Consistent Functional Cross Field Design for Mesh Quadrangulation

OMRI AZENCOT, Technion – Israel Institute of Technology
ETIENNE CORMAN, LIX, École Polytechnique
MIRELA BEN-CHEN, Technion – Israel Institute of Technology
MAKS OVSJANIKOV, LIX, École Polytechnique

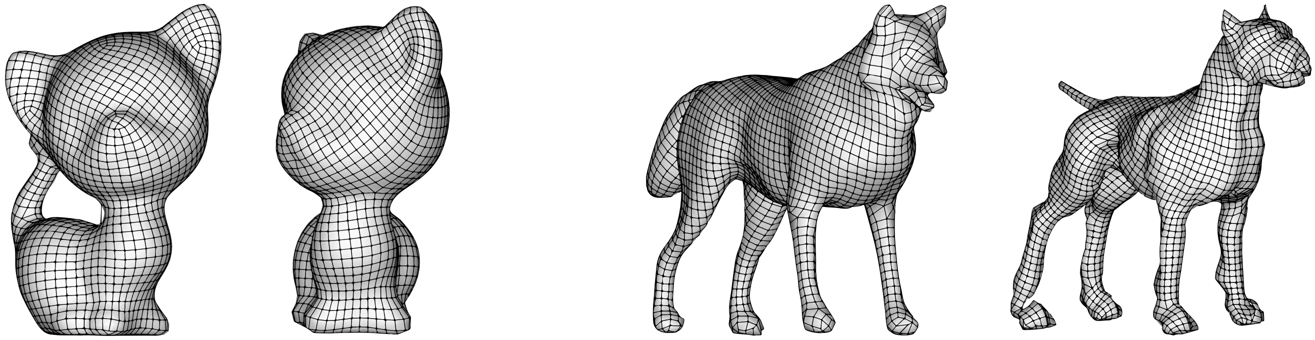

Fig. 1. Our method computes guiding fields on triangle meshes which respect either the underlying symmetry of a single surface (left) or the related correspondence between a pair of shapes (right), while being able to handle arbitrary topology such as the genus one surface on the left. We use these fields to compute approximately consistent quad meshes with off-the-shelf quadrangulation methods.

We propose a novel technique for computing consistent cross fields on a pair of triangle meshes given an input correspondence, which we use as guiding fields for approximately consistent quadrangulations. Unlike the majority of existing methods our approach does not assume that the meshes share the same connectivity or even have the same number of vertices, and furthermore does not place any restrictions on the topology (genus) of the shapes. Importantly, our method is robust with respect to small perturbations of the given correspondence, as it only relies on the transportation of real-valued functions and thus avoids the costly and error-prone estimation of the map differential. Key to this robustness is a novel formulation, which relies on the previously-proposed notion of *power vectors*, and we show how consistency can be enforced without pre-alignment of local basis frames, in which these power vectors are computed. We demonstrate that using the same formulation we can both compute a quadrangulation that would respect a given *symmetry* on the same shape or a map across a pair of shapes. We provide quantitative and qualitative comparison of our method with several baselines and show that it both provides more accurate results and allows to handle more general cases than existing techniques.

CCS Concepts: • **Mathematics of computing → Discrete optimization**; • **Computing methodologies → Shape analysis**;

Additional Key Words and Phrases: quad remeshing, correspondence, consistent remeshing, cross field design

This work was supported in part by the Marie-Curie CIG-334283, a CNRS chaire d'excellence, chaire Jean Marjoulet from Ecole Polytechnique, FUI project TANDEM 2, a grant from the Direction Générale de l'Armement (DGA) and a Google Focused Research Award, European Research Council (ERC starting grant No. 714776 "OPREP"), the Israel Science Foundation (grant No. 699/12), and an Adams Fellowship. We thank Daniele Panozzo and Matthias Vestner for supplying the code for their methods. In addition, we thank Keenan Crane, AIM@Shape, SCAPE, and TOSCA for the models.

ACM Reference format:
Omri Azencot, Etienne Corman, Mirela Ben-Chen, and Maks Ovsjanikov. 2017. Consistent Functional Cross Field Design for Mesh Quadrangulation. *ACM Trans. Graph.* 36, 4, Article 92 (July 2017), 13 pages.
DOI: http://dx.doi.org/10.1145/3072959.3073696

1 INTRODUCTION

Remeshing of triangle meshes to quad meshes is a fundamental task in geometry processing and related domains with applications in shape modeling, texture synthesis and numerical simulation, to name a few. In many cases, quad remeshing is jointly applied to several shapes and when their correspondences are given, the results are frequently required to be *consistent* with respect to those mappings. For instance, the quad mesh which models an animated character should be aligned to the underlying deformation modes [Marcias et al. 2013]. Similarly, on a single shape which exhibits symmetry, a symmetric quadrangular mesh is often preferred [Panozzo et al. 2012]. The goal of this paper is to propose a robust, unified framework for approximately consistent quad remeshing which is applicable to a single shape or a pair of shapes, without assumptions on the mesh connectivity or shape topology.

To date, there exist several automatic methods for generating quadrangular surfaces from triangle meshes. A common approach, which we will also follow in our paper, uses a guiding field within a parametrization-based method. Namely, remeshing is achieved by designing a smooth *cross field* that accounts for local features, followed by an optimization part which seeks a parametrization whose gradients are aligned with the computed field. Quadrangulation is then performed in the parameter domain, where correct stitching of isolines is maintained along cut graphs [Bommes et al. 2013]. In this context, our algorithm produces a set of consistent cross fields, which are used as input to previous remeshing machinery [Bommes et al. 2009; Ebke et al. 2013]. Namely, quadrangulation is computed on each mesh separately, and thus we obtain only *approximate* consistency of quads.

One option for designing smooth cross fields is to encode the angle with respect to a local basis per triangle. The goal is then to minimize the squared difference of these angles along edges, while allowing for integer period jumps [Ray et al. 2008]. Unfortunately, the resulting mixed-integer problem is non-convex and achieving a global optimum is challenging in practice. To rectify this, using trigonometric periodic functions on the angles multiplied by 4 allows to avoid integer variables altogether, see e.g., [Ray et al. 2009], at the cost of introducing pointwise unit-length constraints. Equivalently, in the complex-valued representation [Knöppel et al. 2013] each cross is encoded using the unique *power vector* obtained by representing the cross directions as complex numbers and taking the 4-th power. Further, dropping the pointwise unit-length constraints yields a convex quadratic problem whose global minimum is attained with a single linear solve.

All of the above commonly-used cross field design approaches depend on a choice of local basis (frame) per triangle. In many cases, this basis dependency does not pose any practical challenges. However, when consistency is needed, computing transformations which align these basis vectors across shapes is essential in order to faithfully compare the measured angles. For instance, for meshes with different connectivities, a triangle is not necessarily mapped to a single triangle, and thus several basis vectors must be taken into account. One of the main advantages of our approach is that we formulate the consistency constraints in terms which are *invariant* to the local basis. This novel change greatly simplifies the problem since the basis vectors can be chosen arbitrarily on each shape.

To enforce consistency of cross fields between two shapes, scalars or vectors need to be mapped and compared using the input map. Therefore, the quality of the map and map differential are of crucial importance to achieve good quadrangulation results. However, computing acceptable approximations of these objects is a hard problem in itself, making the entire remeshing pipeline highly dependent and potentially sensitive to high frequency noise in the given correspondences. In our framework, we relax this constraint by assuming that only functional correspondences are given. Functional maps [Ovsjanikov et al. 2012] provide robust means to encode mappings between surfaces by putting in correspondence their function spaces. We pose the consistency requirements solely in the functional language, which allows us to apply our machinery to any shapes for which functional mappings are available. This includes both functional correspondences obtained via a pull-back with respect to a given pointwise map (thus represented in the full basis), and functional maps computed automatically and represented in a reduced spectral basis. An advantage of our formulation is that it allows a separation of the involved components. Namely, the smoothness and alignment constraints are high-dimensional but sparse, whereas the consistency terms are either high-dimensional and sparse or low-dimensional and dense. In both cases, this separation leads to a structured Hessian of the minimized energy, allowing us to employ efficient optimization techniques.

In this paper, we suggest an effective methodology to design consistent cross fields for the purpose of compatible mesh quadrangulation. Thanks to our functional approach, the obtained machinery is similar regardless of whether a single symmetric shape or two shapes are used. Moreover, unlike most previous techniques, such as [Marcias et al. 2013], we place no restriction on the connectivity of the triangle meshes, and further can handle shapes with arbitrary topology. To summarize, our main contributions include: the invariance of the proposed method with respect to a local basis, the ability to design fine details separately on each mesh while requiring consistency only in a low-dimensional space, and the ability to handle arbitrary meshes. We also demonstrate that our method is simple and robust, in large part due to its ability to avoid the potentially difficult and error-prone step of computing a map differential, and at the same time scalable, as it can accommodate functional correspondences represented in a reduced basis. To achieve these goals, we formulate consistent cross field design via a simple, global quadratic energy minimization problem which we efficiently solve by evaluating the action of the Hessian on a general vector.

2 RELATED WORK

Quadrangular remeshing is a challenging problem, and in the last few years there has been a surge of research in this direction. We refer the reader to recent reviews for a general overview of quadrangulation methods [Bommes et al. 2013] and direction field design [Vaxman et al. 2016] on a single shape, and focus our literature review on *joint* design of cross fields and quadrangular meshes.

Perhaps closest to our approach is the Functional Vector Field work [Azencot et al. 2013] where joint design of smooth *vector fields* is formulated in the functional framework. The optimization there is convex, yet the vector fields need to be represented in a low dimensional basis, which is computed using the eigenfunctions of the Hodge Laplacian. We generalize this approach to cross fields, by representing vector fields in a local frame per face, thus avoiding the need for a low dimensional basis, and formulating a functional consistency constraint which is invariant to this choice of frame.

