In this paper we propose an approach for computing multiple high-quality near-isometric dense correspondences between a pair of 3D shapes. Our method is fully automatic and does not rely on user-provided landmarks or descriptors. This allows us to analyze the full space of maps and extract multiple diverse and accurate solutions, rather than optimizing for a single optimal correspondence as done in most previous approaches. To achieve this, we propose a compact tree structure based on the spectral map representation for encoding and enumerating possible rough initializations, and a novel efficient approach for refining them to dense pointwise maps. This leads to a new method capable of both producing multiple high-quality correspondences across shapes and revealing the symmetry structure of a shape without a priori information. In addition, we demonstrate through extensive experiments that our method is robust and results in more accurate correspondences than state-of-the-art for shape matching and symmetry detection.

CCS Concepts: • Computing methodologies → Shape analysis.

Additional Key Words and Phrases: Shape Matching, Spectral Methods, Functional Maps

ACM Reference Format:

1 INTRODUCTION

Computing dense correspondences across non-rigid 3D shapes is a classical problem in computer vision, computer graphics, and related fields. Dense correspondences between a pair of shapes are typically encoded by a map that assigns points on one shape to points on the other shape. There exist many optimization methods that define a map energy and try to find a desirable map by optimizing it [Bronstein et al. 2006; Ezuz et al. 2019; Huang et al. 2008; Ovsjanikov et al. 2010; Vestner et al. 2017b]. Unfortunately, the space of maps is very complex.
First, this space is exponential in size and can contain many local minima. For example, Fig. 2 shows the commonly-used geodesic distortion (the average change in geodesic distances across all pairs of points) energy of 10K self-maps on a human shape. We can see that the energy function is non-convex with multiple local minima that correspond to different types of maps. Some local minima correspond to semantically meaningful maps that exist due to the presence of symmetries, e.g., the left-to-right symmetry of a human shape. Other local minima correspond to other smooth high-quality maps, e.g., a twisted mapping where the lower part of a human is correctly mapped, but the upper part is flipped from left-to-right. In addition, there are local minima (typically with higher energy) that correspond to undesirable maps, e.g., a mostly smooth map, but with one large error such as the head being mapped to a hand.

Second, the optimization in the space of maps is difficult and current algorithms need to make many simplifications, e.g., via relaxations [Dym et al. 2017; Fogel et al. 2013; Solomon et al. 2016], early termination, etc. As a result, algorithms may not actually converge to local minima, but terminate at points that may or may not be close to a local minimum. We call these points termination points (of a particular algorithm).

Therefore, we can identify two types of errors in existing shape matching algorithms. 1) The algorithm terminates close to the correct local minimum, but it does not fully reach it because of the simplifications in the algorithm. This typically results in many small errors at each vertex, but only very few large errors. 2) The optimization algorithm ends up in a wrong part of the space, generally, because of the initialization. This latter problem is particularly prominent as many algorithms are based on pre-defined descriptors or landmarks [Ovsjanikov et al. 2012; Ren et al. 2018; Vestner et al. 2017a]. In Fig. 3 we visualize some example maps (A-F) that are results of a map optimization algorithm, ZoomOut [Melzi et al. 2019b], starting from different initializations. In the example, four maps are recovered corresponding to symmetries: the direct map (A), the symmetric map (B), the back-to-front flipped map (C), and the doubly flipped map (D). In addition, there are two twisted maps. In map (E) the upper body is assigned the symmetric counterpart, while the lower body arises from the direct map. In map (F) the upper body arises from the direct map and the lower body from the symmetric map. In addition, when comparing different map optimization algorithms, one can observe that different methods can lead to different maps, see Fig. 4.

We would like to advocate a departure from previous work, that has primarily focused on recovering a single high-quality map, and suggest that recovering multiple (or all) high quality maps in the space of maps is an important goal. Recovering multiple maps can provide information about all the intrinsic symmetries of the shapes and they also provide insights into the energy landscape of the optimization. For example, surprising low energy solutions of the geodesic distortion energy like the twisted maps can be recovered (not only accidentally) by existing methods. We therefore consider the problem of multi-solution shape matching and derive a new solution to this problem. We also note that multi-solution symmetry detection is a special case of this problem (See Fig. 1). However, our algorithm for recovering multiple maps is more than just a solution to a new problem statement. It can be used as building block to improve upon a long standing problem in traditional single-solution shape matching. For the vast majority of existing methods, the initialization is a considerable source of error and it can lead to a solution in the wrong part of the search space, e.g., a symmetric or twisted map is found when the goal is to compute the direct map. Complementary to efforts in improving the initialization or the optimization, we demonstrate that a considerable improvement in single-solution shape matching can be achieved by first computing a set of all high-quality maps and then selecting among them. As we show below, this can lead to significant improvements over the state-of-the-art even in traditional single-solution shape matching and single-solution symmetry detection.

For designing an algorithm to recover multiple high-quality maps, we can identify two criteria that an ideal solution should satisfy: (1) the recovered maps should be sufficiently different from each other. (2) each dense map should be smooth and near isometric. This includes maps that correspond to symmetries and other globally smooth maps that minimize energies that promote near isometries, e.g., based on geodesic distortion.

To tackle this problem, we propose to exploit the spectral properties of correspondences, encoded through the functional map representation [Ovsjanikov et al. 2012]. Key to our approach is a novel multi-resolution functional map exploration mechanism that completely avoids the need for descriptors or landmark correspondences.
and is capable of producing multiple meaningful dense correspondences that might exist in particular due to symmetries on the shapes. Our method is based on three observations, which we discuss below in detail: 1) pointwise maps that have large differences in the low frequencies of the spectral basis; 2) maps that are globally similar mainly differ in the high frequencies; 3) Large functional maps can be recovered from a small functional map [Melzi et al. 2019b]. Therefore, we explore the space of solutions through an efficient analysis of low frequency functional maps, which we then refine to obtain very accurate final correspondences. Depending on the application the recovery of multiple maps may be complemented by an algorithm for map selection, e.g., in single-solution shape matching the direct map should be selected.

To summarize, our main contributions are:

1. A new method to solve the multi-solution shape matching problem that is able to output multiple high-quality dense correspondences.
2. Significantly improved performance in near-isometric shape matching and symmetry detection compared to the state of the art.
3. A compact tree structure for enumerating and organizing the space of maps between a pair of shapes, that allows exploration and map selection a posteriori.

2 RELATED WORK

Below, we briefly review work that is most related to ours, while focusing on intrinsic symmetry detection, spectral methods and shape matching techniques.

2.1 Intrinsic Symmetry detection

In computer vision and computer graphics, intrinsic symmetries have been widely investigated as a tool to gain insight into the structure of shapes or scenes. We refer the interested readers to recent surveys [Liu et al. 2010; Mitra et al. 2013].

While early symmetry detection work primarily focused on extrinsic or embedding-dependent symmetries [Mitra et al. 2006], a significant number of techniques have also been proposed for intrinsic symmetry detection, including [Lipman et al. 2010; Ovsjanikov et al. 2008; Raviv et al. 2010; Xu et al. 2012, 2009] among others. Importantly, most early intrinsic symmetry detection methods either only detect similar parts, and thus do not produce dense correspondences [Lipman et al. 2010; Xu et al. 2012, 2009], suffer from robustness issues [Ovsjanikov et al. 2008] or result in difficult non-convex optimization problems [Raviv et al. 2010], making them sensitive to initialization and strongly relying on descriptors and subsampling.

More closely related to ours are recent methods based on advances in shape correspondence, especially the functional maps framework, including [Liu et al. 2015; Nagar and Raman 2018; Ren et al. 2018; Wang and Huang 2017]. These methods also aim to produce dense correspondences but in most cases require additional information such as detection of extremities [Liu et al. 2015] or a priori knowledge of symmetry orbits of a specified landmark of the shape [Wang and Huang 2017]. The method of Nagar et al. [Nagar and Raman 2018] that we compare to below, is related to our approach in its exploration of near-diagonal functional map structure, but relies on very sensitive eigenfunction sign detection.

2.2 Shape Matching

Our framework is also related to the problem of shape matching, and thus to methods that look for dense correspondences between non-rigid 3D shapes. For an in-depth review of this area we refer the readers to [Biasotti et al. 2016; Tam et al. 2013]. Several approaches to shape matching directly solve for correspondences between points on the two surfaces by minimizing an explicit energy, e.g., [Bronstein et al. 2006; Huang et al. 2008; Ovsjanikov et al. 2010]. The main limitation of these methods is that they often lead to complex combinatorial problems. An alternative is to first map the shapes to a canonical domain (e.g. a sphere), and then solve for the correspondence between these parametric representations [Aigerman and Lipman 2016; Aigerman et al. 2015; Lipman and Funkhouser 2009], or blend across multiple such maps [Kim et al. 2011].

