves are not limited to infinitesimal (or local) deformations. Finally, as we show below, our constructions can be extended to tetrahedral meshes, resulting in a *complete* operator-based representation for deformation fields of *volumes*, not sensitive to the exceptional cases, present in the case of surfaces. Moreover, in the case of volumetric meshes, the tangent space at every tet is naturally identified with the entire ambient 3D space. This means that unlike the case of surfaces, no special treatment is necessary for the boundary.

5 RELATION TO SHAPE DIFFERENCES

The functional deformation field representation introduced above is closely related to the previously proposed shape difference operators. In this section we describe this relation in detail, and highlight the following two key insights: 1) How our analysis leads to a novel *unified* shape difference operator, and 2) How alternative functional deformation field representations can be constructed, to be sensitive to only a particular class of metric distortions. Our analysis also sheds light on the discretization of functional deformation fields. Nevertheless, the discussion in this section is not required for the understanding of either the implementation or the results of our approach, apart from the intrinsic symmetrization application (Sec. 8.2), in which we use this relation. As such, this section can be skipped by readers not interested in these relations.

5.1 Shape Difference Operators

Introduced by [44], the shape difference operators describe a shape deformation by considering the change of inner products between functions. Namely, given a pair of shapes M, N and a diffeomorphism $\varphi : N \to M$, with the associated linear functional map (pullback) defined by $C_\varphi(f) = f \circ \varphi$, the authors introduce the area-based and conformal shape difference operators D_A and D_C respectively, as linear operators acting on (and producing) real-valued functions on M implicitly via the following equations:

$$\langle f, D_A(g)\rangle_{L^2(M)} := \langle C_\varphi(f), C_\varphi(g)\rangle_{L^2(N)} \qquad \forall f, g \qquad (3)$$

$$\langle f, D_C(g)\rangle_{H^1_0(M)} := \langle C_\varphi(f), C_\varphi(g)\rangle_{H^1_0(N)} \qquad \forall f, g \qquad (4)$$

where the inner products are defined as $\langle f, g\rangle_{L^2(M)} := \int_M fg\mathrm{d}\mu$ and $\langle f, g\rangle_{H^1_0(M)} := \int_M \langle \nabla f, \nabla g\rangle \mathrm{d}\mu$.

The existence and the linearity of the operators D_A and D_C is guaranteed by the Riesz representation theorem. As shown in [44], for smooth surfaces, the map φ is area-preserving (resp. conformal) if and only if D_A (resp. D_C) is the identity map between functions. From this it follows that φ is an isometry if and only if D_A and D_C are *both* identity.

Note that in the discrete setting the shape difference operators are obtained simply by considering transposes and inverses of the functional map and Laplacian matrices, as highlighted in [44]. This makes properties such as existence and linearity trivial to see. Below we adopt the continuous (surface) formulation proposed in the original article as it helps to highlight both the generality of these concepts and also the relation to our representation of extrinsic vector fields.

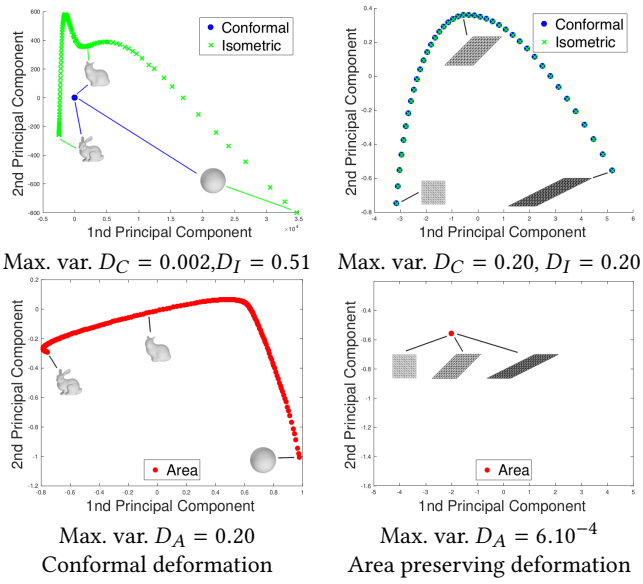


Fig. 3. Left: Approximately conformal deformation of a bunny to a sphere. The PCA applied to shape differences confirms the presence of large area (bottom) and isometric distortion in contrast to small conformal distortion (top). Right: Area-preserving deformation of a plane, which results in nearly constant area-based shape differences, unlike the conformal and isometric differences, which agree. We also report the maximal singular value of the PCA covariance matrices of the scaled operators in each case, for quantitative comparison of these results.

Infinitesimal Shape Difference Operators. Our main goal in this section is to consider a one-parameter family of shapes M_t, given by displacing the points of a base shape along some fixed deformation field. Specifically, given a surface M embedded in $\mathbb{R}^3$ we consider a family M_t, parameterized by a scalar t and given by $p_t = p_0 + tV(p_0)$, where p_0 is a fixed point in $\mathbb{R}^3$, and $V(p)$ is a vector in $\mathbb{R}^3$ that represents the displacement of the point.

Now consider the family of maps $\varphi_t : M \to M_t$, given trivially via $\varphi_t(p) = p_t$, and the associated functional maps $C_{\varphi_t^{-1}}$ mapping functions from M_0 to M_t. This gives rise to a one-parameter family of shape difference operators D^V_t (which can be taken either to be the area or conformal-based operators). We then introduce *the infinitesimal shape difference operator* as follows:

Definition 5.1. The *infinitesimal area-based shape difference operator* associated with an extrinsic vector field V on a surface M is defined as:

$$E^V_A := \left.\frac{\partial D^V_{At}}{\partial t}\right|_{t=0}, \qquad (5)$$

We define the *infinitesimal conformal shape difference operator* E^V_C similarly by replacing D^V_A by D^V_C on the right side of Eq. (5).

Remark that since both E^V_A and E^V_C are defined as derivatives of a one-parameter family of linear operators acting on real-valued functions on a surface, both the range and the domain of these

operators are also real-valued functions on M. Moreover, as D_A^V and D_C^V reflect (or, equivalently, are sensitive to) changes in the area and conformal metric structure, this implies that E_A^V and E_C^V will only reflect extrinsic vector fields up to infinitesimally area-preserving or conformal deformations. This naturally raises the question of whether there exists another "unified" shape difference operator D_I, which would be sensitive to general (non-isometric) metric changes. If so, would such D_I lead to an infinitesimal shape difference E_I that would agree with the definition of E^V given in Eq. (2)? Below, we provide precisely such a definition which both extends the applicability of shape difference operators and helps to establish a deeper link with our functional deformation fields.

Unified shape difference. The main reason for which D_C is only sensitive to conformal changes is that both the inner product and the integration are taken on the target shape. To define a unified shape difference taking into account all intrinsic changes one should compare the pullback metric to the metric on M while keeping the integrating measure fixed. We thus propose a unified shape difference operator D_I that fully characterizes isometric distortion.

Definition 5.2. Assuming that $\varphi : N \to M$ is a diffeomorphism, the unified shape difference $D_I : C^\infty(M) \to C^\infty(M)$ is defined implicitly by:

$$\langle f, D_I(g)\rangle_{H_0^1(M)} := \int_M C_{\varphi^{-1}} \left(\langle \nabla C_\varphi(f), \nabla C_\varphi(g)\rangle\right) \mathrm{d}\mu^M.$$

The existence of D_I is once again guaranteed by the Riesz representation theorem. Moreover, as we claimed above, the following proposition (proved in the supplemental material) shows that the unified shape difference fully characterizes isometric deformation.

Proposition 5.3. *$D_I(f) = f$ for all $f \in C^\infty(M)$ if and only if φ is an isometry.*

To illustrate the properties of the three shape differences we use a simple low-dimensional description of a shape collection in Figure 3. Here we choose a fixed base shape and compute the shape difference matrices with respect to the remaining shapes in a collection. Then, we represent each shape by its shape difference matrix and plot them as points in PCA space. We show the unified and conformal shape differences in the same plots to stress that they are derived from the same principles and are equal for any area-preserving deformations, and thus are expressed in the same units. Figure 3 represents the conformal deformation of a bunny into a sphere as viewed by the three shape differences. As expected D_C is almost identity while the area and isometric shape differences both capture the distortion. In the second experiment, shown in Figure 3, we explore another collection obtained by the shearing of a plane patch. As this deformation is area preserving, the area-based shape difference provides no information, unlike the other two operators which are guaranteed to be equal in this case. Finally, we also report the largest singular values of the PCA covariance matrices in each case, after vectorizing and pre-scaling the operators to unit L_2 norm to make the results comparable. These maximal variance values provide a quantitative validation of the same observations.

With the definition of the unified shape difference D_I in hand, we introduce its infinitesimal counterpart E_I by following the same construction as done in (5) above. The following proposition (proved in the supplemental material) characterizes these new operators.

Proposition 5.4. *Let V be a smooth deformation field on M, the derivatives of D_A, D_C and D_I at time zero satisfy for all smooth functions f, g:*

$$\langle f, E_A^V(g)\rangle_{L^2(M)} = \int_M \mathrm{div}(V) f g \mathrm{d}\mu,$$
$$\langle f, E_C^V(g)\rangle_{H_0^1(M)} = \int_M \mathrm{div}(V)\langle \nabla f, \nabla g\rangle - \mathcal{L}_V \mathbf{g}(\nabla f, \nabla g)\mathrm{d}\mu,$$
$$\langle f, E_I^V(g)\rangle_{H_0^1(M)} = -\int_M \mathcal{L}_V \mathbf{g}(\nabla f, \nabla g)\mathrm{d}\mu.$$

As can be seen, the infinitesimal shape differences inherit the properties of the original operators. Namely, E_A^V vanishes if and only if $\mathrm{div}(V)$ is equal to zero, i.e., whenever V infinitesimally preserves the volume form. On the conformal side, finding an extrinsic vector field V such that $E_C^V = 0$ is equivalent to solving the *conformal Killing equation*: $\mathcal{L}_V \mathbf{g} = \mathrm{div}(V)\mathbf{g}$ characteristic of a conformal vector field. Both properties combined lead to an isometric deformation induced by the vector field V captured by E_I.

Moreover Prop. 5.4 reveals a clear link between shape differences:

$$\langle f, E_I^V(g)\rangle_{H_0^1} = \langle f, E_C^V(g)\rangle_{H_0^1} - \langle 1, E_A^V(\langle \nabla f, \nabla g\rangle)\rangle_{L^2}. \quad (6)$$

Thus, intuitively, the operator E_I, representing isometric distortion, can be decomposed into an area and a conformal part. We note that linear dependence between shape operators shown in Eq. (6) can be understood as the decomposition of the matrix $\mathcal{L}_V \mathbf{g}$ into a trace free part, linked to the conformal Killing equation, and a divergence part, related to the change in area.

Finally, this proposition shows that the functional deformation field representation introduced in Section 4 is exactly the same as the infinitesimal shape difference operator E_I^V arising from the unified shape difference. Remarkably, this relation also holds exactly in the discrete setting as we show in Section 7.