One of the first approaches to joint quad mesh design was presented by Yao et al. [2009]. There, the user sketched compatible skeletons which were used to generate compatible base meshes, from which compatible quadrangulations were extracted. Unfortunately, this approach requires extensive manual input, and affords little control on the quality and smoothness of the resulting quads. Later approaches to interactive design of quadrangular meshes were based on learning quad templates from examples [Marcias et al. 2015; Tierny et al. 2011], with the goal of computing quad meshes which are *approximately* consistent with quadrangular meshes designed by artists. These methods are local, as they rely on segmenting the input into disk-like patches, which may yield sub-optimal results.

The more general problem of computing consistent or approximately consistent quad meshes jointly on a pair or a collection of shapes with respect to an input correspondence has only been addressed by a few methods so far. Given a collection of shapes in correspondence, Meng et al. [2016] co-extract compatible feature lines, and then design cross fields independently for each shape, using the feature lines as alignment constraints. In the following step, they co-design a compatible cut graph, and then align all the shapes in a common parameter domain. However, since the cross fields are designed independently, the correspondence of feature-less regions is not taken into account. Alternatively, Marcias et al. [2013] take as input a set of shapes with compatible triangulations, use the principal directions of the deformation gradient as alignment constraints for designing a single cross field on one of the shapes, extract from it a quad mesh and then propagate it to the rest of the sequence. The case where the connectivities of the triangle meshes are different, is not handled in that work.

These two approaches highlight the main challenge of cross field-guided compatible quad remeshing: transporting the cross fields across meshes. The first approach avoids this issue by designing each cross field separately, whereas the second approach uses a high quality triangle-to-triangle map to transport vectorial information. In an attempt to address this challenge in the context of symmetry-aware cross field design, Panozzo et al. [2012] use a *fuzzy* symmetry map, which averages the contribution of the transported cross field from the neighborhood of a few triangles. While achieving excellent results in some cases, this approach has some important limitations. First, it requires the computation of a high-quality symmetry map, especially tailored to their approach. As is shown in [Panozzo et al. 2012], when using other symmetry maps, the results can be sub-optimal. This is a practical limitation, as their proposed symmetry computation method does not handle, for example, high genus intrinsic symmetries. Second, their algorithm uses hard constraints to align the cross field with the symmetry line. This constraint prevents singularities from appearing on the symmetry line, unnecessarily limiting the space of feasible cross fields. Furthermore, a high quality symmetry line, which might be challenging to compute, is required for this constraint. Finally, the formulation provided there is not in the form of a global optimization problem, and the optimality of the solution under their proposed error metric is not guaranteed.

Our method overcomes the limitations exhibited by previous approaches, as it is robust to the input map, can be applied to meshes with different triangulations and to symmetric meshes without requiring the computation of the symmetry line, and is formulated as a convex quadratic optimization problem whose global optimum is efficient to compute.

3 OVERVIEW AND BACKGROUND

Given a shape along with a self-map (e.g., associated with a symmetry) or a pair of shapes with maps between them our goal is to produce quadrangulations that would be consistent with respect to the input map. To this end, we first design consistent *cross fields*, and then use existing methods to extract quadrangulations from them. Thus, the main focus of our work is to devise a robust method for consistent cross-field design, and we present quad-remeshing as the main target application, among many possible others.

Our main contribution is a novel cross field *consistency energy* (Section 4.3) which we combine with existing smoothness and curvature directions alignment energies into a global convex quadratic optimization problem. For the method to be widely applicable, the consistency energy should be robust to imperfections in the input map. Thus, instead of transporting cross fields with estimated map differentials, which are often noisy for imperfect input maps, we formulate consistency in terms of *scalar functions* and use *composition* with the input map for transport.

Cross fields can be represented discretely in a few ways (see e.g., [Vaxman et al. 2016, Sec 5]), which affects the way the difference between two crosses is measured. Since each cross is composed of a set of 4 indistinguishable vectors, any comparison between two crosses should be invariant to reordering of the sets. A common way to handle this is to define a local basis, represent the cross as a complex number in this basis, and then compute its *power vector*, namely the 4-th complex power of this vector [Knöppel et al. 2013, Fig. 5]. As the power vector is unique for each cross, power vectors can be compared directly in order to compare crosses.

This link between cross fields and their corresponding power vector fields hints at the possibility to leverage techniques used for robust vector field transport [Azencot et al. 2013] for cross field transport. However, one caveat is that smoothness of power vector fields is measured differently than smoothness of vector fields, therefore a low dimensional basis of smooth vector fields can no longer be used for the representation. Furthermore, the power vector fields are dependent on the local basis in which they were computed. We show how this dependence can be eliminated, and use this insight to formulate the consistency energy.

We efficiently solve the resulting optimization problem, then normalize the output power fields and convert them to their associated cross fields. To extract the quadrangulations, we feed the resulting cross fields to a Mixed Integer Quadrangulation (MIQ) [Bommes et al. 2009] implementation that computes parametrization functions whose gradients align with the directions of the cross fields. Finally, the meshes and the parametrizations are given as input to a Quad Extraction (QEx) [Ebke et al. 2013] implementation which robustly extracts the quad meshes associated with the parametrizations. We use the implementations of MIQ and QEx directly and thus we omit further discussion on these methods, and refer the interested reader to the respective papers for additional information. Technical details such as the actual code packages and the parameters we used are described in Section 6.

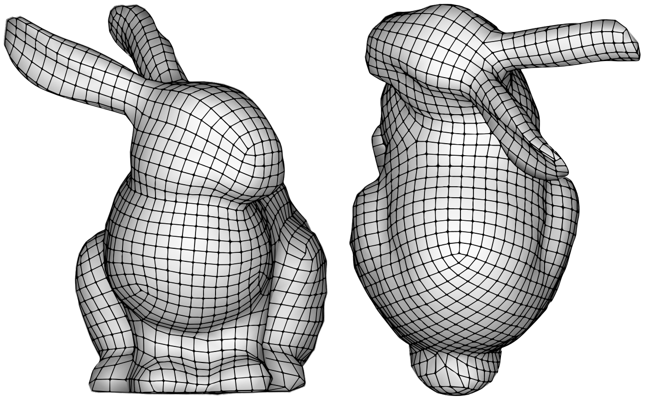

Fig. 2. A quad mesh generated with our method using $k = 100$ eigenfunctions on the intrinsically symmetric bunny model.

We emphasize that we optimize for a consistent cross field and thus the quadrangulation is only approximately compatible in practice. Indeed, we concentrate specifically on the design of consistent cross-fields and our method is not intended to provide guarantees about the quality of the quad final meshes. Nevertheless, in practice, we achieve highly-consistent quad meshes, as can be seen in Fig. 2. The bunny model exhibits an intrinsic symmetry (the head is rotated), which makes the stationary line non-trivial, and yet, our method produces a visually appealing quasi-symmetric quad mesh.

In the following two sections, we describe the design of consistent cross-fields first with respect to a symmetry on a single mesh (Section 4) and then with respect to a pair of meshes with a (functional) map between them (Section 5). We then present results obtained using our approach, by focusing on joint quadrangulation as a principal potential application.

4 SELF-CONSISTENT CROSS FIELD DESIGN

We assume to be given an orientable manifold triangle mesh M with vertex set $\mathcal{V}$, edge set $\mathcal{E}$ and face set $\mathcal{F}$. Vector fields as well as cross fields are piecewise-constant on faces in our setup. Namely, per triangle, a vector is encoded using 2 numbers with respect to a local basis $(b, b^{\perp})$ and thus both the cross field x and its power field y can be expressed as vectors in $\mathbb{R}^{2|\mathcal{F}|}$, where $|\mathcal{F}|$ is the number of faces. To compute the power vector corresponding to a cross in a given face, we take an arbitrary vector of the given 4, compute the angle θ it makes with b, and the resulting power vector in this face is the unit length vector in the 4θ direction. Next, we describe the energy terms we use to design the power field y.

4.1 Smoothness

Following previous work, we use *Dirichlet's energy* which is defined via the *covariant derivative* of power fields, to quantify how much y changes across the edges of the mesh. Integrating the squared norm of this measure over the surface leads to the following smoothness energy term:

$$E_s = \frac{1}{2} \| \operatorname{grad}_p y \|_M^2 = \frac{1}{2} y^T \operatorname{grad}_p^T G_{\mathcal{E}} \operatorname{grad}_p y \,, \qquad (1)$$

where $G_{\mathcal{E}} \in \mathbb{R}^{2|\mathcal{E}| \times 2|\mathcal{E}|}$ is a diagonal matrix which encodes the barycentric mass of edges, and $\operatorname{grad}_p \in \mathbb{R}^{2|\mathcal{E}| \times 2|\mathcal{F}|}$ is the covariant derivative, also referred to as the discrete Levi-Civita connection, modified to account for taking the 4-th power, whose construction is given in e.g., [Diamanti et al. 2014, Eq. (3)].

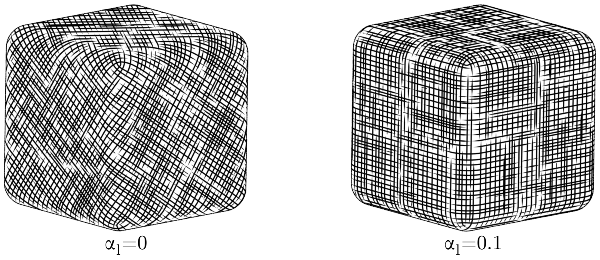

Fig. 3. Optimizing for smooth cross fields which are not aligned (left) or aligned (right) to curvature directions produces equally smooth cross fields, where the right field better respects the underlying geometry.