One successful strategy for shape matching is to relax the search space from permutation matrices to a space more amenable to continuous optimization, e.g., doubly stochastic matrices [Dym et al. 2017; Fogel et al. 2013; Solomon et al. 2016], but also other successful relaxations exist [Azenot et al. 2019; Kezurer et al. 2015; Leordeanu and Hebert 2005; Maron et al. 2016; Solomon et al. 2012]. This, however, requires a possibly error-prone projection step to compute an integer solution, and makes it difficult to compute multiple correspondences, which might exist due to symmetries.

Our goal is to automatically and simultaneously produce multiple high-quality maps present due to different symmetries. Though there are no existing methods to solve our problem directly, there is some pioneering work for shape matching where multiple maps are involved. For example, multiple maps are generated and then blended into a single map in [Kim et al. 2011]. [Sahillioğlu 2018] proposed a genetic method for shape matching, where a “population” of maps are maintained for “evolution”. However, these approaches do not aim to produce multiple different high-quality dense correspondences and thus they cannot be easily adapted to our setting. [Sung and Kim 2013] output multiple maps that potentially contain the symmetric map for a bilateral shape but is hard to generalize to shapes with multiple symmetric axes. [Sahillioğlu and Yemez 2013] considered the symmetry ambiguity when computing a single map to avoid symmetry flips. All of the above mentioned methods involve multiple maps, however, none of the them aim to recover all possible symmetry-aware maps. The ones that produce multiple maps provide either a given class of maps (like the direct and
the symmetric for bilateral shapes) or a set of maps without any property (like intermediate maps).

2.3 Shape Matching with Functional Maps

Our approach is based on the functional map representation [Ovsjanikov et al. 2012]. The vast majority of methods that use this framework for shape matching start with a set of descriptor functions, derived from point signatures or from landmarks, and use them jointly with global map quality criteria to compute a correspondence [Aflalo and Kimmel 2013; Kovnatsky et al. 2013; Nogneng and Ovsjanikov 2017] (we refer to [Ovsjanikov et al. 2017] for an overview). While computing a functional map reduces to solving a least squares system, the conversion from a functional map to a point-wise map is not trivial and can lead to errors and noise [Ezuz and Ben-Chen 2017; Rodolà et al. 2015]. To improve accuracy, several desirable map attributes have been attributed via regularizers for the functional map estimation first using geometric insights [Burghardt et al. 2017; Eynard et al. 2016; Gehre et al. 2018; Litany et al. 2017b; Nogneng and Ovsjanikov 2017; Ren et al. 2018; Rodolà et al. 2017; Shoham et al. 2019; Wang et al. 2018a,b], and more recently using learning-based techniques [Halimi et al. 2019; Litany et al. 2017a; Roufosse et al. 2019]. Nevertheless, despite significant progress, the reliance on descriptors and decoupling of continuous optimization and pointwise map conversion remains common to all existing methods.

Our method is also related to spectral embedding methods. Some previous methods [Jain et al. 2007; Mateus et al. 2008; Ovsjanikov et al. 2008; Rustamov 2007] try to solve the shape matching (or symmetry detection) problem by directly aligning each dimension of the spectral embedding to obtain correspondences. In the functional map language this means forcing the functional maps to be diagonal [Nagar and Raman 2018]. Our method aims to explore block-diagonal functional maps as initializations that we then refine to dense correspondences without imposing diagonality. As we demonstrate below, this is crucial to obtain high quality correspondences in diverse and challenging cases.

2.4 Map Refinement

A common strategy for improving estimated correspondences consists in iterative map refinement as a post-processing step, e.g. [Man dad et al. 2017; Solomon et al. 2016; Vestner et al. 2017a,b]. The simplest refinement in the functional maps framework is the Iterative Closest Point algorithm in the spectral domain [Ovsjanikov et al. 2012]. Recently, other more advanced refinement methods for both functional and pointwise maps have been proposed in [Ezuz et al. 2019; Ren et al. 2018], that, respectively, try to minimize the bi-directional geodesic Dirichlet energy, and promote the bijectivity, smoothness and coverage of the correspondences. When shape collections are considered, a common strategy is to use cycle consistency constraints [Huang et al. 2014; Wang et al. 2013; Wang and Singer 2013]. Most closely related to ours, is a recent ZoomOut method proposed in [Melzi et al. 2019b], and based on iterative conversion between functional and pointwise maps.

The graph isomorphism problem. We also mention briefly that computing all high quality (e.g., nearly isometric) maps between shapes is related to counting all isomorphisms between graphs. This problem has been studied from a theoretical perspective, [Kobler et al. 2012; Malthon 1979], and although complexity equivalence results exist, no polynomial time algorithm is known.

3 MOTIVATION, BACKGROUND & NOTATION

Our goal is to find multiple high-quality maps on a shape pair or a single shape that encode different symmetries. In order to motivate our work, we would like to briefly review three simple solutions to the problem.

The first idea is to exhaustively enumerate all maps in the pointwise map space (with size $O(n^2)$, $n$ is the number of vertices) and select some maps based on some criterion (e.g., geodesic distortion energy). This is infeasible when the shapes have more than about 10 vertices due to the size of the space.

The second idea would be to use random sampling to only evaluate a subset of the maps. However, the probability of obtaining a high-quality map from random sampling is almost zero given the large size of the search space.

A third idea would be to take an existing optimization algorithm and start it with multiple different initializations. Common approaches for initializing maps include using pointwise descriptor, e.g., [Eisenberger et al. 2019; Vestner et al. 2017b] or initial segment or landmark correspondences [Ezuz et al. 2019; Kleiman and Ovsjanikov 2018; Ren et al. 2018]. For these methods it is actually very challenging to modify the initialization in a meaningful way so that the optimization methods would explore different parts of the map space. It would be necessary to run these methods with different random initializations. However, there is also no guarantee that different parts of the space will be explored and that a random initialization will lead to a meaningful map. While this approach is more promising than randomly sampling maps, it would still require a large number of random initializations. We will compare to this approach as a baseline to demonstrate that only few distinct maps can be recovered compared to our method (e.g. Table 3)

A second challenge also becomes evident. Simply proposing a set of maps is not sufficient for an application. We need to have a map

---

Table 1. Our method outputs 8 maps for this pair of human shapes, where the first four maps contain the complete left-to-right, or back-to-front symmetries, and the last four contain a combination across the upper and lower body. We report map quality measures, including orthogonality (“Ortho”), Laplacian Commutativity (“lapComm”), orientation preservation (“Orient”), edge length distortion (“EdgeDist”), and as-rigid-as-possible (“ARAP”) distortion. Note that according to most metrics all maps are comparable.

<table>
<thead>
<tr>
<th>Measure</th>
<th>map1</th>
<th>map2</th>
<th>map3</th>
<th>map4</th>
<th>map5</th>
<th>map6</th>
<th>map7</th>
<th>map8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ortho</td>
<td>19.9</td>
<td>20.3</td>
<td>19.9</td>
<td>19.9</td>
<td>18.6</td>
<td>17.3</td>
<td>19.3</td>
<td>20.5</td>
</tr>
<tr>
<td>LapComm</td>
<td>12.0</td>
<td>11.3</td>
<td>16.1</td>
<td>16.7</td>
<td>21.2</td>
<td>21.6</td>
<td>23.0</td>
<td>24.3</td>
</tr>
<tr>
<td>Orient</td>
<td>0.72</td>
<td>1.24</td>
<td>1.28</td>
<td>0.65</td>
<td>0.79</td>
<td>1.23</td>
<td>0.91</td>
<td>1.18</td>
</tr>
<tr>
<td>EdgeDist</td>
<td>1.46</td>
<td>1.97</td>
<td>2.56</td>
<td>3.73</td>
<td>4.54</td>
<td>4.30</td>
<td>5.73</td>
<td>4.57</td>
</tr>
<tr>
<td>ARAP</td>
<td>1.28</td>
<td>1.39</td>
<td>1.47</td>
<td>1.45</td>
<td>2.99</td>
<td>2.96</td>
<td>3.36</td>
<td>3.36</td>
</tr>
</tbody>
</table>
selection algorithm that either selects the direct map, a particular symmetric map, or all meaningful symmetric maps from a set of candidate maps.