Summary. To summarize, in this section we first showed that an alternative way for constructing a linear functional operator representation of extrinsic vector fields consists in considering a family of deformations of the shape, constructing the associated shape difference operators, and taking their derivative at zero, which leads to infinitesimal shape differences. This also suggests alternative functional deformation field operators, sensitive only to specific kinds of deformations (e.g., non area-preserving or non-conformal). Finally, we showed that by modifying the definition of shape differences, a new, unified difference operator can be constructed and that its derivative at time zero leads precisely to the functional deformation field formulation introduced in the previous section.

6 DISCRETE SETTING

In this section we provide the discretization of functional deformation fields. For this, we first propose a particular discretization of the Levi-Civita connection and the Lie derivative of the metric on the triangle mesh, which leads to a simple formula for the operator E^V. In the following section, Sec. 7, which can be skipped similarly to Sec. 5, we demonstrate that the deep connection between functional

deformation fields and infinitesimal shape difference operators also holds in the discrete setting.

Throughout this section, we assume that we are given a manifold triangle mesh. We denote by $(\mathcal{X}, \mathcal{E}, \mathcal{F})$ respectively the set of vertices, edges and faces. We will consider the deformation field V, which we also call an extrinsic vector field, to be given as a three-dimensional vector per vertex.

Discrete connection. To build the discrete operator E^V we need a consistent discretization of the Levi-Civita connection. While several discrete connections have been proposed (e.g. [4, 31]), because of the special nature of our problem, we choose to build our own. This is because, applications such as parallel transport require that the vectors u, v and $\nabla_u v$ are expressed in the same space (at vertex or face or edge) so often an averaging step has to be introduced to transfer, for example, a face-based representation of a vector to an edge based representation. In our setting such a requirement is not needed and it is easier to distinguish tangent vector fields that will be expressed by one vector per face and extrinsic vector fields expressed at vertices. Thus, our goal is to obtain a connection of the ambient space $\bar{\nabla}_u V$ where u is a tangent vector and V is an extrinsic vector field :

$$\begin{array}{rccc} \bar{\nabla}: & \mathbb{R}^{3|\mathcal{F}|} \times \mathbb{R}^{3|\mathcal{V}|} & \rightarrow & \mathbb{R}^{3|\mathcal{F}|} \\ & (u, V) & \mapsto & \bar{\nabla}_u V \end{array}$$

We build the connection $\bar{\nabla}$ using finite differences as follows. Since extrinsic vector fields are defined at vertices the differences are taken along the edges.

Definition 6.1. In a given triangle $T \in \mathcal{F}$ the ambient covariant derivative along the edge e_{ij} is defined by

$$\left(\bar{\nabla}_{\frac{e_{ij}}{\|e_{ij}\|}} V\right)_T = \frac{V_i - V_j}{\|e_{ij}\|}.$$

Thus the ambient connection in the directions $E = (e_{ij}, e_{jk})$ can be stored in a matrix

$$(\bar{\nabla}_E V)_T = \begin{pmatrix} V_i - V_j & V_j - V_k \end{pmatrix}. \tag{7}$$

Then, given any tangent vector $x = E\alpha$, the covariant derivative in its direction can be computed as $\bar{\nabla}_x V = (\bar{\nabla}_E V)\alpha$.

Our construction is closely related to previous discretizations of the Levi-Cevita connection (see, e.g., Section 3.4 in [4]) with the main difference that we do not project the result onto the tangent plane at a point and also avoid the averaging of values from faces onto the mesh edges. Given the expression above, the discrete Lie derivative of the metric at triangle T follows immediately, using Eq. (1). Namely for any pair of tangent vectors $x = E\alpha, y = E\beta$ in the triangle T, we have:

$$\mathcal{L}_V \mathbf{g}(x, y)_T = \langle x, (\bar{\nabla}_E V)\beta \rangle + \langle (\bar{\nabla}_E V)\alpha, y \rangle. \tag{8}$$

If W_M denotes the cotangent-weight Laplacian, which classically represents the inner products of H_0^1 (and is also called *stiffness matrix*), we obtain the discrete functional deformation field operator from its definition (2):

$$f^\top W_M E^V g = -\sum_{T \in \mathcal{F}} \mathcal{L}_V \mathbf{g}(\nabla f, \nabla g)_T \mu(T).$$

Then we obtain $E^V(u) = W_M^{-1} H$, where H is a Laplacian matrix whose weights depend on the extrinsic vector field:

$$\begin{aligned} (H)_{ij} &= \frac{1}{2} \sum_{j \sim i} (c(T_{\alpha_{ij}}) + c(T_{\beta_{ij}})), \\ c(T) &= (\langle e_{jk}, V_j - V_i \rangle + \langle e_{ij}, V_j - V_k \rangle) \frac{1}{4\mu(T)} \\ &\quad - \operatorname{Tr}\left((E^\top E)^{-1} E^\top (\nabla_E V)\right) \frac{\langle e_{jk}, e_{ki} \rangle}{\mu(T)}. \end{aligned}$$

The computations can be found in the supplemental material.

6.1 Properties

Interestingly, many of the properties of the continuous operators are satisfied exactly by their discrete counterparts.

Linearity. The discretization $E^V(f)$ naturally preserves the linearity with respect to both V and f which is very convenient for practical purposes.

In practice, it is often convenient to use a functional basis, so that any function can be represented as a linear combination of some basis functions ϕ_i. Given such a basis, the operator E^V can be seen as the (possibly infinite) matrix: ${E^V}_{ij} = \langle \phi_i, E^V(\phi_j) \rangle_{L^2(M)}$. The choice of basis depends on the application. Since we are interested in smooth deformations of a surface, we take a subset of the smoothest functions given by the first k eigenfunctions of the Laplace-Beltrami operator. In that case, E^V will be represented simply as a $k \times k$ matrix. As shown in Figure 18 the size of the basis k affects the deformation field that we can represent and recover. Increasing k allows a more faithful representation of high frequency deformation fields.

The linearity with respect to V allows the same operation for vector fields. Therefore, if the deformation field is given in some basis $V = \sum_i \alpha_i X_i$ then the operator reads $E^V = \sum_i \alpha_i E^{X_i}$. This means that when designing a deformation field V we can consider an objective as a function of the coefficients α.

Vector Fields representation. In the continuous setting the kernel of $V \mapsto E^V$ is the set of infinitesimal isometries. However, to the best of our knowledge, there is no characterization of how often this set is reduced to rigid motion. In the particular setting of our discretization some standard results can be applied, however.

Proposition 6.2. *For almost all triangle meshes M without boundary, the operator E^V uniquely defines the extrinsic vector field V up to rigid motion.*

Thanks to this proposition, we can guarantee that E^V is almost always a complete coordinate-free representation of extrinsic vector fields V. Triangle meshes containing perfectly flat neighborhoods fall in the category of shapes on which the map $V \mapsto E^V$ is not injective. Namely, since by definition of the strain tensor (Eq. 2), whenever $\nabla_x V$ and is normal to the surface for all x, (as is the case when e.g. V is a normal field on a flat part and zero elsewhere), the tensor $\mathcal{L}_V$ will lead to the zero operator. Although we have found that for organic and natural shapes, such vector fields are rare or non-existent, they can nevertheless be important for coarse meshes or man-made objects with flat areas.

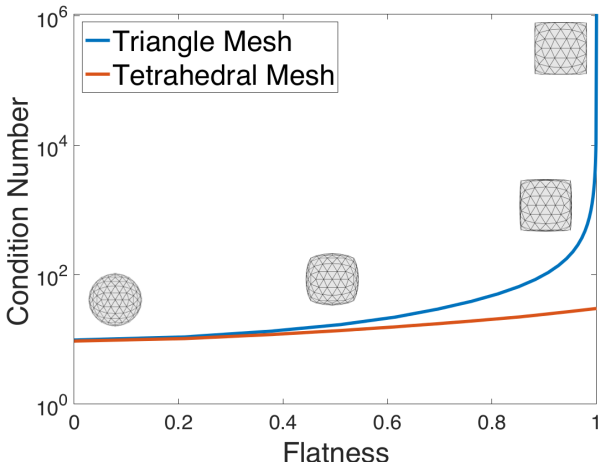


Fig. 4. Comparison of the condition number of the linear map $V \mapsto E^V$ when computed on triangle and tetrahedral meshes with respectively 162 and 163 vertices. The condition number is computed as the ratio of the largest and the 7th lowest singular value to avoid infinitesimal rotations and translations naturally mapped to zero. We consider the collection of meshes formed by a sphere morphing into a cube. For a triangle mesh the condition number goes to infinity as the shape becomes increasingly flatter. When considering a tetrahedral mesh the condition number remains bounded.

6.2 Construction for Tetrahedral Meshes

To remedy this problem, we extend our discretization to tetrahedral meshes thus avoiding ill-defined vector fields as the kernel of $V \mapsto E^V$ is of dimension 6 (translations and infinitesimal rotations). For this we follow the construction provided in Section 6, by adapting it to tet meshes. Namely, we extend the ambient covariant derivative matrix E in Eq. (7) to three dimensions, by considering the covariant derivative along three directions of a tet mesh, and thus storing a 3x3 matrix $\bar{\nabla}_E V$ per simplex. We then use Eq. (8) without any modifications to obtain a discretization of the functional deformation fields on tet meshes. The final resulting formula for the matrix E_I^V is provided in the supplementary material.

We compare the stability of our representation between tetrahedral and triangle meshes in Figure 4, by plotting the condition number of the linear system for recovering the vector field V from its operator representation E_I^V in the case of surface (triangle) and tet mesh reprentations of a cube. We note that although the condition number becomes unbounded for the triangle mesh representation as the shape approaches a flat cube, it nevertheless remains remarkably stable: even at 0.9 where the sphere is almost a cube the condition number is about 100. In contrast the condition number for tet meshes remains bounded even for a perfectly flat shape.

An important consequence of Proposition 6.2 is that deformation fields are fully encoded by the operator E^V up to *infinitesimal* rigid motions. Therefore any deformation can be recovered regardless of its scale and nature. For instance Figure 18 shows that non-infinitesimal *global rotations* are correctly encoded and recovered from our operator representations.

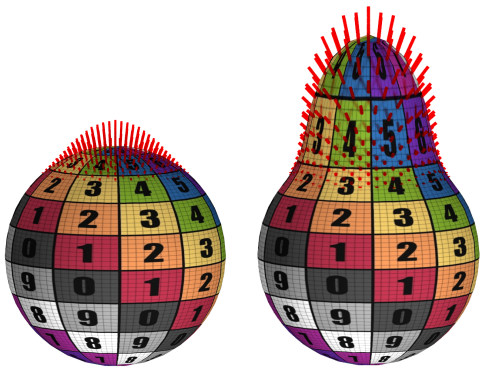


Fig. 5. Example of deformation fields that commute with the diffeomorphism represented by texture transfer. Note that both the direction and the magnitude of the vector field have to adapt to the underlying geometry to produce the same metric distortion.

7 DISCRETE INFINITESIMAL SHAPE DIFFERENCES

Similarly to the link established in Section 5 between our initial definition for functional deformation fields and infinitesimal shape differences, we can consider an alternative discretization to the one above by considering a family of deformed meshes and taking the derivative of shape difference operators. In this section we show that this approach leads to exactly the same result, which means that remarkably Proposition 5.4 is satisfied *exactly* in the discrete setting. To demonstrate this result we first provide a discretization of the unified shape difference operator and then highlight the link between the infinitesimal shape difference operators and functional deformation fields. Similarly to Section 5, this section is primarily of conceptual interest and can be skipped by readers who wish to proceed to the practical results.