4.2 Alignment to input directions

In many situations, the designed field will be required to align with certain directions, where the principal curvature directions are a natural choice for quad remeshing. Given an input cross field, we compute its associated power vector field $w \in \mathbb{R}^{2|\mathcal{F}|}$, and arrive at the straightforward alignment term:

$$E_l = \frac{1}{2} \| S(y - w) \|_M^2 = \frac{1}{2} (y - w)^T S^T G_{\mathcal{F}} \, S(y - w) \,, \qquad (2)$$

where $G_{\mathcal{F}} \in \mathbb{R}^{2|\mathcal{F}| \times 2|\mathcal{F}|}$ is the diagonal mass matrix for the faces, and $S \in \mathbb{R}^{2|\mathcal{F}| \times 2|\mathcal{F}|}$ is a diagonal matrix of weights given by the user, indicating the relative importance of the alignment constraints. For instance, when the principal curvature directions are used for alignment, S is usually a measure of the anisotropy of the curvature. In Fig. 3, we show that without alignment constraints (left), the smoothest cross field may not necessarily follow the curvature directions, whereas even a modest alignment requirement yields a smooth field which is parallel to the cube's edges (right).

4.3 Consistency

Vector fields. A vector field is *consistent* with respect to a self-map $\phi : M \to M$, if for any point $q \in M$, the following equation holds:

$$\mathrm{d}\phi(v(q)) = v(\phi(q)) \,. \qquad (3)$$

Namely, points which match under the mapping should be equipped with identical vectors, via the transformation of the tangent spaces given by the *map differential* $\mathrm{d}\phi$. Two major challenges are related to enforcing the above equation in practice. Firstly, when ϕ is approximate, enforcing Eq. (3) to too many outliers may erroneously affect the result. Secondly, computing the map differential $\mathrm{d}\phi$ is a non-trivial and potentially unstable task, especially in the presence of noisy maps. See Section 7 for further details and comparisons.

Instead of working directly with Eq. (3), vector fields can also be seen as *derivations* [Morita 2001, pg. 37]. That is, we can apply Eq. (3) to a real-valued function $f : M \to \mathbb{R}$ and obtain the following consistency constraint:

$$v(f) \circ \phi = v(f \circ \phi) \,, \qquad (4)$$

where $v(f) = \langle v, \operatorname{grad} f \rangle$ is the pointwise directional derivative, and $\circ$ denotes composition with a map. It is a well-known direct consequence of the chain rule [Morita 2001, Eq. (1.14)] that for a fixed tangent vector field v, Eq. (4) is satisfied for all smooth functions f, if and only Eq. (3) holds. Note that for a fixed f and ϕ, Eq. (4) is *linear* in v, meaning that it can be optimized, for example by solving a linear least squares system.

Power vector fields. Cross fields are only smooth up to rotation by integer multiples of $\pi/2$, and thus Eq. (4) cannot be applied directly without incorporating integer constraints. With a local basis $(b, b^{\perp})$ per face, the four vectors of the cross can be mapped to a single vector per face, by taking the 4-th complex power of the vector with respect to the basis b. This *power vector field* is smooth, if the change of basis between neighboring faces is taken into account. However, power vector fields are not canonical, as they depend on the choice of local basis $(b, b^{\perp})$ per face. Thus, comparing two power vectors at a given point is only meaningful if they were defined *with respect to the same local basis.* In other words, it is not enough for

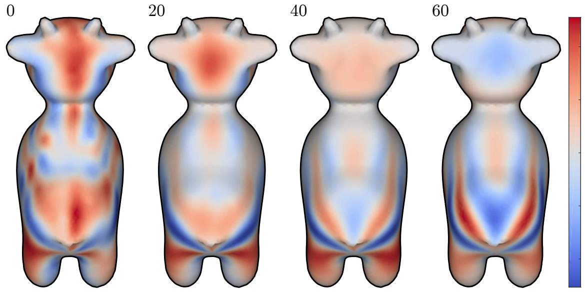

Fig. 4. We iteratively optimize for a consistent cross field whose action on a fixed function produces a symmetric result. In this example, y's action at iteration 0 is not highly symmetric, but it quickly improves during the iterations 20, 40 and 60.

the power vector field to satisfy Eq. (3) or, equivalently Eq. (4) for all f, to guarantee a consistent cross field. Instead, the transformation between the *basis vectors* (at every pair of points q and $\phi(q)$) should also be accounted for.

To overcome this difficulty, one can try to treat the basis b as a smooth vector field, and design it simultaneously with the power vector field y, such that b also fulfills the consistency condition in Eq. (3). However, this yields an optimization problem which is twice as large in the number of variables and constraints, and which is non-linear because of the dependency between the basis vector field and the power field y. Instead, we show how to remove the basis dependency altogether using the following observation:

PROPOSITION 1. *Given a cross field x and an arbitrary point $q \in M$, we compute the associated power vectors y_1 and y_2 at q using two different basis vectors b_1 and b_2, respectively. Then, for any real-valued function f, the following relation holds*

$$\langle y_1, (\mathrm{grad}\, f)_{1,p} \rangle = \langle y_2, (\mathrm{grad}\, f)_{2,p} \rangle \ ,$$

where $(\mathrm{grad}\, f)_{i,p}$ is the power vector of $(\mathrm{grad}\, f)$ at q in the basis b_i.

In other words, the inner product of two power vectors defined with respect to the same local basis is invariant to the choice of basis. Intuitively, the inner product between two power vectors encodes the angle between *the underlying crosses* and is thus basis independent. See Appendix A for the straightforward proof. Thus, in the case of power fields, we modify Eq. (4) and consider instead:

$$\langle y, (\mathrm{grad}\, f)_p \rangle \circ \phi = \langle y, (\mathrm{grad}(f \circ \phi))_p \rangle \ . \tag{5}$$

Note that the two sides of the equation are computed at different tangent spaces, of the symmetric points q and $\phi(q)$, with respect to arbitrary basis vectors. The comparison between these values is meaningful due to the proposition above.

Intuitively, for the constraint to hold, the function $\langle y, (\mathrm{grad}\, f)_p \rangle$ should be symmetric under the map ϕ. Fig. 4 visualizes this function on the surface during our iterative optimization process, described below. As the optimization proceeds the function becomes more symmetric, and thus the consistency error is reduced. We note that unlike Eq. (4), Eq. (5) is *non-linear* in the function f. However, both of these equations are linear in y, which allows us to use this equation directly to enforce consistency of a cross field with respect to a given map ϕ, with an arbitrary local basis.

Discretization. In practice we work with functions represented in a chosen functional basis B, with the two most commonly used bases in our setting being either the indicator (hat) basis at the vertices, or a multiscale low-dimensional basis $B \in \mathbb{R}^{|\mathcal{V}| \times k}$ such as the Laplace–Beltrami eigenfunctions. In that case, functions are represented as vectors of size k, where $k < 300$ in all our experiments. We are given as input a functional map, which maps real-valued functions represented in the basis B to other such functions, and we represent it as a matrix C of size $k \times k$.

The operator grad is the standard gradient operator for functions in the piecewise linear hat basis, and thus we use the transformations $\tilde{f} = Bf$ and $f = B^+ \tilde{f}$ between functions $f \in \mathbb{R}^k$ in the basis B and functions $\tilde{f} \in \mathbb{R}^{|\mathcal{V}|}$ in the hat basis, where B^+ is the pseudo-inverse of B. We further use the matrix $I_{\mathcal{V}}^{\mathcal{F}} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{F}|}$ to interpolate face-wise values to vertex-wise values.

The function $\langle y, (\mathrm{grad}\, \tilde{f})_p \rangle \in \mathbb{R}^{|\mathcal{F}|}$ is linear in y, and thus its computation can be encoded as a matrix-vector product. We interpolate the face-wise values of the inner product to the vertices and define $\tilde{D}(\tilde{f}) \in \mathbb{R}^{|\mathcal{V}| \times 2|\mathcal{F}|}$ such that $\tilde{D}(\tilde{f}) \cdot y = I_{\mathcal{V}}^{\mathcal{F}} \langle y, (\mathrm{grad}\, \tilde{f})_p \rangle$. To use functions f given in a reduced basis B we define the reduced operator $D(f) \in \mathbb{R}^{k \times 2|\mathcal{F}|}$ as: $D(f) = B^+ \tilde{D}(Bf)$.

Finally, using these operators we enforce our consistency rule (5) on a subset of m functions f_i given in the basis B, and arrive at the following novel constraint:

$$E_c = \frac{1}{2} \sum_{i=1}^{m} \| C\, D(f_i) \cdot y - D(C \cdot f_i) \cdot y \|^2 \ . \tag{6}$$

Discussion. While in the above formulation we describe the use of functional maps encoded in a reduced basis of size k, we stress that our framework can be easily applied in the particular case when a dense (vertex to vertex) or precise (vertex to point on face) map is known. In this setting, B is the identity matrix of size $|\mathcal{V}|$ and $C \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is a *sparse* matrix encoding the dense correspondence or using three values per row for the precise (inside the face) map. Similarly, $D(f) = \tilde{D}(f)$. For all the figures we provide in this paper, we show the results obtained using the *reduced* functional map, unless noted otherwise (see e.g., Figs. 10, 11 and 15).

4.4 Energy minimization

We combine the above design constraints into a single minimization problem. Linear blending of energy terms is controlled via two parameters α_c and α_l, both in the range $[0, 1]$. Overall, the power field that we use to generate the quad mesh with is the minimizer

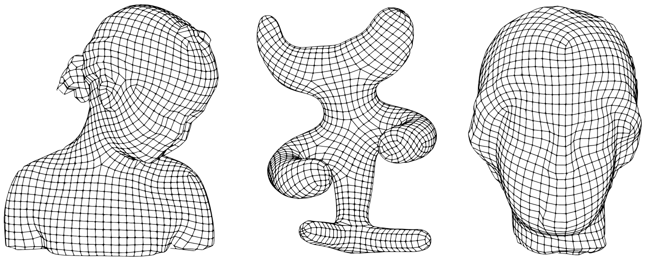

Fig. 5. Quadrangulation results for models with intrinsic symmetries.

of the problem:

$$\underset{y}{\operatorname{argmin}}\,(1-\alpha_l)[(1-\alpha_c)\,E_s+\alpha_c\,E_c]+\alpha_l\,E_l\;. \tag{7}$$

Notice that the above problem is *quadratic* in y since we omit the unit length constraint on y as was done in [Knöppel et al. 2013]. Consequently, we have a linear gradient and a constant Hessian. In Appendix B, we discuss how to efficiently solve the above problem using a standard optimization toolbox. In particular, we show that although the Hessian of our energy is large and dense, its product with a given vector can be computed efficiently. In Fig. 5, we present a few examples of models with intrinsic symmetries and the quadrangulation result we obtained using only $k = 100$ eigenfunctions.