We propose a novel algorithm, called MapTree, that tackles these two challenges. First of all, instead of dealing with the unorganized point-wise map search space, we propose to solve the same problem in the spectral domain based on the following observations. (1) each point-wise map can be represented as a functional map in a reduced basis [Ovsjanikov et al. 2012]. (2) the functional-map representation utilizes the eigenfunctions associated with specific frequencies (ordered eigenvalues). Therefore, the functional-map space can be well organized w.r.t. the frequency which makes it much easier for map exploration. (3) a single point-wise map can be converted to functional maps with different dimensions, where the low dimension encodes the global information of the point-wise map (such as the symmetry orientation), while the high dimension encodes the local details of the map. Therefore, to disambiguate two maps with good-quality, we can convert them to functional maps with different dimension for comparison.

Based on these observations, we propose a tree structure to explore the functional map space, where at low dimensions we can enumerate all possible functional maps in good-quality. The idea of exploring low-dimensional functional maps has been used for symmetry detection in e.g., [Nagar and Raman 2018] and to some extent in [Wang and Huang 2017]. However, those methods are limited by specific choice of the dimensionality (typically very low such as 6-10) in order to maintain efficiency. Moreover, our method is not limited to symmetry detection and can naturally be applied to compute multiple maps across shapes.

We clarify the notations that are used in our method. Given two shapes \( S_1 \) and \( S_2 \) represented as triangle meshes, we associate to each the Laplace-Beltrami operator using the cotangent weight discretization [Meyer et al. 2003]. The first \( k \) eigenfunctions of this operator on shape \( S_1 \) form a basis denoted by \( \Phi^{(k)}_{S_1} = \{ \phi_1^{(1)}, \phi_2^{(1)}, \ldots, \phi_k^{(1)} \} \), having the eigenfunctions stored as columns in a matrix.

Given a pointwise map \( T_{12} : S_1 \rightarrow S_2 \), that maps each vertex in shape \( S_1 \) to a vertex in shape \( S_2 \), we represent it with a binary matrix \( \Pi_{12} \), such that the \( \Pi_{12}(i,j) = 1 \) if \( T_{12}(i) = j \) and 0 otherwise. Throughout our paper, we use \( \Pi \) to denote pointwise correspondences implicitly restricting the matrix to be binary with exactly one value 1 at each row.

As introduced in [Ovsjanikov et al. 2012] the functional map representation of \( T_{12} \) is a linear transformation mapping functions on \( S_2 \) to functions on \( S_1 \) (note the change in direction compared to \( T_{12} \)). When a functional map corresponds to a pointwise map it can be written as a matrix:

\[
C_{21} = \Phi_{S_2}^{\dagger} \Pi_{12} \Phi_{S_1},
\]

where \( \dagger \) denotes the Moore-Penrose pseudo-inverse.

4 \ MAP SPACE EXPLORATION

Our overall objective is to recover a set of high quality maps between a given shape pair. As mentioned above, the recovered maps should be sufficiently different and, ideally all be approximately smooth and near isometric.

To tackle this problem, our main idea is to exploit the spectral map representation, in order to efficiently enumerate accurate low-dimensional functional maps, which we refine to near-isometric pointwise correspondences. We organize the functional maps into a tree structure, where each level corresponds to the map size, and a node corresponds to all functional maps that share a particular leading principal submatrix. Our general approach then consists in progressively expanding this tree to obtain high quality diverse solutions. In this section, we first discuss how the spectral techniques can help us explore the map space. We then describe our approach for recovering the map tree hierarchy starting with a pair of shapes.

4.1 How to efficiently explore the space of sufficiently different but high-quality maps?

As noted above, Fig. 2 illustrates that the commonly used geodesic distortion energy is strongly non-convex with multiple strong local minima. Fig. 3 further shows that two maps with similar geodesic distortion error can encode completely different symmetries. Finally, in Table 1 we show another shape pair, and report multiple commonly-used map quality criteria. Remark that these criteria cannot disambiguate maps with different symmetry effectively.

To better structure the space of different maps, we propose to use the multi-scale functional map representation. Namely, for each pointwise map \( T_{12} \) from \( S_1 \) to \( S_2 \), we can convert it into a functional map \( C_{21} \) by Eq. (1) of a particular size. Our general observation is that small (low-dimensional) functional maps provide a convenient way of structuring the map space for three reasons: 1) they allow to encode sufficiently different pointwise maps efficiently 2) as demonstrated below they can be explored effectively due to their small size, 3) they can be refined to obtain accurate pointwise correspondences.

The discriminating ability of low-dimensional functional maps comes from two complementary results. First, given two pointwise maps that are sufficiently close to each other, their functional map representation, especially in the low frequencies will be similar. Indeed, we have the following theorem (proved in Appendix A):
Theorem 4.1. Given two pointwise maps \( T^1_{12}, T^2_{12} : S_1 \rightarrow S_2 \) that are \( \delta \)-close, their functional map matrix representation \( C^1_{21}, C^2_{21} \) will satisfy the following bound
\[
\sum_i \left( C^1_{21}(i, i) - C^2_{21}(i, i) \right)^2 < (5\delta \lambda_1)^2 A_1 \forall i.
\]
Here two maps \( T^1, T^2 \) are \( \delta \)-close if for any vertex \( v \) the geodesic distance between \( T^1_{12}(v) \) and \( T^2_{12}(v) \) is smaller than \( \delta, \epsilon \) is some constant (which depends on \( S_2 \), \( A_1 \) is the surface area of shape \( S_1 \), and \( \lambda_1 \) is the \( i \)-th eigenvalue of shape \( S_2 \). This bound suggests that for two maps that are close to each other, \( \delta \) is small, we need larger \( \epsilon \) (higher frequencies) to disambiguate two maps.

Moreover, while Theorem 4.1 only establishes the bound in one direction, showing that nearby pointwise maps will result in similar low frequency functional maps, a complementary result can also be found. Specifically if two pointwise maps are sufficiently different, their low frequency functional map representations must also be different. Intuitively, this is because for two different pointwise maps it must be possible to find indicator functions of regions that are mapped differently by them. If those regions are large enough so that their projection onto the low frequency eigenfunctions are significant, this implies that the functional maps must be different in the sense of the Frobenius norm. We formalize this intuition in Appendix A.2.

In summary, there are several advantages of the spectral representation via functional map: (1) the Frobenius-norm between two functional maps is a cheap but relatively accurate approximation of the distance between two pointwise maps. (2) for two maps that are far from each other (e.g., with different symmetry), we only need relatively low frequencies to convert functional maps that can disambiguate the two pointwise maps (see Fig. 5 for an example). (3) for two maps that are close to each other, we then need higher frequencies for disambiguation. Therefore, we can organize pointwise maps in the spectral domain along the frequencies.

To exploit these properties we propose a pipeline that effectively explores low-frequency functional maps and then refines them to higher-frequency functional maps and accurate pointwise correspondences. For the latter stage, we extend the approach proposed in [Melzi et al. 2019b] that allows to obtain high quality maps from small low frequency functional maps.

### 4.2 Hierarchical Map Recovery

As suggested above, we propose to organize the functional maps in a tree structure (which we call the Map Tree) from low frequency to high frequency. The nodes at each level \( k \) of the tree represent different functional maps of size \( k \)-by-\( k \), and two nodes across adjacent levels \( k \) and \( k+1 \) are connected if and only if the functional matrices have the same leading principal \( k \)-by-\( k \) submatrix. Such a map tree structure provides a convenient way of map organization using the ordered nature of the Laplacian spectrum. It also helps to explore the map space efficiently.

Our main idea is to build the map tree in a coarse to fine fashion. Namely, we start with the root node and progressively test whether the descendant nodes can arise from a high quality map. Our basic algorithm, which we call Algorithm A, is summarized as follows:

1. Initialize the set \( S = \{ C_0 \} \) to contain the single functional map of size \( 1 \times 1 \) equal to the ratio of the values of the constant basis functions.
2. For each \( C_i \in S \):
   a. Let \( C^+_{ij} \) and \( C^-_{ij} \) be two functional maps obtained by adding an extra row and column to \( C_i \), with zero values, except for the bottom right diagonal, set to 1 and \(-1\) respectively.
   b. Apply a map refinement algorithm \( R \) given \( C^+_{ij} \) as initialization.
   c. If the resulting pointwise map \( T \) of is sufficiently high quality, convert \( T \) to a functional map of the same size as \( C^+_{ij} \) and add it to \( S \). Do the same for \( C^-_{ij} \).
   d. Repeat the previous two steps until all functional maps in \( S \) have size greater than some threshold \( \kappa \).