To compute the shape differences we start from the discretization of the inner product $\langle ., . \rangle_{H_0^1}$ using standard first order finite elements. We will denote by L the classical cotangent Laplacian matrix, W the inner product of H_0^1 and A the lumped mass matrix such that $L = A^{-1}W$. As before μ is a measure and $\mu(T)$ denotes the area of triangle T.

7.1 Discrete unified shape difference

The discretization of the unified shape difference is straightforward when N and M are triangle meshes and share the same connectivity. In Definition 5.2 given above, the gradients and the point-wise scalar products are taken on N while the measure $\mathrm{d}\mu^M$ comes from M. Therefore the right hand side can be discretized by a modified cotangent weight formula:

$$W_M D_I = W_N^M, \text{ where}$$

$$(W_N^M)_{i,j} = \frac{1}{2}\left(\frac{\mu^M(T_\alpha)}{\mu^N(T_\alpha)}\cot\alpha_{ij}^N + \frac{\mu^M(T_\beta)}{\mu^N(T_\beta)}\cot\beta_{ij}^N\right). \quad (9)$$

Here T_α, T_β are the two triangles adjacent to edge i, j, which is opposite to angles α and β, while μ^M and μ^N are the triangle areas on shapes M and N respectively. Note that W_N^M differs from the standard cotangent weight matrix W_N only by the ratio of weights

per triangle. Moreover,notice that if the transformation is area preserving for all triangles then D_I reduces to the conformal shape difference defined in [44] (Option 1 in Section 5).

From the expression above it follows that $D_I = W_M^{-1} W_N^M$.

Expression in a basis. Similarly to the construction given in [44] we can also express the unified shape difference when the basis Φ_M on the source shape M is given by the eigenfunctions of the Laplace-Beltrami operator. In that case, using a diagonal matrix Λ_M of eigenvalues, D_I becomes:

$$D_I = \Lambda_M^{-1} \Phi_M^T W_N^M \Phi_M.$$

This expression has the advantage of avoiding the inverse of a large sparse matrix, and can be used to analyze deformation of a shape with fixed connectivity in multi-scale basis, which can make the computations resilient to local perturbations (see Option 3 in Section 5 of [44]).

Approximation with a functional map. Note that both expressions above assume that the source and target meshes share the same connectivity. When the meshes have different connectivity this discretization requires a map between triangles making it challenging to use in practice. To overcome this problem we approximate this discrete formulation by transferring the weights on triangles to lumped weights on vertices. The approximation then reduces to the usual discrete quantities:

$$(\tilde{W_N^M})_{ij} \approx \frac{\sum_{t \sim i} \mu^M(T_t)}{\sum_{t \sim i} \mu^N(T_t)} \frac{1}{2} \sum_{j \sim i} (\cot \alpha_{ij}^N + \cot \beta_{ij}^N).$$

We recognize here the cotangent Laplacian L_N with lumped area weights, namely $A_M L_N$. In the case of meshes with different connectivity, this remark suggests the following approximation of the isometric shape difference, valid only in a discrete sense, for an arbitrary linear functional map C between M and N:

$$f^\top A_M L_M D_I g \approx f^\top A_M C^{-1} L_N C g.$$

In the reduced basis of the Laplacian eigenvectors, the approximation of the shape difference becomes $D_I \approx \Lambda_M^{-1} C^{-1} \Lambda_N C$, which preserves the principal property of the operator: D_I is identity if and only if the deformation is an isometry since the Laplacian on N has to be equal to the Laplacian on M. We used this discretization in Figure 3 and observed that the two expressions given above typically produce similar results.

7.2 Shape difference derivative

Suppose that each vertex p_i of the mesh is displaced by the vector V_i by $p_i^t = p_i + tV_i$. This produces a family of triangle meshes $(\mathcal{X}^t, \mathcal{E}, \mathcal{F})$ with identical connectivity. It is now possible to take the derivative with respect to t of Eq. (9) at time 0. This way we obtain a discretization of the infinitesimal shape differences. Remarkably the resulting discretization is strictly identical to the discrete functional operator E^V proposed in Section 6 based on the discrete Levi-Civita connection.

Proposition 7.1. *The discretization of E based on the discrete Levi-Civita connection is equivalent to the one obtained by differentiating the unified shape difference operator.*

Shape difference decomposition. Since the discretization using a discrete connection and through the time derivative agree, the decomposition described by Eq. (6) is also satisfied exactly. Namely, the matrix ${E^V}_C$ representing the discrete infinitesimal conformal shape difference splits into the discrete functional deformation field E^V and an appropriately defined discrete divergence:

$$f^\top W {E^V}_C^V g = f^\top W E^V g + \sum_{T \in \mathcal{F}} \operatorname{div}(V)_T \langle \nabla f, \nabla g \rangle_T \mu(T).$$

Thus, the decomposition of E^V, representing isometric distortion, into area and conformal parts given in Eq. (6) in the continuous case holds exactly in the discrete case as well.

8 EXPERIMENTS

In this section we apply our constructions to various tasks in shape correspondence, deformation design and analysis. As our framework relies on manipulating moderately-sized matrices, all of the applications are very efficient, even when combining multiple objectives.

In some applications (Sec. 8.2 - 8.5), it is necessary to recover the deformation field from its function operator representation. For this, we construct a reduced basis (dictionary) of deformation fields and recover the coefficients of the unknown deformation by solving a convex problem similar to basis pursuit. Namely, given a target functional deformation field operator E^V represented as a matrix, expressed in some fixed functional basis, we solve for V via:

$$\min_\alpha \| \sum_i \alpha_i E^{X_i} - E^V \|_F^2 + \tau \|\alpha\|_1, \tag{10}$$

where α is a vector of coefficients and E^{X_i} are the functional representation of the i^{th} deformation field in an overcomplete basis (dictionary). Of course, the choice of basis is application dependent. The simplest and most general choice would be to consider a basis which consists of independent displacements at each vertex of the given mesh. For a mesh with n_V vertices, this results in $3n_V$ unknowns when solving for a deformation field, which is feasible when n_V is small (and is used in the experiment in Figure 18), but can be expensive for larger meshes. When needed (Sec. 8.2 - 8.5) we use the following deformation bases:

- The simplest option is to take the eigenfunctions of the Laplace-Beltrami operator as the basis for each component of the deformation field. While simple, this basis might not preserve rotation invariance.
- Alternatively we construct a basis via modal analysis of a deformation energy. In particular we consider an energy of the form $V \mapsto \int_M \|\bar{\nabla} V\|^2 \mathrm{d}\mu$. This corresponds to the energy on a particular discretization of the Bochner Laplacian of extrinsic vector fields. To obtain the basis we take the eigenvectors of the Hessian of the energy, which correspond to smooth deformation fields.
- Lastly, we use the handle-based deformation model described in [1]. Unlike the other families, the deformation fields arising from this model are compactly supported and therefore better suited to reproduce local deformations.

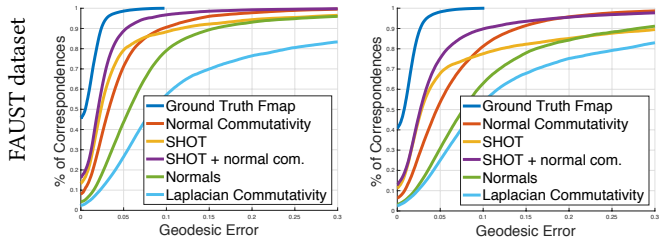


Fig. 6. Evaluation of the computed maps on 100 pairs of shapes from the FAUST dataset [7]. Left: different character taking a similar pose, right: different characters in arbitrary poses.

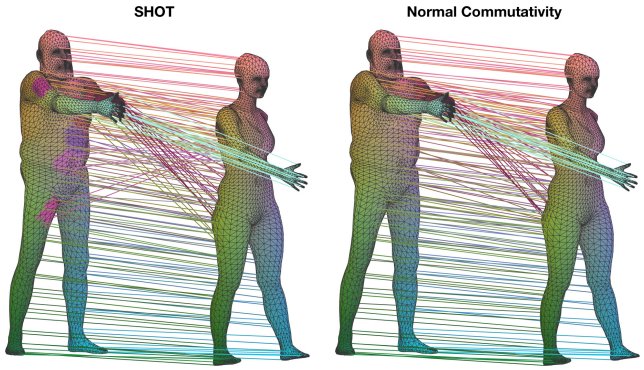


Fig. 7. An example of the point-to-point map evaluated in Figure 6. The RGB channel represents the xyz-coordinates, which are transferred using the recovered point-to-point map. The correspondences obtained with SHOT (left) are precise near sharp features (e.g., on the fingers) whereas our constraint (right) is informative on the entire shape.

In our experiments in Sec. 8.2 - 8.5, we parameterize the space of deformations by computing 180 deformation fields in each of the three categories mentioned above to build an over-complete dictionary. This implies that the number of unknowns is relatively small: $\alpha \in \mathbb{R}^{540}$ and is independent of the resolution of the underlying mesh. We choose the parameter τ, controlling the sparsity of the representation, to be 10^{-4} times the largest singular value of the linear map $V \mapsto E^V$. We use the L_1 norm regularizer in Eq. (10) as a sparsity-promoting prior following the widely-used LASSO regularization. In our case, this allows to better recover localized deformation fields, instead of the L_2 penalty, which can lead to dense solutions, possibly resulting in global deformations.

8.1 Functional map inference

In our first application, we show how our functional deformation field representation can be used as a regularization in shape matching problems. In particular, we show how this representation can be used to add information related to the second fundamental form to the computation of functional maps [36]. The vast majority of the existing methods for shape correspondence with functional maps use the assumption of approximate intrinsic isometries (see [37] for an overview) and are either purely intrinsic or inject extrinsic or embedding-dependent information by adding extrinsic descriptors.

On the other hand, our functional deformation field representation provides a natural coordinate-free way to add embedding-dependent information into the map estimation pipeline. In particular, our approach below is based on the following key observation:

Proposition 8.1. *Given a pair of surfaces M, N embedded in 3D, and a diffeomorphism $T : N \to M$, let C be the corresponding functional map $L^2(M) \to L^2(N)$. Then M and N are related by a rigid motion in space if and only if:*

$$\|C\Delta_M - \Delta_N C\| + \|CE^n_M - E^n_N C\| = 0,$$

where Δ are the LB operators, while E^n are functional deformation fields arising from the normal fields.

This proposition is simply a consequence of the fundamental theorem of surface theory and the relation between functional deformation fields and the second fundamental form described in Section 4.1. Note that enforcing the condition of this proposition in practice reduces simply to penalizing the lack of commutativity of the functional map C with predefined operators, which can be done efficiently in practice. Therefore, we can see that functional deformation fields provide an effective way to capture embedding-dependent information in a coordinate-free way that *fully characterizes the shape geometry* up to rigid motions.