4.5 Relation to other functional approaches.

Functional maps. Given a point-to-point map ϕ, the linear operator that maps any real-valued function f to the pull-back $f \circ \phi$ was denoted by Ovsjanikov et al. [2012] as the functional representation of ϕ or a *functional map*. Several follow up works [Ovsjanikov et al. 2016] have since then observed that it can be easier to frame problems using functional rather than pointwise maps. Our work follows this theme, as in order to enforce Eq. (5), it is only necessary to have access to the functional map $f \circ \phi$. As we show below, this greatly simplifies the computations and at the same time extends the applicability of the resulting algorithm. This is because our formulation avoids not only the estimation of the map differential, but does not even require the knowledge of a precise point-to-point map, the estimation of which from a functional map can be challenging [Rodolà et al. 2015], and is required by some state-of-the-art mapping methods [Litany et al. 2016; Ovsjanikov et al. 2016].

Functional vector fields. Joint design of smooth vector fields has been done in [Azencot et al. 2013] by leveraging Eq. (4). There, to avoid working with a local basis per face, vector fields have been represented by matrices, and Eq. (4) was implemented as a commutativity constraint. However, to reconstruct the face-wise vector field from its matrix representation, and to enforce smoothness on the resulting vector field, Azencot et al. worked in a *low dimensional basis* of tangent vector fields computed as the eigenfunctions of the Hodge Laplacian. To generalize their approach to cross fields, one would need to modify the basis to be able to represent smooth cross fields, and in addition modify the consistency term to take into account the local basis in which the cross-field was computed. We avoid these issues by working directly with the face-based vector fields as the variables in a basis-invariant formulation. Note that simply designing a power vector field using the functional vector field machinery would not yield the smoothest cross-field, as the singularities that arise are different (see Fig. 6, and also [Ray et al. 2006, Fig. 8]).

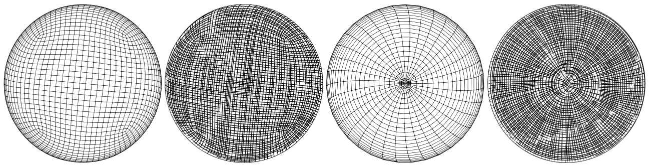

Fig. 6. The method of Azencot et al. [2013] can not be directly used to design smooth cross fields. For comparison, we show the quad mesh computed from the smoothest cross field (left and middle left), and from the smoothest vector field converted to a cross field treated as a power field with a smooth local basis (middle right and right).

5 CONSISTENT FIELD DESIGN ON TWO SHAPES

To extend the model we proposed in Section 4, we consider the following scenario. Given a pair of triangle meshes M_1 and M_2, possibly with *different* vertex and face sets, our pipeline requires as input the functional maps C_{12} and C_{21}, which map functions on M_1 to functions on M_2 and vice versa. One of the advantages of our approach to consistent cross field design, is that it naturally generalizes from the case of a single shape to a pair of shapes. Indeed, the new smoothness and alignment components are extremely similar to the former case, whereas the main change is in the consistency term where we now optimize for two power fields instead of one. Our objective is to optimize for fields y_1 on M_1 and y_2 on M_2 such that the following energy terms are minimized. For example, we show in Fig. 7 the different results we obtain with (bottom) and without (top) our consistency condition.

In this setting, given two shapes, we simply add together the smoothness and alignment constraints for each y_i. Formally,

$$E_s = \frac{1}{2}\|\operatorname{grad}_p y_1\|^2_{M_1} + \frac{1}{2}\|\operatorname{grad}_p y_2\|^2_{M_2}\;, \tag{8}$$

$$E_l = \frac{1}{2}\|S_1(y_1 - w_1)\|^2_{M_1} + \frac{1}{2}\|S_2(y_2 - w_2)\|^2_{M_2}\;, \tag{9}$$

where w_i and S_i are typically the curvature directions and their weights on M_i. To avoid clutter of notation, we uniformly use grad_p for the covariant derivatives on both of the meshes, in cases where no confusion might arise. Notice that while being stacked jointly, Eqs. (8) and (9) are independent of the relations between M_1 and M_2, i.e., the associated Hessians are *block-diagonal*.

To develop the consistency rule for a pair of shapes, we recall the geometric meaning of our constraint on a single mesh. Namely, in the former case we required that for a given function, taking the appropriate inner product with y and the functional map (pull-back) should commute (Eq. (5)). For two shapes, we have a similar scenario, being different in that the mapped versions are on the other mesh, where before we had only one surface. In addition, to avoid favoring a particular mapping direction, we symmetrize our constraint by adding an analogous term in the other direction, and thus we need both $C_{12} \in \mathbb{R}^{k_2 \times k_1}$ and $C_{21} \in \mathbb{R}^{k_1 \times k_2}$. We obtain the following consistency condition:

$$\begin{aligned} E_c = &\frac{1}{2}\sum_{i=1}^{m}\|C_{21}\,D(f_{i,2})\cdot y_2 - D(C_{21}\cdot f_{i,2})\cdot y_1\|^2 \\ &+ \frac{1}{2}\sum_{i=1}^{m}\|C_{12}\,D(f_{i,1})\cdot y_1 - D(C_{12}\cdot f_{i,1})\cdot y_2\|^2\;, \end{aligned} \tag{10}$$

where $\{f_{i,1}\}$ and $\{f_{i,2}\}$ are sets of m functions chosen arbitrarily on M_1 and M_2, respectively. Again, we remind that our operators are given in some pre-calculated functional basis. For instance, C_{12} maps a function f_1 represented in the basis $B_1 \in \mathbb{R}^{|\mathcal{V}_1| \times k_1}$ to a function $f_2 = C_{12} \cdot f_1$ given in the basis $B_2 \in \mathbb{R}^{|\mathcal{V}_2| \times k_2}$.

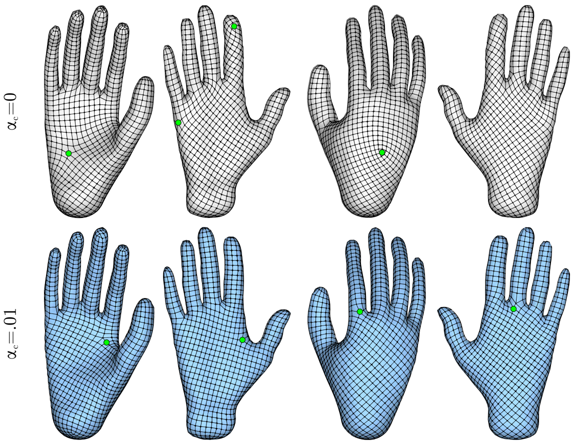

Fig. 7. Curvature information can sometimes lead to quasi-consistent results even without consistency $\alpha_c = 0$ (top row). However, we show that facilitating our compatibility condition $\alpha_c = .01$ with the precise mapping from BIM represented using a functional map of size $k = 50$, produces more consistent quad meshes (bottom row).

Finally, we gather the above energy terms into a single problem, where we optimize for power fields y_1 and y_2. Notice that, as in the case of a single shape, while y_i are encoded in a specific *local* basis in every tangent plane of every point on shape $i \in \{1, 2\}$, our formulation is invariant to the choice of these bases. We employ the same weighting parameters as before, and arrive at our final optimization energy:

$$\underset{y_1, y_2}{\operatorname{argmin}} \, (1 - \alpha_l)[(1 - \alpha_c) \, E_s + \alpha_c \, E_c] + \alpha_l \, E_l \; . \tag{11}$$

Discussion. We point out that our approach can be extended to the case of shape *collections* in a straightforward way. That is, smoothness and alignment constraints are simply stacked as in Eqs. (8) and (9), and consistency could be achieved by either enforcing Eq. (10) between each of the shapes and a template mesh or by exhaustively enforcing it between all possible pairs. One challenge involved in taking this approach is that it might be not practical to solve the obtained problem when the collection is large. As we were focused on developing the cases of a single shape and a pair of shapes in this paper, we leave further investigation of consistent quadrangulation of shape sets for future work.

6 IMPLEMENTATION DETAILS

We implemented our method using MATLAB and tested it on a Intel Xeon 3.20GHz processor with 32GB RAM. The optimization problems we consider in Eqs. (7) and (11) could be re-arranged as standard quadratic programming problems, composed of sparse components, E_s and E_l, and a dense element, E_c (see Appendix B). Thus, we were able to use MATLAB's quadprog optimization tool with a user-handle to compute Hv, where H is the Hessian and v is a vector, in order to avoid storing the full dense Hessian. The initial solution was the smoothest power field ($\alpha_c = 0$ and $\alpha_l = 0$) in all our tests. For problems with 5k/11k/20k/27k/50k vertices, the power field design part converges in 5/7/13/34/90 seconds with a point to point mapping or in 10/18/68/90/169 seconds using a reduced basis of size $k = 100$, respectively.