Intuitively, this algorithm progressively tries different diagonal functional map initializations and applies a refinement algorithm \( R \) to them. In order to analyze this map exploration approach, we need an assumption on the refinement algorithm. Specifically, we will call algorithm \( R \) complete if, given an initial functional map \( C \) of size \( k \times k \), and assuming there exists at least one isometric map, whose functional map representation has \( C \) as the leading principal \( k \times k \) sub-matrix, then algorithm \( R \) will recover such an isometry, and an empty set otherwise.

### ALGORITHM 1: Functional Map Expansion Rules

**Input:** Two shapes \( S_1 (i = 1,2) \) with corresponding eigenfunctions \( \{\phi^S_i\} \) and eigenvalues \( \{\lambda^S_i\} \); an initial functional map \( C_{ini} \in \mathbb{R}^{k \times k} \); A refinement method \( R \)

**Output:** A set of functional maps \( C \) expanded from \( C_{ini} \)

1. (1) group similar eigenvalues for both shapes: find the index \( p_i \) for shape \( S_1 \) such that \( \lambda^S_i - \lambda_{p_i}^S \leq \epsilon < \lambda^S_{p_i+1} - \lambda_{p_i+1}^S, i = 1, 2 \)
2. (2) enumerate the mappings between \( \{\phi^S_{p_1}, \ldots, \phi^S_{p_2}\} \) and \( \{\phi^S_{k_1}, \ldots, \phi^S_{k_2}\} \); store the assignment as a \( p_2 \times p_1 \) matrix \( C_{perm} \)
3. (3) construct \( C = \begin{pmatrix} C_{ini} & 0 \\ 0 & C_{perm} \end{pmatrix} \)
4. (4) refine each \( C \) using the given refinement method \( R \). Convert the refined maps to functional maps of size equal to that of \( C \), and output the result.

### ALGORITHM 2: Map Tree Exploration

**Input:** A pair of shapes \( S_1 (i = 1,2) \) with corresponding eigenfunctions \( \{\phi^S_i\} \) and eigenvalues \( \{\lambda^S_i\} \); some stopping criterion and some tree pruning rules

**Output:** A set of maps and the corresponding map tree

**Initialization:** Initialize the map tree with a node \( C_{ini} = \frac{\Sigma \phi^S_i}{\Sigma \phi^S_i} \) with the status “un-explored”

**while** Stopping Criterion Not Met **do**

1. **for each un-explored leaf** \( C \) of the map tree **do**
   a. (1) Apply Algo. 1 and obtain a set of functional maps \( C \)
   b. (2) Update the map tree by attaching each \( C \in C \) to the node \( C \) with the default status “un-explored”
   c. (3) Prune the tree w.r.t. the given pruning rule and label the pruned leaves as “explored”.

end
The following theorem shows that under certain assumptions, **Algorithm A** is guaranteed to recover the map tree up to level \( \kappa \).

**Theorem 4.2.** Suppose the Laplacians of \( S_1, S_2 \) have the same eigenvalues, none of which are repeating. Let \( \mathcal{M} \) denote the tree constructed from all isometries between \( S_2 \rightarrow S_1 \). Further suppose that the refinement algorithm \( \mathcal{R} \) is complete. Then the subtree of \( \mathcal{M} \) until level \( \kappa \) will coincide with the tree built from the output of Algorithm A.

**Proof.** See Appendix.

We remark that although there are \( 2^k \) possible functional maps of size \( k \times k \) that have \( +1 \) or \( -1 \) on the diagonal the complexity of **Algorithm A** depends on the number of recovered isometries. For example if there exists only one isometric correspondence, this algorithm will only test \( 2k \) initializations.

**Practical Implementation.** We propose the Map Tree Exploration algorithm (Algo. 2) with Functional Map Expansion Rules (Algo. 1) to implement the idea of Algorithm A with several practical modifications.

First, we take special care in the presence of eigenvalues that are close to being repeating, which can arise particularly for symmetric shapes (see Fig. 5 for an example). In this case, as has been observed in, e.g. [Ovsjanikov et al. 2008; Wang and Huang 2017] the corresponding eigenfunctions will form an isometry-invariant subspace. Therefore, we must allow not only \( \pm 1 \) on the diagonal, but also consider all orthogonal combinations among the corresponding eigenfunctions. To achieve this, we need two building blocks: (1) detecting repeating eigenvalues, and (2) enumerating combinations among the grouped eigenfunctions. Importantly, since our method tries a set of initializations for refinement, we only need a conservative estimate for declaring eigenvalues to be repeating, since poor initializations will be discarded. Therefore, we detect repeating eigenvalues by simple thresholding. For enumerating possible initializations we consider all permutations (allowing sign flips) among grouped eigenfunctions. We have observed this to remarkably well, due, in particular, to the robustness of our refinement method, although theoretically the invariant subspace can be continuous. Algorithm 1 describes the details of the functional map expansion rules used in practice. We stress that our approach is geared towards a discrete set of maps, and we do not consider continuous spaces, such as isometries between two spheres. Also note that the detected repeating eigenvalues of two shapes might not have the same size, therefore, the expanded functional map after applying Algorithm 1 can be rectangular.

Algorithm 2 describes our complete map tree construction method, which uses the expansion Algorithm 1 as a subroutine. Note that Algorithm 2 also relies on pruning rules for deciding when a map is of sufficiently good quality. In practice we use the following:

1. **Discard a refined map if its orthogonality error** \( E_{\text{ortho}}(\mathcal{C}) > \epsilon_1 \) or its Laplacian Commutativity error \( E_{\text{lapComm}}(\mathcal{C}) > \epsilon_2 \).
2. **Prune the leaf \( \mathcal{C} \) that converges to the same functional map as another leaf.**

Since the orthogonality \( (E_{\text{ortho}}(\mathcal{C}) = \|CC^T - I\|_F^2) \) and the Laplacian Commutativity \( (E_{\text{lapComm}}(\mathcal{C}) = \|CA_1 - \Delta SC_1\|_F^2) \) are strong indicators for detecting bad functional maps, We use them to remove the leaves/functional maps that are unlikely to be good. We introduce rule (2) to avoid initializations that converge to the same functional map after refinement. The rationale behind it is that, as we show in the next section, our new refinement method has strong convergence power, capable of producing high quality maps even from poor initializations. Thus, we use rule (2) to remove initializations that lead to the same maps to avoid redundancy. Fig. 6 shows how a map tree is constructed using our algorithm.

![Fig. 6. Illustration of Algorithm 2 for exploring the self-symmetry space of a vase. The red/purple squares represent nodes pruned due to poor quality and duplicates of refined maps respectively. The last column shows the final computed functional and corresponding point-wise maps that capture different self-symmetries.](image)

### 4.3 Bijective ZoomOut

In Algorithm 1 (step 4), we need a refinement method \( \mathcal{R} \) to refine the expanded functional map from the enumeration. For efficiency, we can use some spectral-based methods such as ICP [Ovsjanikov et al. 2012] or ZoomOut [Melzi et al. 2019b]. ZoomOut is the current state-of-the-art refinement for near-isometric shape matching, which progressively increases the frequency during the refinement. Specifically, it proposes the following energy to minimize, where each principal \( k \times k \) submatrix is constrained to be orthonormal:

\[
E_{\text{ZM}}(C_{21}) = \sum_k \frac{1}{k} \|C^{(k)}_{21}C^{(k)}_{12} - I_k\|_2^2
\]

subject to \( C_{12} = \Phi_{S_1}^\dagger \Pi_{12} \Phi_{S_2} \).

To remove the direction bias and improve upon ZoomOut, we propose to add the bijectivity constraint on the two functional maps mapping in different directions. Specifically, we propose to minimize the following energy:

\[
E_{\text{CZM}}(C_{21}) = E_{\text{ZM}}(C_{12}) + E_{\text{ZM}}(C_{21}) +
\sum_k \frac{1}{k} \|C^{(k)}_{12} - I_k\|_2^2 + \sum_k \frac{1}{k} \|C^{(k)}_{21} - I_k\|_2^2
\]

subject to \( C_{12} = \Phi_{S_1} \Pi_{12} \Phi_{S_2}, C_{21} = \Phi_{S_1}^\dagger \Pi_{12} \Phi_{S_2} \).

We use a similar half-quadratic splitting technique as [Melzi et al. 2019b] to decouple the pointwise maps \( \Pi \) and the corresponding functional maps \( C \). We provide the pseudo-code and implementation details in the appendix.

### 5 RESULTS

In this section, we show extensive quantitative and qualitative results on three applications: (1) multi-solution shape matching,
Table 2. We compare our MapTree method to other simple baselines on a pair of human shapes from the FAUST dataset. We sample 500 random point-wise maps and apply different refinement methods. We then report the quality of the refinement methods and the total runtime.