Inspired by this observation, we propose to solve the following problem: given two shapes and a sparse set of correspondences recover a dense map. The shapes come from the FAUST dataset [7] and we are given five corresponding landmarks at the hands, feet and head. The baseline method following the logic of the original paper is to represent the landmark points as delta functions δ_M and δ_N and look for the most isometric functional map $C : L^2(M) \to L^2(N)$, by enforcing commutativity with the Laplace-Beltrami operator. Thus, the straightforward approach would be to solve the optimization problem:

$$\min_C \|C\Delta_M - \Delta_N C\|_F^2 \quad \text{s.t.} \quad C\delta_M = \delta_N.$$

The basic way to add extrinsic information to this problem is to constrain the map to preserve extrinsic descriptors denoted F_M, F_N respectively on M, N:

$$\min_C \|C\Delta_M - \Delta_N C\|_F^2 + \|CF_M - F_N\|_F^2 \quad \text{s.t.} \quad C\delta_M = \delta_N.$$

We evaluate two commonly-used descriptors: 1) the normal vector field encoded as three independent functions and 2) the purely extrinsic descriptor SHOT [52] successfully used for solving partial matching problems [30].

We compare these descriptor-based approaches to our coordinate-free constraint that promotes the preservation of the second fundamental form. According to Prop. 8.1 if a diffeomorphism commutes with the Laplace-Beltrami operator and E^n then the shapes admit the same embedding. Our new optimization problem thus reads:

$$\min_C \|C\Delta_M - \Delta_N C\|_F^2 + \|CE^n{}_M - E^n{}_N C\|_F^2 \text{ s.t. } C\delta_M = \delta_N.$$

Once the functional map are obtained they are converted to a point-to-point map using the basic approach described in [36]. In all experiments, we use $k = 100$ Laplace-Beltrami eigenfunctions to encode the functional map and the operators E^n. The results are shown for two non-isometric shape matching problems: different

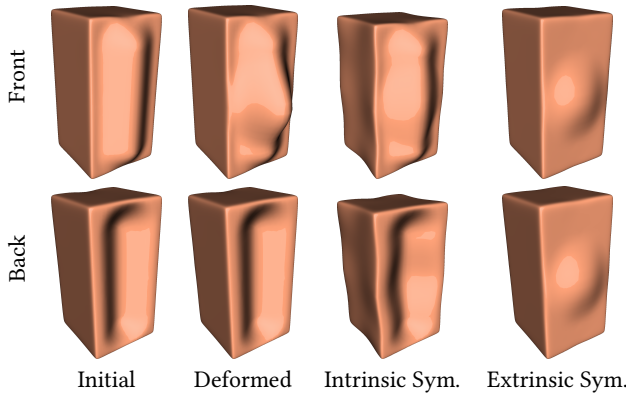


Fig. 8. An initially intrinsically symmetric bar (far left) is artificially made asymmetric (middle left). Our algorithm (middle right) is able to recover the symmetry while maintaining the intrinsic structure of the shape. In comparison an extrinsic symmetrization scheme (far right) would erase the intrinsic structure.

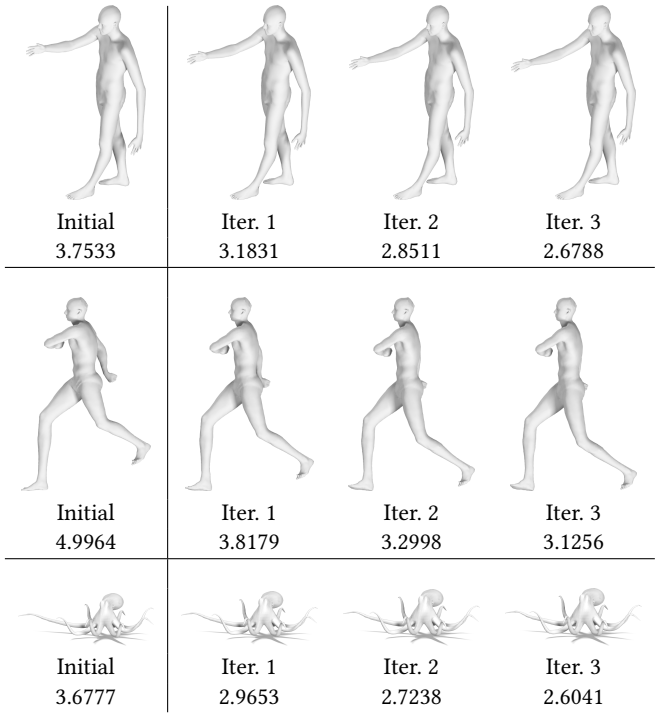


Fig. 9. Three iterations of our intrinsic symmetrization method, Algorithm 1, given an approximate symmetry map. At each step we measure the distance to the symmetry by the Frobenius norm between the intrinsic shape difference induced by the symmetry map and identity, namely $\|D_I - I\|_F$. Although not directly taken into account, this energy decreases at each iteration. Note that our algorithm works with any type of symmetries: see bottom row for a non-reflectional symmetry.

characters taking the same pose and taking different pose. Figure 6 shows the percentage of correspondences within a given geodesic distance. Interestingly, SHOT provides valuable information on sharp features allowing accurate matching near salient points but tends to fail on featureless regions. In comparison our constraint provides information everywhere on the shape making the results less subject to obvious mismatching but it is less informative on the placement of salient features. This intuition is confirmed by Figure 7 which provides a visualization of the point-to-point correspondences by transferring the coordinates functions encoded as RGB channels. Finally, the combination of those two constraints overcomes the limitations of both methods taken independently. Figures 23, 24 and 25 (in the appendix) further illustrate the effect of our approach on challenging non-isometric shape pairs both in similar and arbitrary poses. Since our approach is based on the preservation of the second fundamental form, we observe that this assumption is well-respected in settings such as articulated motion of humans or animals. Thus, our normal commutativity term leads to a particularly strong improvement for shapes in similar poses, but also, to a smaller extent, in other, more diverse, settings. Nevertheless, in some cases, our constraint can have a negative effect. For example, on the FAUST dataset, we obtained 1 pair (out of 100 tested) on which the average error with normal commutativity is worse than without it. For the non-isometric man-woman and dog-lion test sets in arbitrary poses, there were 5 cases out of 60, and 3 out of 60 pairs respectively, where normal commutativity led to higher average error. In our experience, these cases are rare and only occur in strongly non-isometric pairs of shapes in very different poses.

8.2 Intrinsic Symmetrization

In this section we show how our representation of deformation fields can be used to deform shapes to make them more intrinsically symmetric, while keeping their general pose. This step is essential for symmetry-aware quad-meshing presented in Sec. 8.3 For example, Figure 8 shows a shape with important features which would be lost by an *extrinsic* symmetrization scheme. However an *intrinsic* symmetrization algorithm would preserve those features while recovering the symmetry. This way, our goal is similar to the one of [62] although our approach, unlike theirs, avoids the computation of a skeleton and is purely intrinsic. In addition, as input we only require a *functional* representation of the symmetry, and do not rely on a precise, e.g., bijective, pointwise map. More specifically, given a shape M and a functional representation of a self-map $\pi : M \to M$ we would like to compute the shape M' such that the self-map ψ on M' is an isometry. If we denote by $\varphi : M' \to M$ the map from M to M' then the symmetry map on the deformed shape is given by $\psi = \varphi^{-1} \circ \pi \circ \varphi$. Using Prop. 5.3 the isometric constraint is satisfied if and only if the unified shape difference D_I^{ψ}, computed with the map ψ, equals identity. If C_T is the functional map representation of a map T, then after simplification this is equivalent to $D_I^{\pi} C_{\pi}^{-1} D_I^{\varphi} C_{\pi} = D_I^{\varphi}$ (see supplementary material).

Note, however that every intrinsically symmetric shape would be a solution of this equation. Therefore we regularize the problem by imposing that φ should be as-isometric-as possible. The equality conditions are enforced in the least squares sense leading to the

Algorithm 1: Intrinsic Symmetrization

Input : Triangle mesh with vertices p and self-map π
Output: New vertices p^t

repeat
  Find V^{t+1} solution of (12) ;
  Compute new embedding: $p^{t+1} = p^t + V^{t+1}$;
  Recompute D_I^π, C_π ;
until $\|p^{t+1} - p^t\| < \epsilon$;

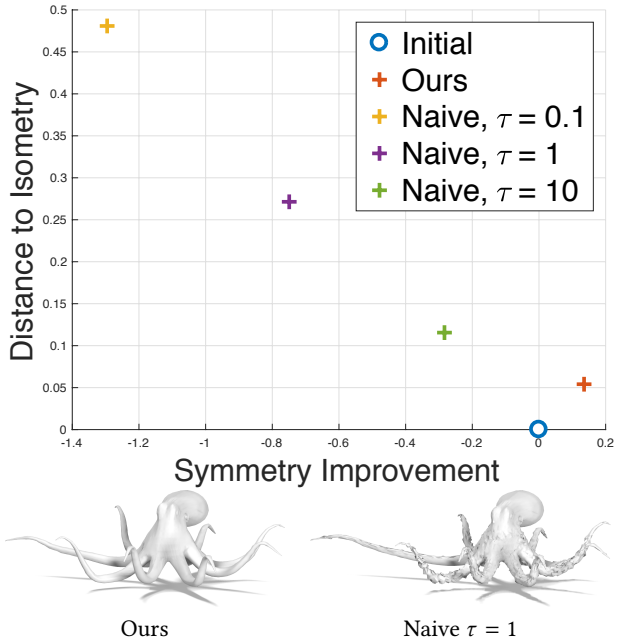


Fig. 10. Comparison between of our intrinsic symmetrization method and the local method Eq. (13). Distance of edge length to their symmetric counterparts versus the distance of edge length to the original mesh in percentage of change w.r.t to the initial mesh. Our method is 13% more symmetric than the original mesh when the local methods lead to noisy output or heavily distorted outputs.

optimization problem:

$$\min_{\varphi} \|D_I^\pi C_\pi^{-1} D_I^\varphi C_\pi - D_I^\varphi\|_F^2 + \tau \|D_I^\varphi - I\|_F^2. \tag{11}$$

The optimization (11) is restricted to the set of diffeomorphisms so a direct approach is challenging to use in practice. A more tractable method is to use functional deformation fields as a first order approximation of shape differences and thus find the deformation field that solves (11) to first order. After linearization, Eq. (11) becomes:

$$\min_{V} \|D_I^\pi C_\pi^{-1} E^V C_\pi - E^V - I + D_I^\pi\|_F^2 + \tau \|E^V\|_F^2. \tag{12}$$

This linearization suggests an iterative algorithm (described in Algorithm 1) which alternates between solving the linearized problem (12) and computing the new vertex positions. In practice, we construct an over-complete dictionary of deformation fields, composed of the three bases described at the beginning of Sec. 8 and compute the optimal deformation field by solving for the coefficients α.

Figure 9 shows two examples where our method successfully recovers intrinsic symmetry from meshes with outstretched parts. To obtain these results we require only a low quality functional map C_π. In [62] the authors propose a method based on skeleton driven deformation to achieve intrinsic symmetry but limited to reflectional symmetries. Our method does not require such assumptions and works for any given self-mapping (e.g. bottom row in Figure 9). Note also that our deformation field representation is essential in this scenario, since for example, representing deformation fields through displacement functions would not provide information on the necessary (or induced) metric distortion.