For the functional basis B, we took the first k eigenfunctions ordered by their eigenvalues, starting with the smallest one. Similarly, we use the first $m = \min(k_1 - 1, k_2 - 1)$ eigenfunctions excluding the constant one for the test functions $\{f_i\}$ which appear in Eqs. (6) and (10). In practice, we test against the power of the gradient $(\text{grad}\, f_i)_p$ weighted by λ_i^{-1}, where λ_i is the associated eigenvalue. To generate point to point mappings, we used implementations of Blended Intrinsic Maps (BIM) [Kim et al. 2011] without landmark correspondences, seamless surface mappings [Aigerman et al. 2015] and the descriptor based pipeline from [Ovsjanikov et al. 2012] with landmark constraints. For the computation of functional maps in a reduced basis we used a pipeline that combined descriptor and sparse landmark correspondences by adapting the approaches of [Ovsjanikov et al. 2012] and [Pokrass et al. 2013].

In all of our experiments, the power version of the principal curvature directions is used for the alignment constraints (Eq. 2). To this end, we implemented the method proposed in [Rusinkiewicz 2004], where the weights are computed per triangle j by $S(j) = |\kappa_1 - \kappa_2|^2$ with κ_i the extremal curvature values, and we clamp values below .1 to zero. Once the cross fields are computed, we use it as input for the implementation of MIQ provided in libigl [Jacobson et al. 2013], and we then feed the resulting parametrization to the implementation of QEx provided by the authors [Ebke et al. 2013] to obtain a quadrangular mesh. Both, MIQ and QEx, were used with default parameters in our tests.

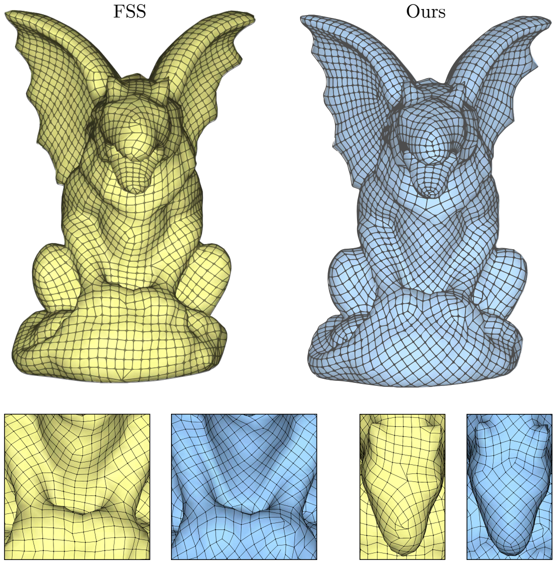

Fig. 8. The approach of Panozzo et al. [2012] constrains the field to be aligned with the stationary line (yellow). Thus, the space of possible minimizers is significantly smaller, yielding sub-optimal results on the chest and nose of the shape (zoomed-in areas). In contrast, our method allows for general cross fields which exhibit intricate behavior along the symmetry line (blue). Consequently, our output better respects the involved geometry, while achieving lower error values (see rightmost column in Figs. 10 and 11).

7 EVALUATION AND RESULTS

7.1 Comparison with FSS

We compare our symmetric quadrangulation results with the state-of-the-art method of Fields on Symmetric Surfaces (FSS) by Panozzo et al. [2012]. In all of the following experiments, for computing the FSS results we used the cross field output data provided by the authors. For generating our cross field, we used as input the symmetry-map generated by their intrinsic symmetry computation method (using code provided by the authors), which results in a vertex-to-point in face mapping, and used the full hat basis unless otherwise noted. For both methods, we generated a quad mesh from the cross field using MIQ and QEx, using the same parameters for both approaches.

Behavior near the symmetry line. As discussed in Proposition 6 in [Panozzo et al. 2012], at the symmetry line the cross field should either have a singularity, be aligned with the symmetry line, or form $\pi/4$ angle with the symmetry line. In their approach, the field is forced to align with the symmetry line using hard constraints. However, allowing singularities on the symmetry line, and therefore allowing the cross field to switch between the two configurations (aligned and rotated by $\pi/4$), may increase the overall consistency and smoothness. Instead, we omit this constraint, and solve for the global minimizer of Eq. (7), allowing us to compute quadrangular meshes which are more consistent with respect to singularity point locations and error metrics. In Fig. 8 we show the quad mesh generated by FSS (yellow) and by our approach (blue) using the full map. Note how forcing the quad directions to align with the symmetry line generates noisy quads in the FSS approach, e.g., along the chest and nose of the gargoyle as shown in the zoomed-in figures, whereas our method generates a smoother edge-flow.

Applicability. As mentioned in their paper (see Figure 8 there), FSS requires a high-quality symmetry map, and a corresponding symmetry line. In addition, for computing the map differential, they use the gradients of two functions (one symmetric and one anti-symmetric), which should also be extracted from the map and be of high quality. Our approach, on the other hand, requires only a functional correspondence, which can be given in a reduced or full basis, and can potentially be noisy. Therefore, our approach is applicable to more general shapes and less robust correspondences and as different mapping methods work better in different scenarios, ours general applicability is a clear advantage. For instance, in Fig. 9, we employ the mapping obtained using BIM on a mesh which is particularly challenging as there is a significantly different density of triangles along the stationary line (left). Nevertheless, our method produces a reasonable quad mesh (right) using only $k = 100$ eigenfunctions.

Quantitative comparison. We ran our method on all the models shown in [Panozzo et al. 2012], for which the FSS intrinsic map computation could be used. We measured the consistency error of the resulting cross fields using two metrics: e_{ours} which is closely related to our consistency condition (Fig. 10) and e_{FSS} which is guiding the FSS approach (Fig. 11). The first metric, e_{ours}, is given by

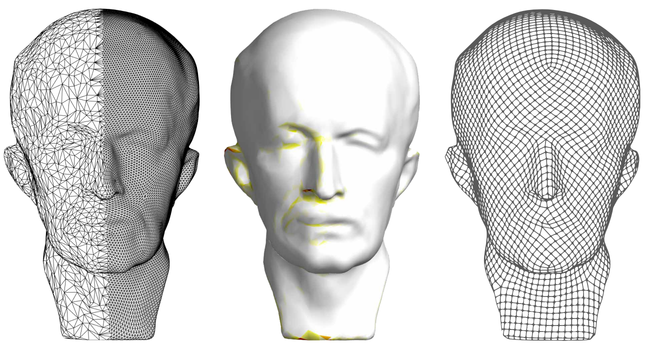

Fig. 9. Robustness to triangulation. (left) We extensively decimated %85 of the vertices in the left part of Max Planck's model, leading to non-symmetric curvature alignment constraints (middle) due to the difference in triangle areas. Nevertheless, our method produces a symmetric cross field whose associated quad mesh is highly consistent (right). Notice that this example is particularly challenging for methods which employ the map differential.

$$e_{\text{ours}} = E_c(m, f_i) + E_c(n, g_i) \ ,$$

where both of the terms are computed using the functional map constructed from the known precise mapping, i.e., $C \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$. In the left term $E_c(m, f_i)$, we use m and f_i as defined in Section 6, and we randomly generated $n = 1000$ vertex indices for which we created hat functions g_i that are used to compute $E_c(n, g_i)$. The second metric is defined as follows.

$$e_{\text{FSS}} = \|x_p - (\text{sym}(x))_p\|_M^2 \ ,$$

where x is the cross field generated by FSS, and sym(x) is the symmetrized version of x, computed by applying the "symmetrization by field transport" step of the FSS algorithm to x. We compare x with its symmetrized version by comparing their power fields, and weigh the errors by the face area, namely $\|y\|_M^2 = y^T G_{\mathcal{F}} y$. If $e_{\text{FSS}} = 0$, it would imply that symmetrizing the cross field x has no effect, and thus x is already exactly symmetric. For both metrics, we show the results for FSS (yellow squares) and for our approach when using the FSS map in the full basis (blue circles) and in the reduced basis (red diamonds). To evaluate the error results together, we consider the *relative* error as it is measured with respect to the value we obtained with our method when using the full basis.

As can be seen in Figs. 10 and 11, our method with the full basis achieves better error results on all of the meshes except for bimba, where the results of the two methods are similar. Note that in these experiments we used the same input including the symmetry map computed by the method in [Panozzo et al. 2012], and only the computation of the cross field is different. Thus, these quantitative results highlight the robustness and accuracy of our functional formulation. In particular, on models where our cross fields switched their behavior along the symmetry line, we gained significant improvement: e.g., of factors 3.3, 4.65, and 9.53 on the models busto, Max Planck and gargoyle, respectively, in our metric e_{ours}. The improvement in e_{FSS} for these models was 1.87, 3.75 and 4.35, respectively. Moreover, we note that our method with a reduced functional basis of size $k < 300$ produced comparable or better results when compared to

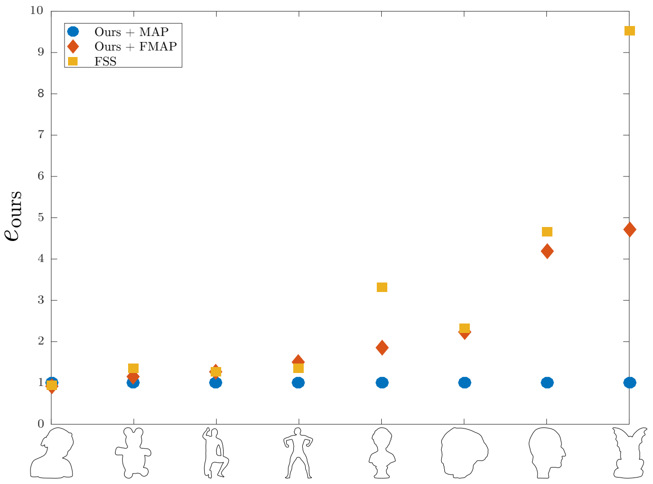

Fig. 10. We compared our method using functional maps given in a full basis (blue circles) or a reduced basis (red diamonds) to FSS [Panozzo et al. 2012]. Using the full basis, our approach achieves superior results on all the models, and in some cases by a large margin, as can be seen in the relative error above. See the text for additional details.