<table>
<thead>
<tr>
<th>Method</th>
<th># difMaps</th>
<th>Acc</th>
<th>GeoDist</th>
<th>Dirichtel</th>
<th>Conformal</th>
<th>runtime (h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Lin</td>
<td>500</td>
<td>0.623</td>
<td>2.088</td>
<td>1092</td>
<td>6.958</td>
<td>1.946</td>
</tr>
<tr>
<td>ICP50</td>
<td>500</td>
<td>0.463</td>
<td>1.880</td>
<td>75.78</td>
<td>21.83</td>
<td>1.946</td>
</tr>
<tr>
<td>PMF50</td>
<td>127</td>
<td>0.050</td>
<td>0.286</td>
<td>44.07</td>
<td>4.049</td>
<td>82.21</td>
</tr>
<tr>
<td>BCICP50</td>
<td>500</td>
<td>0.154</td>
<td>1.727</td>
<td>30.49</td>
<td>13.83</td>
<td>63.62</td>
</tr>
<tr>
<td>RHICP50</td>
<td>500</td>
<td>0.285</td>
<td>1.726</td>
<td>53.76</td>
<td>6.956</td>
<td>5.841</td>
</tr>
<tr>
<td>ZoomOut50-50</td>
<td>500</td>
<td>0.372</td>
<td>1.767</td>
<td>48.09</td>
<td>6.697</td>
<td>5.221</td>
</tr>
<tr>
<td>ZoomOut50-20</td>
<td>454</td>
<td>0.031</td>
<td>0.775</td>
<td>13.68</td>
<td>2.944</td>
<td>0.080</td>
</tr>
</tbody>
</table>

MapTree (Ours)     | 10        | 0.019| 0.375   | 11.26     | 1.815     | 0.002       |

Fig. 7. We show the shape pair that is tested in Table 2. Top: we show the best refined map of each method. Bottom: we show the four obtained maps using our method, including the direct map, symmetric map, back-to-front map, and doubly flipped map via texture transfer.

where our method is applied to produce multiple maps/symmetries; (2) standard single-solution non-rigid shape matching; (3) self-symmetry detection. In addition, we provide information on the parameters of the algorithm and its runtime.

To compare different methods, we measure the accuracy and the smoothness of the computed maps:

- **Accuracy**: The accuracy of a given map is obtained by measuring the geodesic distance between the computed correspondence and the given ground-truth correspondence, where the geodesic distance is normalized by the surface area to make this measurement comparable across different shape pairs.

- **Smoothness**: We use three different metrics to measure the smoothness of a given map, namely, the geodesic distortion, Dirichlet energy, and conformal distortion:
  - Geodesic Distortion: Assume that matrix $G_k(i, j)$ stores the normalized geodesic distances between all pairs of vertices $i$ and $j$ for shape $S_k$. Then for a given map $T_{12}: S_1 \rightarrow S_2$, we measure its geodesic distortion on every pair of $(i, j)$ as:
    \[
    \sum_i \sum_j \left( G_1(i, j) - G_2(T_{12}(i), T_{12}(j)) \right)^2
    \]
  - Dirichlet Energy: We measure the Dirichlet energy of the mapped vertex positions on the target shape.
  - Conformal Distortion: We use the same metric as proposed in [Ezuz et al. 2019], which is modified from [Hormann and Greiner 2000]: for each face $f$ in $S_1$, we compute $\frac{\sigma_1}{\sigma_2} + \frac{\sigma_2}{\sigma_1} - 2$.

Table 3. Quantitative comparison to (1) RandIni + ZoomOut, (2) IntSymm, and (3) OrientRev + ZoomOut. We report the number of detected high-quality maps (i.e., corresponding to some intrinsic symmetries) and the runtime on several shapes with different resolution.

<table>
<thead>
<tr>
<th>Shape</th>
<th>Figure Number</th>
<th>#Vtx</th>
<th># High-quality maps</th>
<th>Runtime (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human pair</td>
<td>Fig. 7</td>
<td>5K</td>
<td>1 1 1</td>
<td>286.2</td>
</tr>
<tr>
<td>Tiger pair</td>
<td>Fig. 8</td>
<td>5K</td>
<td>1 - 2</td>
<td>311</td>
</tr>
<tr>
<td>Gorilla pair</td>
<td>Fig. 9</td>
<td>5K</td>
<td>2 - 4</td>
<td>173.4</td>
</tr>
<tr>
<td>Cat pair</td>
<td>Fig. 17</td>
<td>2K</td>
<td>- 2</td>
<td>156.7</td>
</tr>
<tr>
<td>Centaur pair</td>
<td>Fig. 18</td>
<td>2K</td>
<td>- 2</td>
<td>140.6</td>
</tr>
</tbody>
</table>

5.1 Multi-Solution Shape Matching

Our algorithm MapTree has no published direct competitor for the problem of multi-solution shape matching. We therefore try to create a baseline method that consists of a modification of existing techniques. The most suitable idea seems to be to start with many different random point-to-point maps as initialization and use an existing optimization technique. Specifically, we randomly generated 500 pointwise maps between a human shape from the FAUST-remeshed dataset, and apply different refinement methods including recent state-of-the-art methods RHM [Ezuz et al. 2019] and ZoomOut [Melzi et al. 2019b]. A suitable competitor must be able to at least find the symmetric or the direct map in such an example. Therefore, when measuring the map accuracy, we consider both the direct and symmetric direction and pick the one that gives a smaller error. Accuracy and different smoothness metrics are reported in Table 2.

As a conclusion, we can observe that only ZoomOUT works in combination with random initialization. Other methods have an excessive computation time (e.g., over 50h for PMF and BCICP) and with the exceptions of PMF they cannot converge to competitive high-quality maps for any of the 500 random initializations. See Fig. 7 for a visual comparison of the best map found by each method. We thus select random initialization + ZoomOut as baseline method.

For our quantitative evaluation, we would like to test how many distinct high-quality near isometric maps can be recovered by different methods. To simplify this test, we only count near isometric maps associated with a symmetry and do not consider other low energy maps such as twisted maps as solution. Since we only have ground truth data for some of the maps, we use visual inspection to decide if the maps are smooth and correspond to a symmetry and the geodesic distortion metric to confirm that they are low energy. We compiled a small test dataset consisting of 7 interesting shapes for the recovery of maps from the shape to itself and 5 shape pairs for the recovery of maps between two shapes. In this test, we compare to three competitors: 1) our baseline consisting of 500
random initializations + ZoomOut 2) IntSymm [Nagar and Raman 2018] and 3) OrientRev [Ren et al. 2018] + ZoomOut. The last two competitors are listed for completeness only, as they are the current state of the art methods for single-solution symmetry detection. These methods can only recover a maximum of one map by design. Our results are shown in Table 3. We can observe that our method is the only algorithm that can reliably recover multiple high-quality near isometric maps. The result of random initialization + ZoomOut recovers fewer maps except for the cup and the centaur pair.

We provide an example visualization to show why random initialization + ZoomOut has difficulty recovering multiple distinct maps using multi-dimensional scaling (MDS) to embed the refined maps in 2D (see Fig. 8). We cluster the embedded points/maps and show the best map of each cluster. To better visualize the map, we show how each map transfers the front-left leg and the back-right leg. The visualization of the complete map via color transfer can be found in Fig. 16 in the appendix. We can notice that most of the maps are of low-quality, e.g., they map a leg to the back or to the head. Some maps are also not smooth and map one leg to two legs. Only one cluster contains high-quality maps, but none of the maps recover the symmetry. This shows that the size of the map search space is too large so that it is almost impossible to sample the random initial maps that can lead to different high-quality meaningful maps. In contrast, our method recovers the two meaningful (direct and symmetric) maps as shown in the insets of Fig. 8 and Fig. 16. We show another example where our tree structure can help to find more meaningful maps on a pair of gorillas in Fig. 9.

Finally, we showcase three models to visualize interesting maps that can be recovered by our method. For example, on the cup model (Fig. 10) we can recover an interesting inside-out map and on the glasses (Fig. 11) there are left-to-right and upside-down symmetries. Fig. 12 shows a challenging case of a knot shape with a smooth/continuous symmetry group. In this case, IntSymm even fails to output a single map (instead, it outputs an identity map) and OrientRev method produces a rather noisy map with mixed symmetry. As a comparison, our method still outputs several high-quality self-symmetric maps.