An alternative approach to intrinsic symmetrization is to consider the local change in edge length of every edge in the mesh. Thus, given a point-to-point map π (e.g., obtained from a functional map), we consider an optimization problem in the vertex positions p:

$$\min_{p} \sum_{(i,j)\in\mathcal{E}} \left(|p_i - p_j| - |p_{\pi(i)} - p_{\pi(j)}|\right)^2 + \tau \|p - p^0\|^2, \tag{13}$$

where the parameter τ controls the strength of the regularization. Figure 10 compares our method to this approach for various values of τ according to two criteria: the isometry of the deformation and the symmetry the results. For the former (y axis), we simply measure the L_1 norm between the initial and final edge lengths, normalized by the sum of the initial edge lengths. To evaluate the symmetry, we compute the normalized difference between lengths of edges given by the ground truth symmetry. A positive value implies an improvement of the symmetry while a negative value means a deterioration (relative to the initial configuration) of the symmetry. Our global, functional method results in better quality, while the local approach provides noisy meshes. In practice the naive method works well only with mesh exhibiting symmetric connectivity where a very high quality symmetry map is available. In comparison our method does not require converting a functional map to a point-to-point map and naturally handles fuzzy self-maps. Thus our method is able to handle gracefully poor quality maps and relies only weakly on the mesh structure itself. We stress that our method is entirely intrinsic, which can be seen, e.g., since the optimization objective in Eq. (12) can be fully written in terms of the mesh edge lengths. In practice the only extrinsic prior given to our method is the choice of a basis for deformation fields. For completeness Figure 26 also compares against the naive approach when the deformations are restricted to this basis. The results, although better than the original naive method, are qualitatively and qualitatively poorer than ours. Please also see the accompanying video for more results.

8.3 Symmetry-aware quad-remeshing

An important application of the intrinsic symmetrization is the generation of symmetrically coherent quad-meshes as presented in the teaser (Fig. 1). The commonly-used three-step quad-meshing algorithm [8, 18, 25] attempts to conform the quad-mesh to the underlying geometry by aligning the quads with the curvature directions. However, this strategy fails to represent global features such as intrinsic symmetries. Some algorithms [3, 38] are able to generate quad-meshes consistent with a given symmetry map. They are, however, limited to isotropic quadrangulations, which can result in non-symmetrically invariant connectivity.

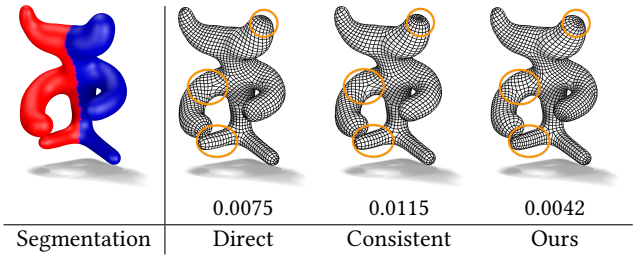


Fig. 11. Symmetric quad remeshing. Left: segmentation using the input symmetry and direct remeshing without symmetry information. Middle: results of the consistent remeshing approach of [3]. Right: quad remeshing after our intrinsic symmetrization. Note that the resulting mesh is more regular (e.g., tip of the ear and hand), and the size of the elements is reduced on the shorter leg, making the mesh more symmetric. The evaluation measures the normalized difference in the number of quads on each side.

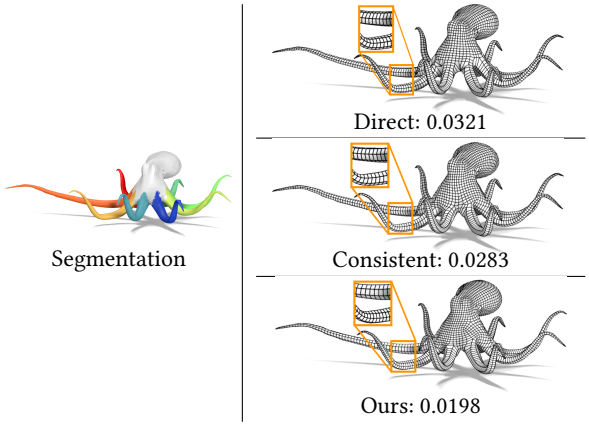


Fig. 12. Symmetric quad remeshing. Left: segmentation using the input symmetry. Right: comparison between direct remeshing, results of [3], and after our intrinsic symmetrization. Note that the size of the elements is increased on the longer leg to making the mesh more symmetric.

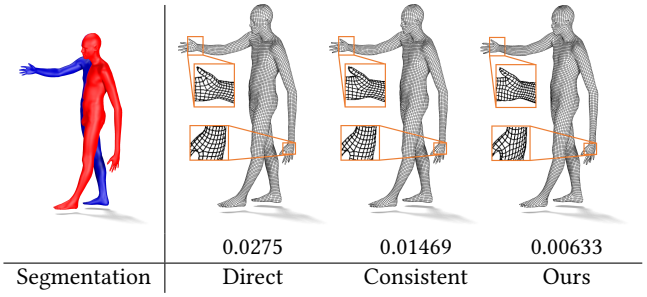


Fig. 13. Symmetric quad remeshing. Left: segmentation using the input symmetry. Right: comparison between direct remeshing, results of [3], and after our intrinsic symmetrization. Note that the quad mesh becomes more regular and the size of the elements is decreased on the shorter arm, reducing the difference in the number of quads.

Our approach adopts a strategy similar to [39], to produce a symmetry-aware quad mesh. The input mesh is first deformed to be intrinsically symmetric using the algorithm described in the previous section. This symmetric mesh is then used for the computation of consistent quad-mesh using [3]. Finally, the quads are pulled-back onto the initial mesh. Figures 11, 12, and 13 compare a direct quad-meshing algorithm, the method recently introduced in [3] and our technique. We also evaluate the results numerically by computing the difference in the number of quads on symmetric parts of the shapes, normalized by the total number of quads (thus, lower is better). Note that our method, due to the anisotropy resulting from the symmetrized mesh, produces quadrangulations that better reflect the intrinsic symmetry on the shape. Figure 14 (top) illustrates this by plotting the percentage of singularities within a geodesic distance of their closest symmetric counterparts, after mapping them using the given symmetry map. Our method spreads singularities more symmetrically across the mesh while maintaining their total number compared to the direct quadrangulation scheme. Moreover the quality of the quads, as measured by the angle distribution shown at the bottom of Figure 14 is comparable with the other two methods.

8.4 Deformation design

In our next application we use our deformation field representation to achieve various global deformation objectives. Similarly to intrinsic symmetrization, we optimize for the deformation field by solving for the coefficients α in our over-complete dictionary described at the beginning of Sec.8.

Since our operator is linear with respect to the deformation field one can easily combine multiple constraints to the deformation vector field. In Figure 15 we show how multiple different constraints can be combined using our representation. First, we can easily require that at a point p the deformation field matches a given vector v, by setting $V(p) = v$, in addition to other global constraints. Second, we can find the most *isometric* (preserving the intrinsic metric) deformation by minimizing $V \mapsto \|E^V\|_F^2$. At the same time, given a self-map represented as a functional map S, we design a symmetric vector field by imposing a constraint of the form $E^V C_S = C_S E^V$.

Similarly, we can impose an *anti-symmetry* constraint by requiring $E^V C_S + C_S E^V = 0$. In comparison, Figure 16 shows an extrinsic deformation design method consisting in projecting each vector field component into the space of symmetric (respectively anti-symmetric) functions. The resulting shapes look quite distorted compared to our solution. Moreover the distance of the conformal shape difference (resp. area-based shape difference) to identity, measuring how far the map S is from an isometry, is of 0.51 (resp. 0.46) for our design and 0.63 (resp. 0.56) for the extrinsic design. Thus, the extrinsic deformation design tends to distort the intrinsic structure of the shape. Finally, we test a regularization technique for the deformation field by imposing the commutativity with the Laplace-Beltrami operator, which tends to spread to the entire shape. Note that despite the diversity of these constraints, they can all be enforced easily using our operator-based representation. In contrast, the straightforward method shown in Figure 16, consisting in projecting the vector field onto the space of symmetric or anti-symmetric functions, fails in this tasks as it is fully extrinsic.

Figures 17 presents an example of joint deformation design. Namely, we impose a set of directional constraints $U(p_j) = u_j$ and $V(q_j) = v_j$

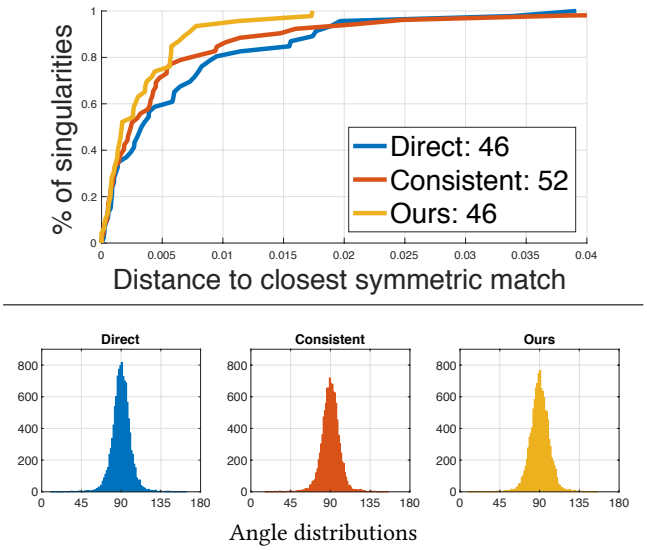


Fig. 14. Top: percentage of singularities within a given geodesic distance of their closet symmetric counterpart, after mapping them using the given symmetry map. Results for the mesh shown in Fig 11. Our method has the most symmetric distribution of singularities while maintaining the angle distributions tightly packed around 90°.

on two different shapes M and N and we solve for two deformation fields, one on each shape, that are "informed" by the deformation of the other shape. On a single shape, our objective is designed to promote smoothness of the resulting deformation field and sparsity in the coefficients of the deformation field basis:

$$\mathbb{E}_M(\alpha) := \sum_{i,j} \alpha_i \langle U_i, \Delta_M U_j \rangle \alpha_j + \tau \|\alpha\|_1.$$

Therefore, on a single shape, the optimization becomes:

$$\min_{\alpha} \mathbb{E}_M(\alpha), \qquad \text{s.t.} \sum_i \alpha_i U_i(p_j) = u_j,$$

where the constraints enforce the given pointwise directions.

To design the deformation fields jointly, we propose to find a field U on shape M and V on shape N such that for a given functional map C we have $E^U C \approx C E^V$ while respecting the local constraints on the respective shape. The resulting optimization problem reads:

$$\min_{\alpha,\beta} \| \sum_i \alpha_i E^{U_i} C - C \sum_j \beta_j E^{V_j} \|_F^2 + \mathbb{E}_M(\alpha) + \mathbb{E}_N(\beta)$$
$$\text{s.t.} \sum_i \alpha_i U_i(p_j) = u_j, \sum_i \beta_i V_i(q_j) = v_j.$$

As a result, the constraints as well as the structure of one shape is transferred onto the other. Moreover the area that could lead to contradictory deformation remains still.

