# A Genetic Isometric Shape Correspondence Algorithm with Adaptive Sampling

YUSUF SAHILLIOĞLU, Middle East Technical University

We exploit the permutation creation ability of genetic optimization to find the permutation of one point set that puts it into correspondence with another one. To this end, we provide a genetic algorithm for the 3D shape correspondence problem, which is the main contribution of this article. As another significant contribution, we present an adaptive sampling approach that relocates the matched points based on the currently available correspondence via an alternating optimization. The point sets to be matched are sampled from two isometric (or nearly isometric) shapes. The sparse one-to-one correspondence, i.e., bijection, that we produce is validated both in terms of running time and accuracy in a comprehensive test suite that includes four standard shape benchmarks and state-of-the-art techniques.

CCS Concepts: • **Computing methodologies** → **Matching**; **Mesh models**; **Mesh geometry models**; **Shape analysis**;

Additional Key Words and Phrases: 3D shape correspondence, isometric shape correspondence, sparse correspondence, bijection, genetic algorithm, adaptive sampling

**ACM Reference format:**
Yusuf Sahillioğlu. 2018. A Genetic Isometric Shape Correspondence Algorithm with Adaptive Sampling. *ACM Trans. Graph.* 37, 5, Article 175 (October 2018), 14 pages.
https://doi.org/10.1145/3243593

## 1 INTRODUCTION

Three-dimensional (3D) content is becoming ubiquitous due to the availability of the economical data sensors, the user-friendliness of the 3D modeling software packages, and the simplicity of sharing over the world wide web. Automated algorithms for a repository of comparable 3D shapes are desired to (i) understand, e.g., decide on the average shape, (ii) update, e.g., transfer texture from the source shape to the others, (iii) populate, e.g., create a new shape based on the existing ones, and (iv) organize, e.g., align shapes in upright orientation, the collection. Established correspondence information between shapes is extremely useful to perform these tasks and a myriad of others.

Motivated by the usefulness of the shape correspondence problem, we develop a robust and efficient solution in the form of a genetic algorithm. In the search of a shape correspondence, it is common practice to resort to the discrete optimization schemes as the correspondence problem is essentially an assignment problem between two finite sets. Despite the convenience of genetic algorithms in the generation of permutations, which is related to the assignment problem, we see no attempt to take advantage of this natural fit in this domain. To this end, we represent a permutation that defines a correspondence as a chromosome and evolve many of them into the fittest one that yields the minimum-distortion correspondence. We also develop an adaptive sampling algorithm that can improve a given correspondence by iteratively moving the matched samples on one side of the correspondence. In particular, we apply it to the resulting correspondences of our genetic algorithm and another well-known technique for further improvement.

We address the most basic and widely analyzed setting of the correspondence problem: finding a correspondence between two full and isometric (or nearly isometric) shapes with no missing parts and no topological noise. The sought correspondence is sparse in that we match a subset of the shape vertices. It is possible to extract a sparse correspondence as a subset of the output of a computationally expensive dense correspondence algorithm. One can, however, avoid this cost by using a light sparse method if only a sparse set is required, e.g., for the dense pipeline initialization application.

**Contributions.** We summarize our key contributions as follows:

- We established the natural connection between the genetic optimization and the isometric shape correspondence problem.
- Our adaptive sampling algorithm that updates the positions of the samples has the potential to improve any existing sparse correspondence algorithm, as demonstrated with improvements over our genetic algorithm as well as over Sahillioğlu and Yemez (2012a).
- Our geometrically intuitive and easily implementable genetic algorithm enables simpler future work extensions. We, in particular, seamlessly extend our work to solve the more challenging problem of partial matching.
- We provided fully automatic initializations for a dense matching algorithm as well as a real 3D scan registration system.
- We outperformed at various levels the recent state-of-the-art shape correspondence methods (Aigerman and Lipman 2015; Aigerman et al. 2015; Kim et al. 2011; Maron et al. 2016; Sahillioğlu and Yemez 2012a; Solomon et al. 2016; Vestner et al. 2017).

We note that the source code and the executables for our method are available at http://www.ceng.metu.edu.tr/~ys/pubs/ysf-ga-as.rar.

This work was supported by TUBITAK under the project EEEAG-115E471.
Authors' address: Y. Sahillioğlu, Middle East Technical University, Computer Engineering Department, Ankara, 06800, Turkey; email: ys@ceng.metu.edu.tr.

0730-0301/2018/10-ART175 $15.00
https://doi.org/10.1145/3243593

## 2 RELATED WORK

To tackle the important 3D shape correspondence problem, there are typical ways that formulate preservation constraints on pairwise geodesic distances (Sahillioğlu and Yemez 2011), angles (Lipman and Funkhouser 2009), and functions (Ovsjanikov et al. 2012). As a work landing in the first category, we will focus on distance-preserving, i.e., isometric, correspondence algorithms and briefly mention the recent advances on the other two categories. A thorough survey of correspondence algorithms is available in van Kaick et al. (2011).