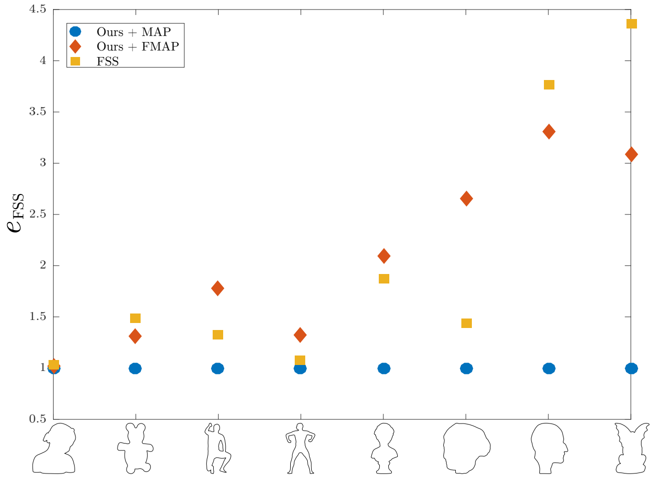

Fig. 11. In addition to the comparison we show in Fig. 10 that uses a modified version of our consistency condition, e_{ours}, we also compute the relative error metric e_{FSS}, which is optimized in FSS. For all the models, we obtain improved error metrics when using the full basis and comparable results for the reduced case. For more details, see the text.

FSS with our metric e_{ours}. However, when measured in e_{FSS}, these cross fields typically generated inferior error results, while being visually plausible as can be seen in Fig. 5 (left and right).

7.2 Comparison with AAQ

We compare our consistent quadrangulation of a pair of meshes with the state-of-the-art method of Animation Aware Quadrangulation (AAQ) by Marcias et al. [2013]. For computing the AAQ results we used the code provided by the authors, and applied it to the two human meshes shown in Fig. 12, using the default parameters. We generated two cross fields, by choosing first the kneeling human as the base mesh (yellow left) and then the standing human as the base mesh (yellow right). Each of these quad meshes should be transported to the second frame to generate exactly consistent quad meshes (we do not show the transported quads). Note, that AAQ can only be applied to meshes with the same triangulation, thus we also used this point-to-point map (in the full basis) as our input and applied our pipeline to generate consistent cross fields (blue). We then generated a quad mesh from the cross fields using MIQ and QEx, using the same parameters for both approaches.

Note, that while the output of AAQ is exactly consistent (as they transport the quads directly), the resulting quad mesh pairs would be very different depending on which mesh is used as the base mesh. Furthermore, our result is qualitatively better for both meshes (as it is both smooth and consistent), even though we only jointly design the cross-fields and quadrangulate separately. In general, the deformation of the triangles between these two meshes is quite large, which, as noted in the AAQ paper, is a challenge for the their method. Further, since AAQ only considers the deformation between the meshes, and not the curvature directions explicitly, the results for a pair of meshes are less aligned with the geometry than our results. We do note that AAQ is geared towards the more complex scenario of a collection of meshes, which we do not currently support, however their approach is specifically designed for triangle meshes with the same connectivity, thus is less general than ours in this respect.

7.3 Parameters exploration

Effect of consistency and alignment constraints. To motivate the use of our compatibility constraints, we demonstrate in Fig. 7 the quad meshes we obtain with and without this constraint on a pair of surfaces. Specifically, in the top row, we show that when requiring zero consistency, i.e., $\alpha_c = 0$, the resulting quads are somewhat related, mainly due to curvature information, but the singularities are in different locations (green points). Increasing this parameter to

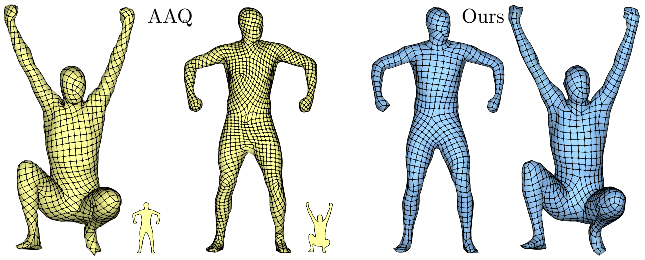

Fig. 12. We compare our consistent quadrangulation of a pair of meshes with AAQ [Marcias et al. 2013] using code supplied by the authors. We computed a quad mesh on each mesh using it as the base mesh (yellow), which should then be transported to the second mesh to yield exactly consistent quadrangulations. For our approach (blue), we used the point-to-point correspondence also given to AAQ. Note that our results are both smooth and consistent, where as while the AAQ results are exactly consistent, they are dependent on the base mesh, and considerably less smooth. See the text for more details.

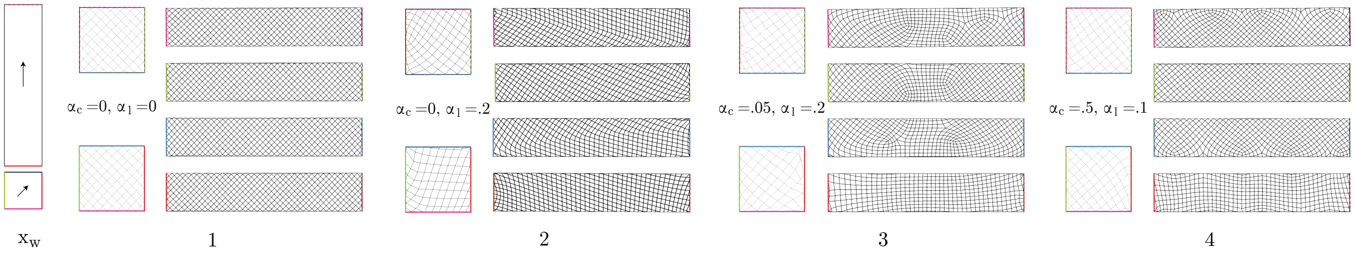
α_c =0, α_l =0
α_c =0, α_l =.2
α_c =.05, α_l =.2
α_c =.5, α_l =.1
x_w
1
2
3
4

Fig. 13. Alignment vs. symmetry on a 3D bar shape. (x_w) The directional constraints are placed on the bottom face and and on one of the sides as marked by the arrows. (1) The minimizer of the Dirichlet's energy yields a smooth and consistent quad structure. Notice that increasing α_c in this case can result in a similar quad mesh or any global rotation of it. (2) However, increasing the alignment constraint, yields a better aligned quadrangulation which is not necessarily consistent. (3) Requiring even a small non-zero value for consistency, leads to an aligned quad mesh which is quasi-consistent in a least squares sense. (4) Finally, taking high values for α_c and α_l produces an aligned and consistent quad mesh.

$\alpha_c = .01$ yields a compelling result, where the isolines and singularity locations are very consistent (bottom row). In addition, we show in Fig. 13 a more thorough evaluation of consistency vs. alignment on a 3D bar model where the parametrization cuts are given by the colored edges. In this example, our results show that the smoothest quad mesh is also consistent (1), whereas requiring alignment to the misaligned constraints (x_w) breaks this natural consistency (2). Nevertheless, prescribing a non-zero solid-to-dashed consistency, yields a quasi-consistent and a highly consistent quadrangulations which are also aligned, as shown in (3) and (4), respectively.

Size of the functional basis. In theory, our consistency rules in Eqs. (6) and (10) should hold for any function. However, these constraints are based on mapping functions between surfaces using a functional map, which is potentially given in a low-dimensional basis. Thus, transferring functions with high-frequencies in this case may result in significant errors due to projection onto the basis. Nevertheless, as our basis is given in terms of a multiscale eigendecomposition of the Laplace–Beltrami operator, we hope to use as few as possible basis elements in our application. In practice, the basis size k determines which functions are well-represented, and in Fig. 14 we try to quantify which k's allow to produce high-quality quad meshes. (left) Using only $k = 10$ eigenfunctions is clearly insufficient as the resulting mesh is only quasi-symmetric, mainly due to curvature information, whereas increasing the basis to size $k = 100$ (middle) produces a highly-consistent mesh. With $k = 5161$, we obtain optimal results.

7.4 Robustness

Noisy point-to-point mappings. A key advantage to working in the functional setup is that it allows to gracefully handle scenarios where approximate or noisy correspondences are given. To evaluate the robustness of our method to inexact mappings in the context of approximately symmetric quad remeshing, we propose the following experiment. The model we use in Fig. 15 is equipped with a *compatible* triangulation, and thus we have the ground-truth mapping ϕ. Using this map, we generate two additional noisy correspondences, ϕ_2 and ϕ_4, where each point is randomly mapped with gaussian weights to the 2-ring and 4-ring neighborhood of its matching point, respectively.

Equipped with this data, we generate symmetric quadrangulations with our method using ϕ, ϕ_2 and ϕ_4 shown in the left, middle and right columns, respectively, where the top row is computed with the full basis and the bottom row uses a reduced basis of size $k = 100$. As can be seen in Fig 15, with perfect information ϕ, the results we obtain are outstanding, exhibiting complex quad structures (top left). However, our outputs are of lesser quality when noisy maps are used, with bent isolines on the head for ϕ_2 (top middle) and non-symmetric stationary line around the chest for ϕ_4 (top right). In contrast, when we use the associated functional maps in a reduced basis, the results we obtain and show at the bottom row reveal comparable consistency quality, regardless of the underlying noise in the mappings. While this result might seem non-intuitive in light of the error values we achieved in graphs 10 and 11, we stress that the intrinsic mappings produced with FSS [Panozzo et al. 2012] are of extremely high quality. However, computing good mappings is a hard problem in the general case, and thus we advocate the use of a reduced basis in cases where inexact data is given.