5.2 Non-rigid shape matching
We apply our map tree exploration Algorithm 2 to the SHREC’19 Challenge dataset [Melzi et al. 2019a], which involves 430 shape pairs from 44 shapes with different mesh resolution, triangulation, and partiality. We consider this to be one of the most challenging datasets available for shape matching and we use it to demonstrate that our MapTree algorithm can significantly outperform the state-of-the-art methods in traditional single solution shape matching. We compare to the best and most closely-related shape matching methods and list the average error, geodesic distortion, and runtime of each method in Table 4.

In single-solution shape matching the algorithms are expected to output a single map with the direct orientation (i.e, map the
Table 4. Quantitative results on the SHREC’19 Challenge. For the map-tree based methods, the geodesic distortion (shown in brackets) is averaged over all produced maps.

<table>
<thead>
<tr>
<th>Method \ Measurement</th>
<th>Error (×10⁻³)</th>
<th>Geodesic Distortion</th>
<th>Runtime (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIM [Kim et al. 2011]</td>
<td>81.2</td>
<td>2.314</td>
<td>169</td>
</tr>
<tr>
<td>Genetic [Sahillioglu 2018]</td>
<td>339.4</td>
<td>5.458</td>
<td>25.11</td>
</tr>
<tr>
<td>OrientOp (Ini) [Ren et al. 2018]</td>
<td>141.2</td>
<td>1.953</td>
<td>5.81</td>
</tr>
<tr>
<td>Ini + ICP [Ovsjanikov et al. 2012]</td>
<td>130.2</td>
<td>1.596</td>
<td>3.82</td>
</tr>
<tr>
<td>Ini + PMF [Vestner et al. 2017b]</td>
<td>143.5</td>
<td>1.862</td>
<td>42.9</td>
</tr>
<tr>
<td>Ini + BCICP [Ren et al. 2018]</td>
<td>85.0</td>
<td>1.147</td>
<td>209</td>
</tr>
<tr>
<td>Ini + RHM [Ezuz et al. 2019]</td>
<td>125.2</td>
<td>1.403</td>
<td>25.2</td>
</tr>
<tr>
<td>Ini + ZoomOut [Melzi et al. 2019b]</td>
<td>118.5</td>
<td>1.367</td>
<td>9.38</td>
</tr>
</tbody>
</table>

MapTree (Ours) - GT selection

<table>
<thead>
<tr>
<th>Method \ Measurement</th>
<th>Error (×10⁻³)</th>
<th>Geodesic Distortion</th>
<th>Runtime (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ini + OrientOp</td>
<td>28.42</td>
<td>0.93 (1.16)</td>
<td>32.5</td>
</tr>
<tr>
<td>Ini + ZoomOut</td>
<td>39.99</td>
<td>0.94 (1.16)</td>
<td>35.5</td>
</tr>
</tbody>
</table>

Fig. 13. Qualitative comparison of maps obtained with different methods. When the initialization (OrientOp) is of poor quality, the refinement methods, such as ICP [Ovsjanikov et al. 2012], PMF, BCICP, RHM, and ZoomOut fail to recover a good map. Meanwhile, BIM outputs a back-to-front symmetric map which is not preferable for symmetry detection. In contrast, our method can output a set of smooth maps with different orientations for different applications.

left of a shape to the left, assuming there is a left-to-right symmetry). Competing methods are: BIM [Kim et al. 2011], and several functional map based methods, where we use the best known automatic initialization [Ren et al. 2018], and refine it using different methods, such as ICP [Ovsjanikov et al. 2012], PMF [Vestner et al. 2017b], BCICP [Ren et al. 2018], RHM [Ezuz et al. 2019], and ZoomOut [Melzi et al. 2019b]. Fig. 13 shows an example of the maps obtained from different methods. In Table 4, we also compare to the genetic algorithm [Sahillioglu 2018], where sparse correspondences are computed and interpolated. As shown in the table, the average direct error (0.339) and the average error that allows symmetry flips (0.289) are significantly higher than ours (0.047). In Fig. 22 we visualize the computed maps by the genetic algorithm on three example shape pairs. Note that the genetic algorithm can produce accurate sparse correspondences but fails to produce high-quality and smooth dense maps.

We propose two versions of our algorithm: semi-automatic and automatic. In the semi-automatic method we assume that the user selects the best map among the available maps. We simulate this user selection by selecting the left-to-left direct map closest to the ground truth and denote this method as GT selection. The average error of our method is 28 while the best baseline gives 81.

The main idea of our automatic selection algorithm is to use cycle consistency. We propose a simple approach to achieve this in the spirit of [Huang et al. 2014], but using the fact that our method produces multiple maps for each shape pair. Specifically, for every pair $S_1, S_2$ in the benchmark, our algorithm produces 10 maps $\{C_{12}^{i}\}_{i=1}^{10}$. We perform an automatic map selection in two steps. We first initialize the selection for each shape pair with the map that minimizes the orientation-preservation error introduced in [Ren et al. 2018]. We then iteratively update this selection by considering triplets of shapes, and by minimizing the following cycle consistency energy:

$$E_{CyCons}(C_{12}^{i}) = \sum_{j \in J} \|C_{12}^{i}C_{2j}^{i} - C_{12}^{i}\|_F + \|C_{12}^{i}C_{2j}^{i} - C_{1j}^{i}\|_F$$

where $J$ is the set of all shape pairs in the benchmark that form a triplet with $S_1$ and $S_2$. Thus, we update the map selection for the pair $S_1, S_2$ by selecting $C_{1j}$ among the 10 possible maps, the one that minimizes this energy, while keeping the selection for the other shape pairs (identified by the superscript ⋆) in Equation (4) fixed from the previous iteration. This very simple strategy converges remarkably fast, within 2-3 iterations over all map pairs.

The results of the cycle consistency-based automatic selection are also shown in Table 4 as “MapTree (ours) - Auto selection.” This combination makes our method a fully automatic method for shape matching, which achieves the state-of-the-art accuracy with a 50% improvement over the best existing baseline. It is also interesting to note that the average geodesic distortion of all the produced maps across the complete dataset is 1.16. This is remarkable, because it shows that most of our recovered maps have smaller geodesic distortion than the direct map recovered by competing methods. It is also an indication that geodesic distortion alone is not sufficient to distinguish between the direct map and other high quality maps.

We perform a second large scale experiment on more than 200 pairs of SCAPE human shapes, where we also achieved a significant improvement over the best baseline by 23%. The details of the test on SCAPE and some additional visualizations including the geodesic distortion and texture transfer can be found in Appendix B.
5.3 Self-symmetry detection.

Our map tree algorithm can also be used to improve upon the state-of-the-art in single-solution self-symmetry detection. Table 5 reports a quantitative evaluation on self-symmetry detection on the SHREC’19 dataset that consists of 44 shapes with different poses. Our method produces eight self-maps per shape. We then select the preferable self-symmetric map among the 8 maps in two ways: (1) “GT selection”: we use the given ground-truth self-symmetric map to select the map with the smallest distance to the ground-truth. (2) “Auto selection”: we simply select the map that is far from the identity map with a small Laplacian-commutativity error. Note that this is a fully automatic method. Comparing to the best baseline IntSymm [Nagar and Raman 2018] on this dataset, our method achieves 24% improvement on accuracy with better geodesic distortion. Also note that, the maps produced by our method achieve the smallest conformal distortion compared to all the baselines, and achieve comparable Dirichlet energy and geodesic distortion to BIM, while outperforming all other methods.

Partial intrinsic symmetries. We also applied our self-symmetry exploration algorithm to shapes with partial intrinsic symmetries without modification. Fig. 14 shows such an example, where the input Neptune model is not strictly symmetric due to the topology change. Middle: we run our algorithm directly and obtain the self-map visualized via color transfer. Right: we only highlight the shape regions that have intrinsic symmetry.

5.4 Robustness

Our method is robust w.r.t. mesh triangulation and decimation. Specifically, the SHREC’19 dataset consists of 44 shapes with different mesh resolution and triangulation. Table 4 and 5 shows that our method is robust w.r.t. mesh triangulation. We further test the robustness of our method w.r.t. decimation (see Fig. 15). For a given shape pair, we downsample the target shape to different number of vertices. Our method still produces accurate and consistent maps.

5.5 Parameters

For our MapTree construction algorithm, we set $\epsilon$ in Algorithm 1 to 1, and the $\epsilon_1$ and $\epsilon_2$ in the pruning rules to 0.5 for most of our tests. For some cases we also set $\epsilon_1$, $\epsilon_2$ to larger values to allow more leaves to be explored with a prior knowledge that the investigated shape (pair) contains more complicated symmetries. We stop the tree construction process when the leaf functional map has size larger than 10 (for non-human shapes) or 20 (for human shapes).