8.5 Functional Deformation transfer

Given a deformation field U on shape M represented as an operator and a functional map C from N to shape M, we can use our method to transfer the deformation to an arbitrary mesh. The transferred deformation $V = \sum_i \alpha_i V_i$ on shape N by solving:

$$\min_{\alpha} \|E^U C - C \sum_i \alpha_i E^{V_i}\|^2 + \tau \|\alpha\|_1. \tag{14}$$

Here, E^{V_i} are the basis deformation fields in our over-complete dictionary, described at the beginning of Sec. 8 and α_i are the unknown coefficients. As in the previous experiments $\alpha \in \mathbb{R}^{540}$ regardless of the mesh size and the linear operators E^{V_i} and C are represented using $k = 200$ eigenfunctions of the Laplace-Beltrami operator. The only exception is the result shown in Fig. 18 where we use the full deformation field basis consisting of $3n_V$ unknowns, for a progressively larger values of k.

We solve this optimization problem with CVX [21], using the default approach based on the interior point method.

Using this setup we solve different instances of the deformation transfer problem, namely:

- Style transfer: we transfer style across poses. Here, given two different shapes in a rest pose and a deformed version of one of them, we transfer the deformation to the other shape (Figure 19). This also shows that our vector field collection is not limited to a specific type of deformation.
- Symmetry transfer: we transfer a deformation from a shape onto itself using a symmetry map (Figure 20). Note that this task cannot be achieved with standard Jacobian-based methods such as [49].

We stress that although enabled by our representation, this is by no means the central application and therefore the results presented below simply serve as an illustration of the functionality that can be achieved using our functional deformation fields.

Style transfer. We use our approach to transfer style across the poses of different shapes in the Faust dataset [7], shown in Figure 19. Here first consider the deformation field U given by the point displacements across two different shapes in approximately the same reference pose. We then use our framework to transfer U to another shape in a different pose and with different mesh structure. In Figure 19 our method consistently preserves the global structure, although some high frequency details of the deformation are lost due to the projection onto a vector field basis.

Symmetry transfer. One interesting feature of the functional representation of deformation fields is that it is "shape aware." For example in Figure 20 we transfer the shrinking of the right leg to the left leg by looking for the operator which commutes with the operator representation of the symmetry map. Since both legs are in different positions this transfer is not easy to achieve by a simple point-to-point transfer of the vector field or even by transferring it using local coordinates. As shown in Figure 20 bottom row, our transferred deformation field adapts to the geometry.

8.6 Relation to existing techniques

An important property of our deformation transfer algorithm is that it relies fully on the deformation of the *metric*. This makes it fundamentally different from the spectral pose transfer described in [28]. Those methods use the strong stability of the first eigenfunctions of the Laplace-Beltrami under deformation. Thus, a deformation field

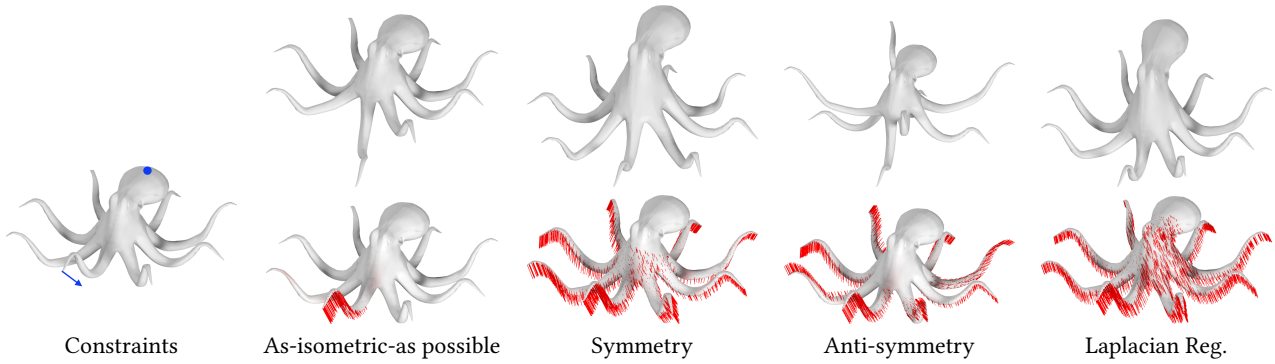


Fig. 15. We design deformations respecting the directional constraints shown on the far left and minimizing various criteria (from left to right): the infinitesimal shape difference leading to the most isometric vector field, the commutativity with a self-map, the anti-commutativity with the same self-map and the commutativity with the Laplace-Beltrami operator.

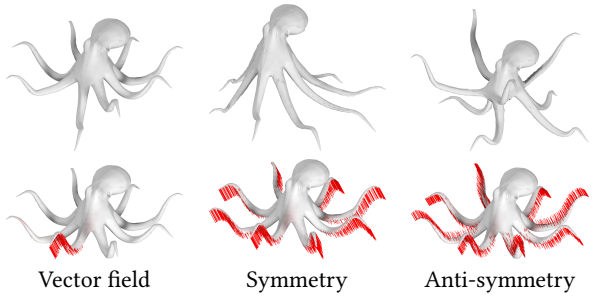


Fig. 16. We design deformations by projecting the vector field shown on the far left onto the space of symmetric and anti-symmetric functions. This direct approach does not deliver the expected results found in Figure 15.

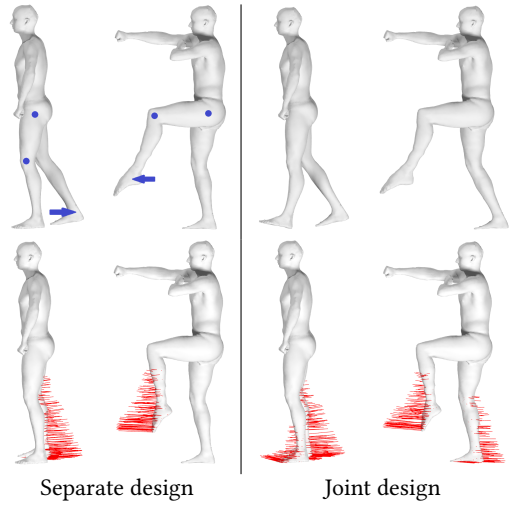


Fig. 17. Left: The deformations are designed separately by minimizing the smoothness term $\langle V, \Delta V \rangle$. Right: joint deformation design by adding the commutativity with the mapping to the optimization. Note that the constraints on one shape tend to be transferred to the other.

V can be efficiently transferred by projecting its components into a reduced eigen-basis Φ of the initial shape and reconstructed using the basis Ψ of the shape to be deformed. The new embedding X' is computed from the old embedding X simply by $X' := X + \Psi\Phi^\top V$.

Recent improvements of this technique [27, 58] include pre-alignment of the spectral basis but the shortcoming are essentially the same. Figure 21 shows that this deformation transfer is by definition extrinsic, orientation dependent and furthermore completely agnostic to the intrinsic structure of the shape. Our method in contrast is rotation-invariant and directly linked to the induced changes in the geometry.

We also compare our method with the algorithm for deformation transfer described in [49]. This method is based on reallocating Jacobian matrices defined on triangles of the source mesh to those of the target mesh. This method, however, is not without limitations. First, this transfer does not take into account changes in orientation from the source to the target thus ruling out any possibility of symmetric transfer and requiring a pre-alignment of the source and target meshes. This can be challenging to achieve in practice in case of non-rigid deformations (e.g. Figure 20). Secondly, it assumes as input a triangle-to-triangle map which can be cumbersome to obtain.

These limitations do not apply to our representation as our approach is based on transferring metric information, and is therefore immune to changes of orientation. Moreover, instead of a triangle-to-triangle map, an approximate functional map is enough. Furthermore, note that although in general reconstructing geometry from metric tensors is more difficult than a reconstruction from Jacobians as local rotations are no longer available (see e.g. [9]) our method relies on solving a moderately-sized convex optimization problem.

Figure 22 shows that working within the functional map framework makes our algorithm more robust to noise usually encounter when using this representation. The computation of functional maps, as described in [36], is done by solving a least squares system incorporating intrinsic descriptors (HKS, WKS) therefore there often exists multiple solutions in presence of an intrinsic symmetry π. We model a noisy functional map C^τ by a linear blending between the direct map (mapping the left to the left and the right to the right) and the anti-symmetric map (mapping the left to the right and the right to the left) represented as operators:

$$C^\tau = \tau C_{\varphi\circ\pi} + (1-\tau)C_\varphi.$$

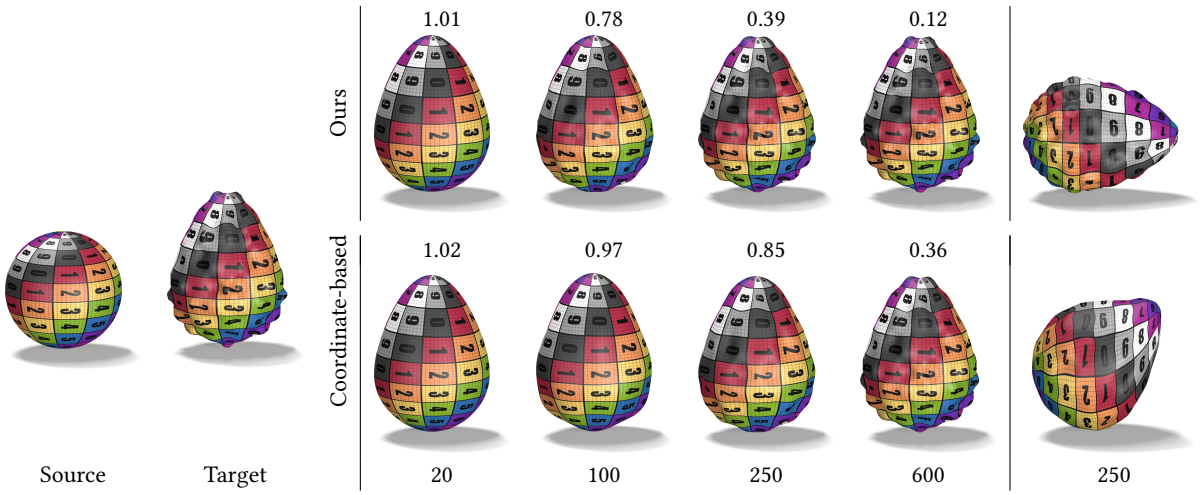


Fig. 18. Reconstructions of a deformation field (left) resulting from the correspondence encoded via the texture map, from its representation as operators (top), or representing it as a triplet of functions (bottom) with increasing number of basis functions. The Hausdorff distance between the deformed mesh and the target is shown on top. Our method also works when the reconstruction is applied to a rotated mesh (right), unlike the coordinate-based approach, used in, e.g. [27, 28]. This latter example is part of our accompanying video.

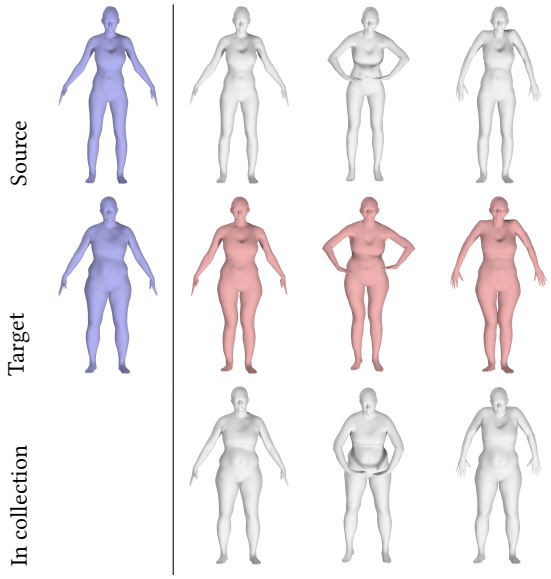


Fig. 19. The deformation field defined by the blue shapes (first column) is transferred to the same shape in different poses (top white shapes). While the style is consistent across the poses (red shapes) some details of the deformation are lost due to the basis representation. The style transfer are compared to the corresponding shape in the collection (bottom white shapes). See also the accompanying video.