Pushforward error evaluation. When given a pair of shapes with the same connectivity, we can provide a more accurate measurement of the consistency error related to our computed cross fields. To this end, we facilitate a decimated version (752 vertices) of the SCAPE dataset [Anguelov et al. 2005]. We compute cross fields y_1 and y_2 on the template pose paired with each of the first 50 poses. Then, using the ground-truth map differential, we push the associated x_1 to M_2

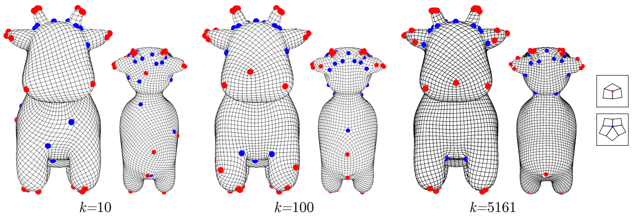

Fig. 14. Effect of changing the functional basis' size. (left) When using only 10 eigenfunctions in B, we show that the resulting quadrangulation is hardly consistent with respect to the existing bilateral symmetry. (middle) Increasing the basis to include 100 elements significantly improves the result. (right) Finally, using the whole spectrum yields nearly perfect results.

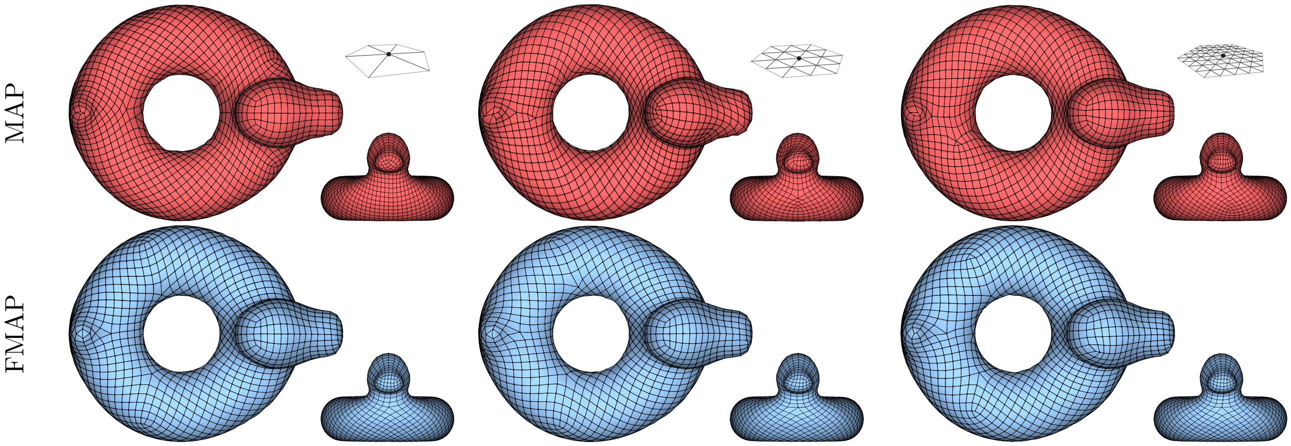

Fig. 15. In this example, we demonstrate the robustness of our method in the presence of imperfect correspondences. We compared the quad meshes we obtain when using maps with deteriorating quality (left to right) in the full basis (top row) and reduced basis (bottom row). While we achieve very good results with the exact mapping (top left), the quad meshes produced with the noisy maps display only quasi-symmetry (top middle and top right). For comparison, employing a small functional map of size $k = 100$, yields consistent quadrangulations in all cases (bottom row). See the text for additional details.

and calculate the error $G_{\mathcal{F}_2}(q)\|x_2(q) - (\mathrm{d}\phi(x_1))(q)\|^2$, for each face $q \in \mathcal{F}_2$. In Fig. 16 we show the resulting sorted error distribution as computed for all of the pairs. The obtained results are consistently within the 10^{-5} range for all pairs, which is reasonable for such coarse triangulations.

Applicability to various mapping methods. In our tests, we use different mapping methods and functional maps. For example, the results in Figs. 2, 5 (left and right), 8, 10 and 11, are based on the intrinsic correspondences generated with FSS. Moreover, we utilized BIM in Figs. 5 (middle), 9, 7 and 14. Example 17 is particularly challenging as it involves non-isometric meshes for which the current state-of-the-art methods produce only approximate maps. Specifically, we used the seamless mapping method [Aigerman et al. 2015], and we generated approximately consistent quad meshes using the full basis (left) and the reduced basis (right). Notice that the resulting quadrangulations are qualitatively similar being slightly more consistent for the full map case.

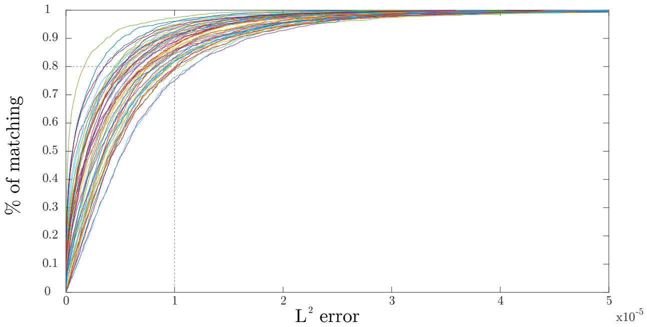

Fig. 16. We design consistent cross fields x_1 and x_2, on pairs of shapes from the SCAPE dataset ($\alpha_c = .1, \alpha_l = 0$), and we measure the pointwise L^2 error of the computed x_2 compared to $\mathrm{d}\phi(x_1)$ which is the pushforward of x_1 using the ground-truth map differential. In the above plot, we show the distribution of the error for all of the pairs. Notice that in most of the cases, 80% of the points have an error of at most 10^{-5}.

Genus 1 examples. In Fig. 1 (left), we show an example on a genus 1 model with intrinsic symmetry. Notice that since the tail is attached to the head of the kitten, it is unclear in this model where exactly the symmetry line goes through, which makes it a stress test for many mapping techniques. Nevertheless, we were able to compute a high quality functional map, which allows us to generate an approximately consistent quad mesh which mostly respects the underlying symmetry. In addition, we tried a similar experiment on a pair of meshes from the FAUST dataset, where we "glued" the hands of one of the persons. Using a reduced functional map of size $k = 50$, we obtain compatible quad meshes, as can be seen in Fig. 18. Notice that the right leg is somewhat less consistent and it is due to the map which is only approximate. We show in the zoomed-in figures that the same singularity structure is maintained on both meshes, but it is twisted on the left person. We validated that the mapping is wrong in this area by mapping a function from the left person to the right, and, indeed, the colors in the zoomed-in area are inconsistent between the meshes.

8 LIMITATIONS

One limitation of our method is that in some cases, we achieve poor consistency results on certain areas of the mesh, even though the general quadrangulation is relatively consistent. We believe it is due to the fact we omitted pointwise unit length constraint, allowing the optimization to reduce energy by scaling vectors in problematic regions. Related to this issue, is that we generate *uniform* quadrangulations, regardless of the underlying geometry. In this context, additionally optimizing for a consistent sizing field might be beneficial. Finally, as we mentioned in Sec. 3, our method produces only approximate consistent quad meshes, since we optimize for a guiding field and not directly for the quads. As a result, our method does not provide guarantees about the exact quality of the resulting quad meshes. All of these shortcoming offer interesting directions for further consideration and future work.

9 CONCLUSION AND FUTURE WORK

In this paper, we presented a novel unified technique for computing consistent quadrangulations of individual and pairs of shapes, with respect to a given symmetry and correspondence respectively. Our method does not require the input shapes to have the same triangulation and can handle shapes with arbitrary topology, while at the same time placing special emphasis on robustness and efficiency. Key to the success of our technique is a novel formulation that only requires a functional (rather than pointwise) correspondence across shapes and allows us to avoid the difficult estimation of the map differential, while being able to accommodate functional maps given in a reduced basis. Our formulation results in a simple and easy to implement method that produces more accurate results compared existing baselines and allows to handle more general difficult cases.

In the future we plan to extend our method to handle entire *collections* of shapes, and also to use our functional formulation to enable cross field design with other (possibly user-guided) novel constraints, which are difficult to enforce locally. In addition, our formulation is applicable to any N-RoSy fields, and not necessarily cross fields, and we wish to further investigate its applicability to joint design of PolyVector Fields [Diamanti et al. 2014].

REFERENCES

Noam Aigerman, Roi Poranne, and Yaron Lipman. 2015. Seamless surface mappings. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 72.

Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. 2005. SCAPE: shape completion and animation of people. In *ACM Transactions on Graphics (TOG)*, Vol. 24. ACM, 408–416.

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. Wiley Online Library, 73–82.

David Bommes, Bruno Lévy, Nico Pietroni, Enrico Puppo, Claudio Silva, Marco Tarini, and Denis Zorin. 2013. Quad-Mesh Generation and Processing: A Survey. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 51–76.

David Bommes, Henrik Zimmer, and Leif Kobbelt. 2009. Mixed-integer quadrangulation. *ACM Transactions On Graphics (TOG)* 28, 3 (2009), 77.

Olga Diamanti, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. 2014. Designing N-PolyVector Fields with Complex Polynomials. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 1–11.

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.

Theodore Frankel. 2011. *The geometry of physics: an introduction.* Cambridge University Press.

A Jacobson, D Panozzo, C Schüller, O Diamanti, Q Zhou, N Pietroni, and others. 2013. libigl: A simple C++ geometry processing library. (2013).

Vladimir G Kim, Yaron Lipman, and Thomas Funkhouser. 2011. Blended intrinsic maps. In *ACM Transactions on Graphics (TOG)*, Vol. 30. ACM, 79.

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

O. Litany, E. Rodolà, A. M. Bronstein, M. M. Bronstein, and D. Cremers. 2016. Non-Rigid Puzzles. *Computer Graphics Forum (Proc. SGP)* 35, 5 (2016), 135–143.

Giorgio Marcias, Nico Pietroni, Daniele Panozzo, Enrico Puppo, and Olga Sorkine-Hornung. 2013. Animation-Aware Quadrangulation. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 167–175.

Giorgio Marcias, Kenshi Takayama, Nico Pietroni, Daniele Panozzo, Olga Sorkine-Hornung, Enrico Puppo, and Paolo Cignoni. 2015. Data-driven interactive quadrangulation. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 65.