5.6 Runtime & Implementation

We implemented the proposed methods in MATLAB, and conducted the experiments on a workstation with a 3.10GHz processor and 64 GB memory. The (average) runtime for different tests in seconds is reported (see Table 3, 4, 5). We used the sub-sampling strategy to speed-up our MapTree algorithm similar to the one used in [Melzi et al. 2019b]. Specifically, we only sample a small number of points (e.g., 200-500) on the shapes to compute a functional map in our MapTree algorithm. In the vast majority of cases our complete pipeline takes 60 seconds end-to-end. We will release our complete implementation for full reproducibility of all of the results.

6 CONCLUSION, LIMITATIONS & FUTURE WORK

In this paper, we proposed a novel approach to recover multiple high quality correspondences between a shape or a shape pair. To achieve this, we propose a compact tree structure to efficiently organize and explore map space without requiring any landmarks or initialization. We presented a large variety of experiments demonstrating that our method can be used to explore the self-symmetry space or even the partial intrinsic symmetry of a single shape, and
to explore the map space between a shape pair, which is not possible for the previous methods. We also show the quantitative results on challenging datasets showing that our method can be used as an automatic shape matching technique that does not require any descriptors or landmarks. The results show that our method significantly outperforms current state-of-the-art. Our algorithm is easy to implement with a small set of parameters.

Our method still has some limitations. First, we only target discrete sets of maps and symmetries, and do not consider continuous spaces such as the symmetry group of a sphere. Second, while outputting a set of high-quality maps can be beneficial, for the specific application of shape matching, it is not trivial to correctly select the preferred map, even from a small set. Third, we only consider manifold meshes in our setting. However, using a Laplace-Beltrami discretization such as [Belkin et al. 2009] or a recent work [Sharp and Crane 2020], our method could be directly generalized to non-manifold meshes or even point clouds. For partial matching, [Rodolà et al. 2017] can be helpful to design partial functional map expansion rules to potentially extend our work.

We also plan to investigate potential applications that can exploit our map tree structure. For example, Fig. 19 in the Appendix shows that the multiple detected maps can help the shape modeling process. Meanwhile, Fig. 5 shows that the global symmetry information is encoded in the relatively low frequencies. This suggests that our map tree structure encodes level-of-symmetry which might be useful for shape modeling and, e.g., making the triangulation of a given shape symmetric w.r.t. multiple symmetry axes.

ACKNOWLEDGMENTS
The authors would like to thank the anonymous reviewers for their valuable comments and helpful suggestions. Parts of this work were supported by the KAUST OSR Award No. CRG-2017-3426, and the ERC Starting Grant No. 758800 (EXPROTEA).

REFERENCES


Suppose we are given two compact surfaces $M$, $N$ and two smooth maps $T_1, T_2 : N \to M$. We consider the pullback operators $T_{F1}, T_{F2} : \mathcal{L}_2(M) \to \mathcal{L}_2(N)$, defined as $T_{F1}(f) = g_1$, where $g_1(x) = f(T_1(x))$, and similarly $T_{F1}(f) = g_2$, where $g_2(x) = f(T_2(x))$.

We will call $T_1$ and $T_2$ $\delta$-close if the following is satisfied: for each $x \in N$, $d_M(T_1(x), T_2(x)) < \delta$ for some $\delta \geq 0$. Here $d_M$ denotes the geodesic distance on $M$.

Now let $\varphi_i$ be an eigenfunction of the Laplace-Beltrami operator corresponding to eigenvalue $\lambda_i \geq 0$. Our goal is to relate $T_{F1}(\varphi_i)$ to $T_{F2}(\varphi_i)$ using the bound above.

Observe that if $g = T_{F1}(\varphi_i) - T_{F2}(\varphi_i)$ then

$$g(x) = \varphi_i(T_1(x)) - \varphi_i(T_2(x))$$

Using the results from [Arnaudon et al. 2017; Donnelly 2006] we recover the following bound: whenever $M$ is a smooth compact manifold, there exists a universal constant $C$ (depending on the manifold), such that for any eigenfunction $\varphi_i$, its gradient satisfies the following:

$$\|\nabla \varphi_i\|_\infty \leq C \lambda_i$$

Combining this bound with the assumption that $T_1$ and $T_2$ are $\delta$-close, and using the gradient theorem for the line integral on the surface we get that:

$$\left| \varphi_i(T_1(x)) - \varphi_i(T_2(x)) \right| \leq C \lambda_i \delta \quad \forall x \in N.$$  

This implies that:

$$\left\| T_{F1}(\varphi_i) - T_{F2}(\varphi_i) \right\|_2^2 = \int_{x \in N} \left( \varphi_i(T_1(x)) - \varphi_i(T_2(x)) \right)^2 d\mu(x)$$

$$\leq (C \lambda_i)^2 \text{Area}(N)$$

Now recall that the $i$th column of the functional map representation of a pointwise map $T$ encodes the coefficients of $TF(\varphi_i^M)$ expressed in $\varphi_i^N$. Therefore, by Plancherel’s theorem we obtain Theorem 4.1:

**Theorem.** Given two pointwise maps $T_1, T_2$ that are $\delta$-close, their functional map matrix representation $C_1, C_2$ will satisfy the following bound $\|C_1(i, j) - C_2(i, j)\|^2 < (C \lambda_i)^2 \text{Area}(N)$ for any $i$.

Note that this bound implies that the higher frequencies become more and more “unstable”. Or, alternatively, in order to disambiguate
pointwise maps that are very close, one must consider higher and higher frequencies.

A.2 Lower bound of functional map representation
Suppose we are given two shapes $M, N$, and two pointwise maps between them $T_1, T_2 : N \rightarrow M$. This induces two functional maps $T_{F1}, T_{F2} : L^2(M) \rightarrow L^2(N)$ given via pull-backs. Specifically for any function $f \in L^2(M)$, the corresponding function $g \in L^2(N)$ is given by $T_{F1}(f) = g$, where $g(y) = f(T_1(y))$ for any $y \in N$ (and similarly for $T_{F2}$ defined via pull-back with respect to $T_2$).

We define the functional map matrices $C_1, C_2$ by expressing $T_{F1}$ and $T_{F2}$ in the Laplace-Beltrami basis. Specifically, $C_1(i, j) = \langle T_{F1} \phi_i^M, \phi_j^N \rangle_M$, where $\phi_i^M$ is the $i$th eigenfunction of the Laplace-Beltrami operator on $M$, and $\langle \cdot, \cdot \rangle_M$ is the standard $L^2$ inner product for functions on $M$.

Our goal is to bound the difference between $T_1, T_2$ and the difference between the matrices $C_1, C_2$. Specifically we would like to show that if $T_1$ and $T_2$ are sufficiently different then the submatrices of $C_1, C_2$ will be different as well.

For this we consider any region $R \subseteq M$ and denote by $P_1, P_2$ their pre-images via $T_1$ and $T_2$. Namely, $P_i = \{ y \in N \mid T_i(y) \in M \}$. Note that if $\varphi_R$ is the indicator function of $R$ and $\varphi_{P_1}, \varphi_{P_2}$ are indicator functions of $P_1, P_2$, then we have $T_{F1}(\varphi_R) = \varphi_{P_1}$ and $T_{F2}(\varphi_R) = \varphi_{P_2}$.

Now observe that for any region $R$ we have:

$$\|T_{F1}(\varphi_R) - T_{F2}(\varphi_R)\|_2^2 = \int_N (\varphi_{P_1}(y) - \varphi_{P_2}(y))^2 dy = \text{Area}(P_1 \cap P_2),$$

where $\cap$ denotes the symmetric difference of the two sets $P_1 \cap P_2 = (P_1 \setminus P_2) \cup (P_2 \setminus P_1)$.

Suppose that the given maps $T_1, T_2$ are sufficiently different in the following sense: there exists a region $R$ s.t. its two pre-images $P_1, P_2$ via $T_1$ and $T_2$ are such that $\text{Area}(P_1 \cap P_2) > A$ where $A$ some constant. Moreover, suppose that the projection of $\varphi_R$ onto the first $k_M$ eigenfunctions of the LB operator on $M$ is accurate up to error $\epsilon$:

$$\|\varphi_R - \sum_{i=1}^{k_M} a_i \phi_i^M \|_2^2 < \epsilon \text{Area}(R),$$

where $a_i = \langle \varphi_i^M, \varphi_R \rangle_M$. Similarly, we assume that the projection of $\varphi_{P_1} \varphi_{P_2}$ onto the first $k_N$ eigenfunctions of the LB operator on $N$ is accurate up to error $\delta$:

$$\|\varphi_{P_1} \varphi_{P_2} - \sum_{i=1}^{k_N} b_i \phi_i^N \|_2^2 < \delta \text{Area}(P_1 \cap P_2),$$

where $b_i = \langle \varphi_i^N, \varphi_{P_1} \varphi_{P_2} \rangle_N$. Finally, for simplicity we will also assume that $T_1, T_2$ are both locally area-preserving, although a more refined variant of this bound can also be established under less restrictive assumptions (e.g., only assuming bijectivity and smoothness).