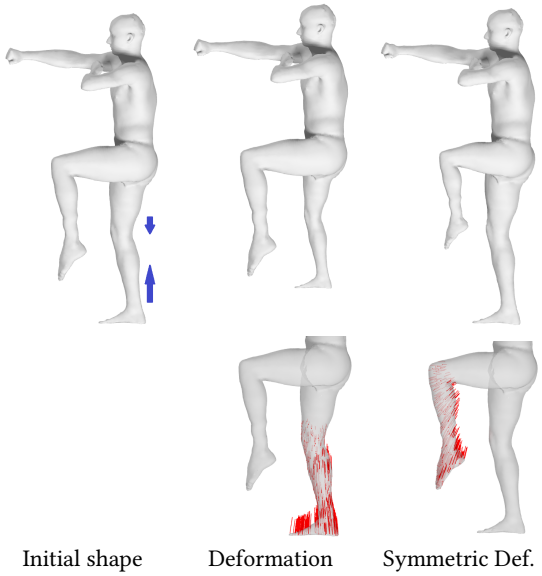


Fig. 20. An initial deformation (first two columns), corresponding to the shrinking of the right leg of a human model, is transferred to the left leg by imposing the commutativity between the infinitesimal shape difference and the symmetry map. Both legs are in different position so the transfer has to adapt to the geometry. See also the accompanying video.

Our method outputs a non-linear interpolation between the deformation and its symmetric version while the method by Sumner et al. exhibits various artifacts.

9 CONCLUSION, LIMITATIONS, FUTURE WORK

In this paper we presented a method for representing extrinsic vector fields as linear operators acting on functions on the shapes, by considering the metric distortion induced by the deformation. We base our representation on the infinitesimal strain tensor and show how it leads to a linear functional operator that can naturally be combined with other such operators including functional maps and the Laplace-Beltrami. We showed how this representation can be used to analyze and design deformations and to introduce extrinsic information into the computation of functional maps.

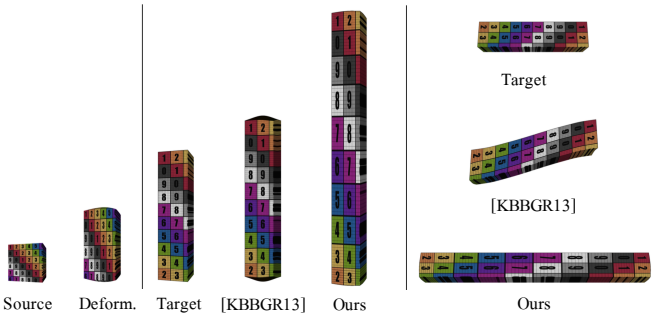


Fig. 21. An initial deformation (first two columns), corresponding to the expansion of a bar is transferred to another longer bar. Using the method described in [27] ([KBBGR13]) the additional height is exactly the same as in the original deformation. Using our method the deformation is indexed on the metric thus the height of the model doubles. Furthermore, our method is rotation invariant (second row).

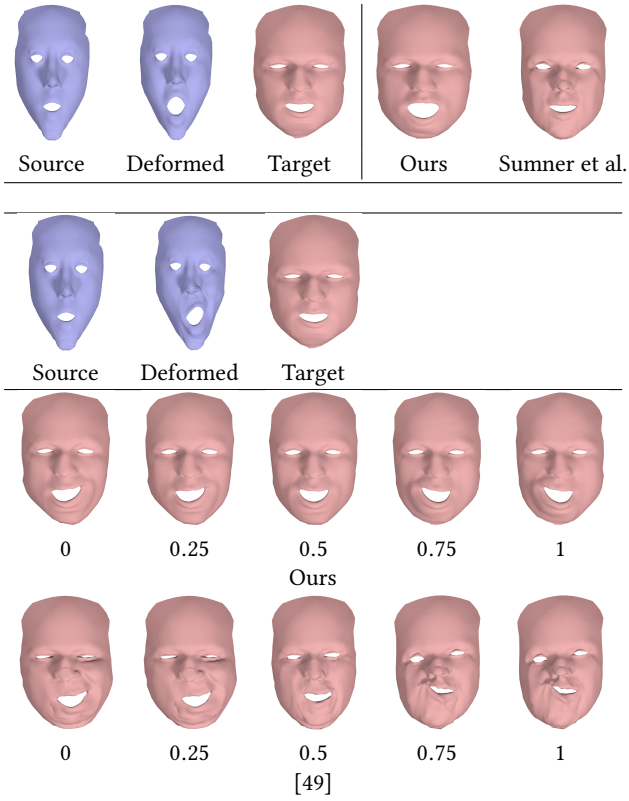


Fig. 22. Deformation transfer using a noisy functional map consisting of a linear blending between the direct functional map ($\tau = 0$) and the symmetric functional map ($\tau = 1$). Our method is robust to this noise and outputs a non-linear interpolation of the deformation and its symmetric version. The method of in [49] fails when provided with a triangle-to-triangle map resulting of the conversion of the noisy functional map.

Our approach has several limitations, depending on the application. In the context of estimating functional maps, our normal commutativity term is only meaningful when the deformation is expected to approximately preserve the second fundamental form. In our experiments, this term leads to particularly strong improvement for, possibly non-isometric, shapes in similar poses. Nevertheless, as mentioned in Sec. 8.1 in rare cases, it can also have a negative effect when this assumption is violated. In other applications, such as deformation transfer or design, one limitation comes from our choice of deformation field basis, which might not be adapted to all types of deformations. A more advanced choice, such as [56] can likely lead to better results in complex scenarios. Finally, our deformation field recovery is based purely on infinitesimal metric distortion, which although, as we prove is generically sufficient, can potentially lead to ill-conditioned systems for specific configurations. Combining our approach with coordinate-based regularization can help avoid such cases.

In the future, we are planning to use the newly introduced functional representation for shape animation. In this context, it would be interesting to establish a connection between the metric on the space of functional deformation fields and different inner products as suggested in [19]. It would also very interesting to use our representation within the framework of shape spaces, e.g. [24], for exploration and design.

REFERENCES

[1] Bart Adams, Maks Ovsjanikov, Michael Wand, Hans-Peter Seidel, and Leonidas J Guibas. 2008. Meshless modeling of deformable shapes and their motion. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 77–86.
[2] Omri Azencot, Mirela Ben-Chen, Frédéric Chazal, and Maks Ovsjanikov. 2013. An operator approach to tangent vector field processing. In *Computer Graphics Forum*, Vol. 32. 73–82.
[3] Omri Azencot, Etienne Corman, Mirela Ben-Chen, and Maks Ovsjanikov. 2017. Consistent functional cross field design for mesh quadrangulation. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 92.
[4] Omri Azencot, Maks Ovsjanikov, Frédéric Chazal, and Mirela Ben-Chen. 2015. Discrete derivatives of vector fields on surfaces–an operator approach. *ACM Transactions on Graphics (TOG)* 34, 3 (2015), 29.
[5] Ilya Baran, Daniel Vlasic, Eitan Grinspun, and Jovan Popović. 2009. Semantic deformation transfer. In *ACM Transactions on Graphics (TOG)*, Vol. 28. ACM, 36.
[6] Alexander I Bobenko, Helmut Pottmann, and Johannes Wallner. 2010. A curvature theory for discrete surfaces based on mesh parallelity. *Math. Ann.* 348, 1 (2010), 1–24.
[7] Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. 2014. FAUST: Dataset and evaluation for 3D mesh registration. In *Proc. CVPR*. IEEE, Piscataway, NJ, USA.
[8] David Bommes, Henrik Zimmer, and Leif Kobbelt. 2009. Mixed-integer quadrangulation. In *ACM Transactions On Graphics (TOG)*, Vol. 28. ACM, 77.
[9] Davide Boscaini, Davide Eynard, Drosos Kourounis, and Michael M Bronstein. 2015. Shape-from-Operator: Recovering Shapes from Intrinsic Operators. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 265–274.
[10] Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, and Bruno Lévy. 2010. *Polygon mesh processing*. CRC press.
[11] Mario Botsch and Olga Sorkine. 2008. On linear variational surface deformation methods. *Visualization and Computer Graphics, IEEE Transactions on* 14, 1 (2008), 213–230.
[12] A. Bronstein, M. Bronstein, and R. Kimmel. 2007. *Numerical geometry of non-rigid shapes*. Springer-Verlag.
[13] George Celniker and Dave Gossard. 1991. Deformable curve and surface finite-elements for free-form shape design. In *ACM SIGGRAPH computer graphics*, Vol. 25. ACM, 257–266.
[14] Philippe G Ciarlet. 2000. *Theory of shells*. Vol. 3. Elsevier.
[15] Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2011. Spin transformations of discrete surfaces. In *ACM Transactions on Graphics (TOG)*, Vol. 30. ACM, 104.
[16] Tamal K Dey, Pawas Ranjan, and Yusu Wang. 2012. Eigen deformation of 3D models. *The Visual Computer* 28, 6-8 (2012), 585–595.
[17] M.P. do Carmo. 2013. *Riemannian Geometry*. Birkhäuser Boston.

[18] Hans-Christian Ebke, David Bommes, Marcel Campen, and Leif Kobbelt. 2013. QEx: robust quad mesh extraction. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 168.

[19] Ilya Eckstein, J-P Pons, Yiying Tong, C-CJ Kuo, and Mathieu Desbrun. 2007. Generalized surface flows for mesh processing. In *Proc. SGP*. Eurographics Association, 183–192.

[20] Michael Eigensatz and Mark Pauly. 2009. Positional, Metric, and Curvature Control for Constraint-Based Surface Deformation. In *Computer Graphics Forum*, Vol. 28. Wiley Online Library, 551–558.

[21] Michael Grant and Stephen Boyd. 2014. CVX: Matlab Software for Disciplined Convex Programming, version 2.1. http://cvxr.com/cvx.

[22] Igor Guskov, Wim Sweldens, and Peter Schröder. 1999. Multiresolution Signal Processing for Meshes. In *Proc. SIGGRAPH*. 325–334.

[23] Tim Hoffmann, Andrew O Sageman-Furnas, and Max Wardetzky. 2016. A Discrete Parametrized Surface Theory in R^3. *International Mathematics Research Notices* (2016).

[24] Martin Kilian, Niloy J Mitra, and Helmut Pottmann. 2007. Geometric modeling in shape space. *ACM Transactions on Graphics (TOG)* 26, 3 (2007), 64.

[25] Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2013. Globally optimal direction fields. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 59.

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

[27] Artiom Kovnatsky, Michael M Bronstein, Alexander M Bronstein, Klaus Glashoff, and Ron Kimmel. 2013. Coupled quasi-harmonic bases. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 439–448.