Min Meng and Ying He. 2016. Consistent quadrangulation for shape collections via feature line co-extraction. *Computer-Aided Design* 70 (2016), 78 – 88. {SPM} 2015.

Shigeyuki Morita. 2001. *Geometry of differential forms.* Vol. 201. American Mathematical Soc.

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.

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*. Article 9, 60 pages.

Daniele Panozzo, Yaron Lipman, Enrico Puppo, and Denis Zorin. 2012. Fields on symmetric surfaces. *ACM Trans. Graph.* 31, 4 (2012), 111–1.

J. Pokrass, A. M. Bronstein, M. M. Bronstein, P. Sprechmann, and G. Sapiro. 2013. Sparse Modeling of Intrinsic Correspondences. *Computer Graphics Forum* 32, 2pt4 (2013), 459–468.

Nicolas Ray, Wan Chiu Li, Bruno Lévy, Alla Sheffer, and Pierre Alliez. 2006. Periodic global parameterization. *ACM Transactions on Graphics (TOG)* 25, 4 (2006), 1460–1485.

Nicolas Ray, Bruno Vallet, Laurent Alonso, and Bruno Levy. 2009. Geometry-aware direction field processing. *ACM Transactions on Graphics (TOG)* 29, 1 (2009), 1.

Nicolas Ray, Bruno Vallet, Wan Chiu Li, and Bruno Lévy. 2008. N-symmetry direction field design. *ACM Transactions on Graphics (TOG)* 27, 2 (2008), 10.

E. Rodolà, M. Moeller, and D. Cremers. 2015. Point-wise Map Recovery and Refinement from Functional Correspondence. In *Proc. Vision, Modeling and Visualization (VMV).*

Szymon Rusinkiewicz. 2004. Estimating curvatures and their derivatives on triangle meshes. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on.* IEEE, 486–493.

Julien Tierny, Joel Daniels II, Luis G. Nonato, Valerio Pascucci, and Claudio T. Silva. 2011. Inspired quadrangulation. *Computer-Aided Design* 43, 11 (2011), 1516 – 1526. Solid and Physical Modeling 2011.

Amir Vaxman, Marcel Campen, Olga Diamanti, Daniele Panozzo, David Bommes, Klaus Hildebrandt, and Mirela Ben-Chen. 2016. Directional Field Synthesis, Design, and Processing. *Computer Graphics Forum* (2016).

Chih-Yuan Yao, Hung-Kuo Chu, Tao Ju, and Tong-Yee Lee. 2009. Compatible quadrangulation by sketching. *Computer Animation and Virtual Worlds* 20, 2-3 (2009), 101–109.

A PROOF OF PROPOSITION 1

PROPOSITION 1. *Given a cross field x and an arbitrary point $q \in M$, we compute the associated power vectors y_1 and y_2 at q using two different basis vectors b_1 and b_2, respectively. Then, for any real-valued function f, the following relation holds:*

$$\langle y_1, (\operatorname{grad} f)_{1,p} \rangle = \langle y_2, (\operatorname{grad} f)_{2,p} \rangle \, ,$$

where $(\operatorname{grad} f)_{i,p}$ is the power vector of $(\operatorname{grad} f)$ at q in the basis b_i.

Proof. Let x be one of the 4 vectors of the cross field at q. We represent it using b_i as $x = s_x R^{\theta_i} b_i$, where $s_x = \|x\|$, $\theta_i \in [0, 2\pi)$ and R^{θ} is counter-clockwise rotation by angle θ in the tangent plane of q. Similarly, let $(\operatorname{grad} f)(q) = s_f R^{\alpha_i} b_i$, where $s_f = \|(\operatorname{grad} f)(q)\|$, and we assumed that both s_x and s_f are not zero. Now, the power vector of x w.r.t the basis b_i is given by $y_i = R^{4\theta_i} b_i$, and similarly $(\operatorname{grad} f)(q)_{i,p} = R^{4\alpha_i} b_i$. The inner product is therefore:

$$\langle y_i, (\operatorname{grad} f)(q)_{i,p} \rangle = b_i^T R^{-4\theta_i} R^{4\alpha_i} b_i = \cos(4(\alpha_i - \theta_i)),$$

since the bases b_i are unit-length. Now simply note that the difference of angles is independent of the basis which gives us the result.

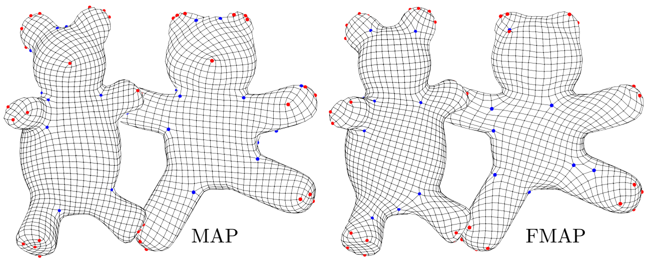

Fig. 17. We generated a precise mapping using sparse landmark correspondences given as input to the seamless method [Aigerman et al. 2015]. With the resulting map, we compute consistent cross fields on both meshes with the full (left) and reduced (right) basis.

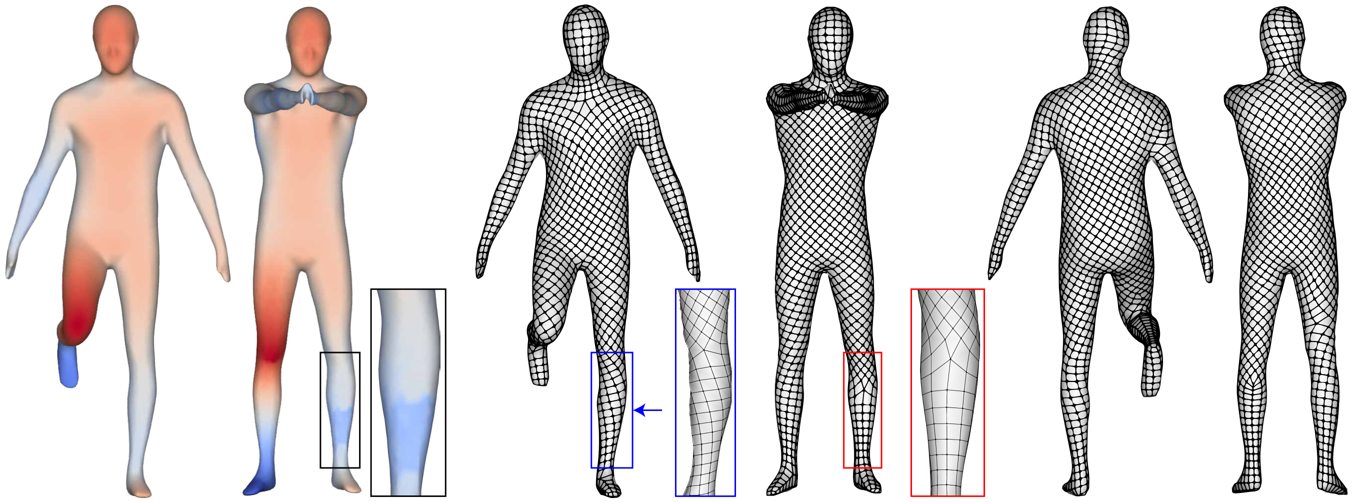

Fig. 18. Unfortunately, methods for mapping surfaces with different genus are scarce. Nevertheless, the robustness of our machinery to different mapping methods allows us to compute consistent quadrangulations even in the difficult case of genus 0 and genus 1 surfaces.

B A PRACTICAL OPTIMIZATION APPROACH

In what follows, we re-formulate our minimization problem (7) as a standard quadratic programming optimization problem. Our analysis shows that the involved Hessian is composed of a sparse term and a dense component, which is the product of a matrix and its transpose. Thus, we can facilitate MATLAB's `quadprog` with the `trust-region-reflective` method, allowing to solve a large and dense problem as long as its Hessian is *structured.* As the derivation for the case of a pair of shapes closely follows the single shape scenario, we omit the discussion of re-formulating the problem given in Eq. (11).

Recalling the smoothness and alignment terms, Eqs. (1) and (2), respectively, we observe that their associated Hessian matrices, H_s and H_l are extremely sparse. We denote

$$\begin{aligned} H_s &= \mathrm{grad}_p^T \, G_{\mathcal{E}} \, \mathrm{grad}_p \,, \\ H_l &= S^T \, G_{\mathcal{F}} \, S \,, \end{aligned}$$

where H_s has a sparsity structure of a Laplacian matrix (one-ring of faces) and H_l is a diagonal matrix. In addition, the alignment component includes a linear term which we denote by $f = -H_l \cdot w$ and a quadratic part in w which does not affect the optimization.

For the consistency component given in Eq. (6), we distinguish between two cases. In the first case, we use a reduced functional basis, i.e., $k < 300$, and we denote $G_{c,i} = C\,D(f_i) - D\,(C \cdot f_i)$, with $G_{c,i}$ being a *constant* matrix of size $k \times 2|\mathcal{F}|$, since C and f_i are fixed throughout the optimization. The consistency condition has the following Hessian:

$$H_c = \sum_{i=1}^{m} G_{c,i}^T \, G_{c,i} \,.$$

Unfortunately, direct computation of H_c results in a large and dense matrix and thus, in practice, we only perform manipulations of the form $G_{c,i}^T \cdot (G_{c,i} \cdot y)$. The second case, when $k = |\mathcal{V}|$, is much simpler as H_c is sparse and problem (7) can be solved directly in this scenario. Overall, we achieve the following Hessian,

$$H = (1 - \alpha_l)[(1 - \alpha_c)\, H_s + \alpha_c \, H_c] + \alpha_l \, H_l \,.$$

Finally, using the above notation, our problem (7) can be written as

$$\underset{y}{\mathrm{argmin}} \; \frac{1}{2} y^T H y + f^T \, y \,. \qquad (12)$$