Now let $C_1$ and $C_2$ be the $k_N \times k_M$ leading principal submatrices of $C_1, C_2$ (i.e., sub-matrices obtained by taking the first $k_N$ rows and first $k_M$ columns of $C_1, C_2$). Our goal is to bound the Frobenius norm: $\|C_1 - C_2\|_F^2$ as a function of the constants $A, \epsilon, \delta$ introduced above.

For this remark that:

$$\|T_{F1}(\varphi_R) - T_{F2}(\varphi_R)\|_2 = \|T_{F1}(\varphi_R) - T_{F2}(\varphi_R) - T_{F1}(\varphi_R) + T_{F2}(\varphi_R)\|_2 \leq \|T_{F1}(\varphi_R) - T_{F1}(\varphi_R)\|_2 + \|T_{F2}(\varphi_R) - T_{F2}(\varphi_R)\|_2.$$
We then embed the refined maps in 2D with MDS. We apply a simple bijective algorithm to a simple baseline: applying ZoomOut to 500 random initial maps only contains a direct map.

For each cluster, we show one representative map that is closest to the direct and the symmetric map visualized via color transfer. (Right) We randomly generate 1000 point-wise maps and apply ZoomOut for refinement. We then embed the refined maps in 2D with MDS. We apply a simple K-means to cluster the 2D points and color each point/map accordingly. For each cluster, we show one representative map that is closest to the direct/symmetric map.

an initialization consisting of a functional map of size whose diagonal equals to vector $v_1$, of size $p_{\text{max}}$ where $v_1(i) = d_{m^2}^{-1}(i)$, $i = 1..p_{\text{max}}$. and, again by construction, must have tested the initialization consisting of a functional map whose diagonal equals to $v_2$, where $v_2(i) = d_{m^2}^{-1}(i)$, $i = 1..p_{\text{max}}$ and $v_2(p_{\text{max}}) = 1$. Note $\mathcal{M}_A$ does not contain nodes $m$, s.t. $d_{m^2}^{-1}(i) = v_2(i)$. Finally, we suppose that $n$ exists in $\mathcal{M}$, and therefore there must exist at least one isometry whose functional maps has as principal submatrix a diagonal matrix with values equal to $v_3$. But this contradicts the assumption that refinement method $\mathcal{R}$ is complete. Therefore, any node in $\mathcal{M}$ must exist in $\mathcal{M}_A$. Conversely, since every node in $\mathcal{M}_A$ corresponds to some isometry it must also exist in $\mathcal{M}$, which proves the theorem.

B ADDITIONAL RESULTS

In Algorithm 3, we give more details about the bijective ZoomOut that is used in our method. Fig. 16 shows the corresponding map visualization as discussed in Fig. 8. Fig. 17 and Fig. 18 show two examples of multi-solution shape matching, where we compare our method to a simple baseline: applying ZoomOut to 500 random pointwise maps. In the following, we show more qualitative and quantitative results of our method.

B.1 Self-symmetry detection

Fig. 19 shows a use case of how the detected self-symmetries can help the modeling process. To model a symmetric model, in general the artist need to manually setup the symmetry axes during the sculpting. However, it is not trivial to setup such symmetry axes when we want to have the model being symmetric w.r.t. another shape. For example, as shown in Fig. 19, we would like to model a dress that is symmetric w.r.t. the given legs shape. In this case, we can use our method to detect the self symmetries of the legs shape and use the obtained symmetries to complete the input garment and make it symmetric from back to front and from left to right w.r.t. the legs shape.
Table 6. Self-symmetry detection on FAUST dataset with 100 human shapes.

<table>
<thead>
<tr>
<th>Method</th>
<th>Measurement</th>
<th>Accuracy ($\times 10^{-3}$)</th>
<th>Geodesic distortion ($\times 10^{2}$)</th>
<th>Runtime (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIM</td>
<td></td>
<td>65.4</td>
<td>2.35</td>
<td>34.6</td>
</tr>
<tr>
<td>GroupRep</td>
<td></td>
<td>224</td>
<td>10.2</td>
<td>8.48</td>
</tr>
<tr>
<td>IniSymm</td>
<td></td>
<td>33.9</td>
<td>2.61</td>
<td>1.35</td>
</tr>
<tr>
<td>OrientRev (Ini)</td>
<td></td>
<td>68.0</td>
<td>5.52</td>
<td>0.59</td>
</tr>
<tr>
<td>Ini + ICP</td>
<td></td>
<td>38.3</td>
<td>3.94</td>
<td>13.3</td>
</tr>
<tr>
<td>Ini + BCICP</td>
<td></td>
<td>29.2</td>
<td>2.33</td>
<td>195</td>
</tr>
<tr>
<td>Ini + ZoomOut</td>
<td></td>
<td>16.1</td>
<td>1.86</td>
<td>22.6</td>
</tr>
<tr>
<td>Ours - GT selection</td>
<td></td>
<td>15.3</td>
<td>1.83 ($\pm 0.09$)</td>
<td>61.0</td>
</tr>
<tr>
<td>Ours - Auto selection</td>
<td></td>
<td>20.5</td>
<td>2.90 ($\pm 0.09$)</td>
<td>61.0</td>
</tr>
</tbody>
</table>

Fig. 20. Shape matching on SHREC’19 Challenge. The summary curves of different methods average over 430 shape pairs.

B.2 Correspondences between a shape pair

In Table 4 we report the accuracy and geodesic distortion of the maps obtained by different methods on 430 shape pairs from the SHREC’19 dataset. Here we show the cumulative curve of the errors in Fig. 20. Fig. 21 shows an example shape pair from the SHREC’19 dataset, for which we visualize both the computed maps and the geodesic distortion. Fig. 22 visualizes three maps generated by the genetic method [Sahillioğlu 2018] on shape pairs from the SHREC’19 dataset.

We also perform a similar test to 210 shape pairs from the SCAPE dataset (see Table 7). Similarly, we performed a simple ablation study and show that the map tree with our refinement method is more accurate and efficient than using the OrientOp initialization plus the ZoomOut refinement. We also use the cycle consistency to automatically select the maps with the preferred direct orientation from our output 5 maps, and the corresponding accuracy is 23% better than the best baseline. Fig. 23 shows a qualitative example of the SCAPE shape pair.

B.3 Partial symmetry detection

Fig. 24 and Fig. 25 show other two examples where the input shape is not symmetric intrinsically. In these cases, our method can still be applied directly to obtain some semantic partial symmetries. Our method can also be applied to explore the multi-scale structure of an input model with multiple objects. Specifically, we first detect the connected components of the input model, and apply our map exploration method to each component and each pair of components. Fig. 1 shows an example of a shape, in which our method detected different number of symmetry axes across different connected components. Fig. 26 shows another complex shape, and the results of our method, used to explore how a single connected component can be mapped to itself and to the other components.
Fig. 24. In this example three dancing girls are connected by a base, which makes the overall shape not symmetric. We apply our self-symmetry exploration algorithm directly to obtain self-maps, visualized via color transfer. Note that the partial symmetric axis (1) detected the symmetry of the upper body of the girl (C); the symmetry axis (2) detected the partial symmetry between the girl A and B; the symmetry axis (3) detected a more global partial symmetry, where the girl (A) and (C) are mapped to each other, and (B) is mapped to itself from left-to-right.

Fig. 25. For non-symmetric shapes, some previous methods [Lipman et al. 2010; Xu et al. 2012, 2009] can detect the partial intrinsic symmetries which are given as clusters of points without any exact pointwise matches (please see Fig. 14 in [Xu et al. 2012]). Our method can be applied to non-symmetric shapes as well and obtain the partial symmetric point-wise maps directly.

Fig. 26. Multi-scale symmetry detection. For an input model with multiple objects, we can first decompose it into individual connected components. (A) for a single component, we explore its self-symmetry space. Here we show three obtained self-symmetric maps of the baby angel, including the left-to-right, back-to-front, and the double flip self-symmetry. (B) we also explore different maps between two components. In this example, we find the direct map, the symmetric map, and the arm-to-leg correspondences between the body of two angels in different pose. (C) shows another example of finding the correspondences between two pieces of cloth.