[28] Bruno Lévy. 2006. Laplace-Beltrami Eigenfunctions Towards an Algorithm That "Understands" Geometry. In *Proceedings of the IEEE International Conference on Shape Modeling and Applications*.

[29] Yaron Lipman, Olga Sorkine, Daniel Cohen-Or, David Levin, Claudio Rossi, and Hans-Peter Seidel. 2004. Differential coordinates for interactive mesh editing. In *Shape Modeling Applications*. 181–190.

[30] Or Litany, Emanuele Rodolà, Alex M Bronstein, and Michael M Bronstein. 2017. Fully spectral partial shape matching. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 247–258.

[31] Beibei Liu, Yiying Tong, Fernando De Goes, and Mathieu Desbrun. 2016. Discrete connection and covariant derivative for vector field analysis and design. *ACM Transactions on Graphics (TOG)* 35, 3 (2016), 23.

[32] J. Martinez Esturo, C. Rössl, and H. Theisel. 2013. Generalized Metric Energies for Continuous Shape Deformation. *Springer LNCS (Proc. Curves and Surfaces 2012)* 8177, 1 (2013), 135–157.

[33] Matthias Müller, Nuttapong Chentanez, Tae-Yong Kim, and Miles Macklin. 2014. Strain Based Dynamics. In *Proc. SCA*. 149–157.

[34] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. 2006. Physically based deformable models in computer graphics. In *Computer graphics forum*, Vol. 25. Wiley Online Library, 809–836.

[35] Andrew Nealen, Olga Sorkine, Marc Alexa, and Daniel Cohen-Or. 2005. A Sketch-Based Interface for Detail-Preserving Mesh Editing. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 24, 3 (2005), 1142–1147.

[36] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. 2012. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 30.

[37] Maks Ovsjanikov, Etienne Corman, Michael Bronstein, Emanuele Rodolà, Mirela Ben-Chen, Leonidas Guibas, Frederic Chazal, and Alex Bronstein. 2016. Computing and processing correspondences with functional maps. In *SIGGRAPH ASIA 2016 Courses*. ACM, 9.

[38] Daniele Panozzo, Yaron Lipman, Enrico Puppo, and Denis Zorin. 2012. Fields on symmetric surfaces. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 111.

[39] Daniele Panozzo, Enrico Puppo, Marco Tarini, and Olga Sorkine-Hornung. 2014. Frame Fields: Anisotropic and Non-Orthogonal Cross Fields. *Proc. ACM SIGGRAPH)* 33, 4 (2014), 134:1–134:11.

[40] Nikolas Paries, Patrick Degener, and Reinhard Klein. 2007. Simple and efficient mesh editing with consistent local frames. In *Computer Graphics and Applications, 2007. PG'07. 15th Pacific Conference on*. IEEE, 461–464.

[41] Dmitry Pavlov, Patrick Mullen, Yiying Tong, Eva Kanso, Jerrold E Marsden, and Mathieu Desbrun. 2011. Structure-preserving discretization of incompressible fluids. *Physica D: Nonlinear Phenomena* 240, 6 (2011), 443–458.

[42] Jonathan Pokrass, Alexander M Bronstein, Michael M Bronstein, Pablo Sprechmann, and Guillermo Sapiro. 2013. Sparse modeling of intrinsic correspondences. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 459–468.

[43] Martin Rumpf and Max Wardetzky. 2014. Geometry processing from an elastic perspective. *GAMM-Mitteilungen* 37, 2 (2014), 184–216.

[44] Raif M. Rustamov, Maks Ovsjanikov, Omri Azencot, Mirela Ben-Chen, Frédéric Chazal, and Leonidas Guibas. 2013. Map-based Exploration of Intrinsic Shape Differences and Variability. *ACM Trans. Graph.* 32, 4 (2013), 72:1–72:12.

[45] Matan Sela, Yonathan Aflalo, and Ron Kimmel. 2015. Computational caricaturization of surfaces. *CVIU* 141 (2015), 1–17.

[46] Justin Solomon, Mirela Ben-Chen, Adrian Butscher, and Leonidas Guibas. 2011. As-Killing-As-Possible Vector Fields for Planar Deformation. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 1543–1552.

[47] Olga Sorkine and Marc Alexa. 2007. As-Rigid-As-Possible Surface Modeling. In *Proc. SGP*. 109–116.

[48] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and H-P Seidel. 2004. Laplacian surface editing. In *Proc. SGP*. ACM, 175–184.

[49] Robert W Sumner and Jovan Popović. 2004. Deformation transfer for triangle meshes. In *ACM Transactions on Graphics (TOG)*, Vol. 23. 399–405.

[50] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. 1987. Elastically deformable models. In *ACM Siggraph Computer Graphics*, Vol. 21. ACM, 205–214.

[51] Bernhard Thomaszewski, Simon Pabst, and Wolfgang Strasser. 2009. Continuum-based Strain Limiting. In *Computer Graphics Forum*, Vol. 28. Wiley Online Library, 569–576.

[52] Federico Tombari, Samuele Salti, and Luigi Di Stefano. 2010. Unique signatures of histograms for local surface description. In *European conference on computer vision*. Springer, 356–369.

[53] Bruno Vallet and Bruno Lévy. 2008. Spectral Geometry Processing with Manifold Harmonics. *Comput. Graph. Forum* 27, 2 (2008), 251–260.

[54] Amir Vaxman, Christian Müller, and Ofir Weber. 2015. Conformal mesh deformations with Möbius transformations. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 55.

[55] Wolfram von Funck, Holger Theisel, and Hans-Peter Seidel. 2006. Vector field based shape deformations. In *Proc. SIGGRAPH*, Vol. 25. 1118–1125.

[56] Christoph Von-Tycowicz, Christian Schulz, Hans-Peter Seidel, and Klaus Hildebrandt. 2015. Real-Time Nonlinear Shape Interpolation. *ACM Transactions on Graphics (TOG)* 34, 3 (2015), 34.

[57] William Welch and Andrew Witkin. 1992. Variational surface modeling. In *ACM SIGGRAPH computer graphics*, Vol. 26. ACM, 157–166.

[58] Mengxiao Yin, Guiqing Li, Huina Lu, Yaobin Ouyang, Zhibang Zhang, and Chuhua Xian. 2015. Spectral pose transfer. *Computer Aided Geometric Design* 35 (2015), 82–94.

[59] Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2004. Mesh editing with poisson-based gradient field manipulation. *ACM Transactions on Graphics (TOG)* 23, 3 (2004), 644–651.

[60] Rhaleb Zayer, Christian Rössl, Zachi Karni, and Hans-Peter Seidel. 2005. Harmonic guidance for surface deformation. In *Computer Graphics Forum*, Vol. 24. Wiley Online Library, 601–609.

[61] Zhibang Zhang, Guiqing Li, Huina Lu, Yaobin Ouyang, Mengxiao Yin, and Chuhua Xian. 2015. Fast as-isometric-as-possible shape interpolation. *Computers & Graphics* 46 (2015), 244–256.

[62] Qian Zheng, Zhuming Hao, Hui Huang, Kai Xu, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2015. Skeleton-Intrinsic Symmetrization of Shapes. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 275–286.

[63] Denis Zorin, Peter Schröder, and Wim Sweldens. 1997. Interactive Multiresolution Mesh Editing. In *Proc. SIGGRAPH*. 259–268.

FUNCTIONAL MAP INFERENCE

Figures 23, 24 and 25 illustrate the performance of our functional map computation on challenging non-isometric shape pairs.

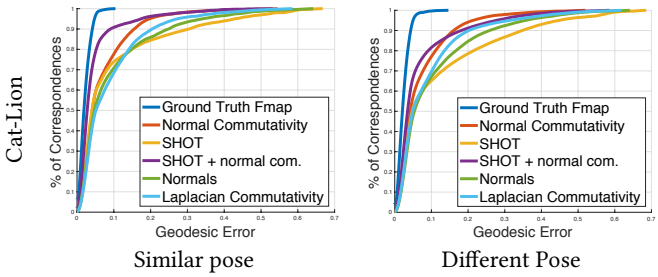


Fig. 23. Evaluation of functional maps computed on 60 pairs of non-isometric cat and lion shapes from [49].

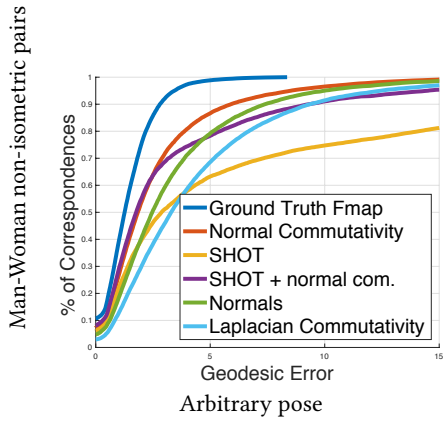


Fig. 24. Evaluation of maps computed on 60 pairs of non-isometric man and woman shapes from the TOSCA dataset [12].

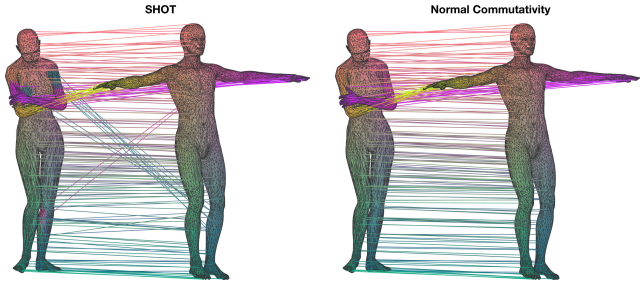


Fig. 25. An example of the point-to-point map evaluated in Figure 24. The RGB channel (left column) represents the xyz-coordinates, which are transferred using the recovered point-to-point map.

INTRINSIC SYMMETRIZATION

Figure 26 provides an additional comparison of our intrinsic symmetrization method to a naive symmetrization method with and without restricting the deformation to lie in the space spanned by a basis of deformation fields, using the same protocol as in Figure 10.

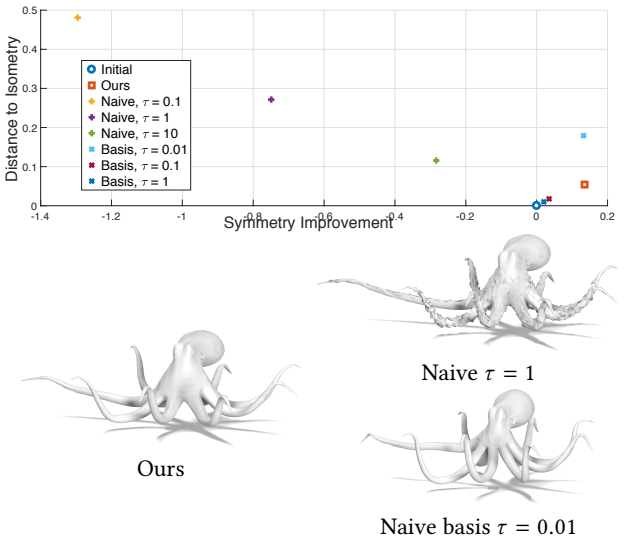


Fig. 26. Comparison between of our intrinsic symmetrization method and the local method Eq. (13) with (+ markers) and without (× markers) using our reduced basis of deformations. Our method produces a deformation that is more isometric, while making resulting in a more symmetric shape than both the original mesh and the naive methods.