Isometric shape correspondence is arguably the most popular category as two shapes to be matched generally differ by isometric deformations in real world, e.g., articulated motions of humans. Besides, semantically similar shapes are expected to have similar metric structures and isometry is an important clue to capture such similarity. Some of the relevant methods, consequently, strive to minimize some distortion function that measures the deviation from isometry for a given correspondence or mapping (Huang et al. 2008; Rodolà et al. 2012; Sahillioğlu and Yemez 2010, 2013a). Compatibility between isometry-invariant descriptors can also be utilized to compute such mappings (Bronstein and Kokkinos 2010; Ovsjanikov et al. 2010; van Kaick et al. 2013).

Another type of isometric shape correspondence methods embeds two shapes to a common intermediate domain in such a way that corresponding points get embedded to nearby points, which in turn gives the desired mapping by simple closest-point searches. Domains include, but are not limited to, Euclidean space (Jain and Zhang 2006), disk (Aigerman et al. 2015), sphere (Aigerman et al. 2017; Aigerman and Lipman 2015; Kazhdan et al. 2012), and hyperbolic plane (Aigerman and Lipman 2016). Instead of an intermediate domain, some methods embed one shape to the curved surface of the other (Aflalo et al. 2016; Bronstein et al. 2006; Chen and Koltun 2015; Maron et al. 2016), which removes the embedding errors to an extra domain at the expense of increased computational time.

We also see learning-based isometric shape correspondence solutions (Rodolà et al. 2014; Wei et al. 2016) that are powerful enough to handle, to a certain extent, the challenging cases with topological noise (Lahner et al. 2016). They are also relatively robust to extremely partial scans, such as single-view range map captures and highly non-isometric deformations of clothed subjects. The downside of these approaches is their restriction to the trained class of shapes. Besides, the training process is quite demanding, e.g., requiring 50 million training examples, as they apply neural networks to extrinsic surface representations that are not invariant to isometric deformations. A set of recent works (Boscaini et al. 2016; Litany et al. 2017a) decreases this size to merely 80 training models by using intrinsic representations. Another intrinsic network (Maron et al. 2017) provides a non-isometric correspondence as long as the shapes come in sphere topology.

Non-isometric shape correspondence category is getting a growing attention thanks to the huge online 3D shape repositories, e.g., SketchUp 3D Warehouse, Yobi3D, that accommodate a diverse collection of semantically similar shapes. Aigerman et al. (2015), Ezuz and Ben-Chen (2017), Kim et al. (2011), Solomon et al. (2016), and Vestner et al. (2017) can deal with the lack of isometry to some extent thanks to their geometric distortion minimization approaches based on concepts such as conformality, area-preservation, regularization, and smoothness prior. Enabling topological updates during this minimization allows correspondence computation between even more challenging shape pairs (Fish et al. 2016; Zhu et al. 2017).

The functional map framework replaces the traditional point-to-point mappings by matching real-valued functions over surfaces (Ovsjanikov et al. 2012), which in turn has further potentials in combining and manipulating mappings (Kovnatsky et al. 2015; Litany et al. 2016, 2017b; Nogneng and Ovsjanikov 2017; Pokrass et al. 2013). The framework is, however, prone to conversion errors when recovering the point-to-point map from the optimal functional mapping (Rodolà and Cremers 2017).

Although we see genetic algorithms for geometry processing in the context of 3D content synthesis (Duda and Jakiela 1997; Pilat and Jacob 2008; Sims 1994; Xu et al. 2012), mesh unfolding (Xi et al. 2016), rigid registration (Chow et al. 2004; Silva et al. 2005; Yamany et al. 1999), and 2D image registration (Delibasis et al. 2010; Wang 2006), there is no application of this optimization scheme to the shape correspondence problem. We exploit the natural connection between this scheme and the isometric correspondence problem for the first time. Genetic algorithms are also widely used in the graph matching problem (Cross et al. 1997; Khoo and Suganthan 2001; Myers and Hancock 2001; Suganthan 2002), which in turn has the potential to enable further geometry processing applications, such as retrieval of 3D shapes represented as attributed relational graphs (Tao et al. 2012). These algorithms, in general, locate the optimum of some global consistency measure by genetic operations subject to a stochastic selection process. Weights and architectures of neural networks are also adjusted with genetic algorithms by mating and mutating networks (Goh et al. 2008; Stanley and Miikkulainen 2002), a feature that may come in handy in future learning-based graphics applications.

Sampling is an important part of correspondence algorithms. Although there are many useful and robust sampling algorithms (Bowers et al. 2010; Eldar et al. 1997; Gelfand et al. 2003; Nehab and Shilane 2004), none of them is designed with the shape correspondence application in mind. In an attempt to address this issue, Tevs et al. (2011) propose a planned landmark sampling that adds the most discriminative samples that minimize the entropy of the posterior distribution of potential matches. It is, however, unclear how to connect the decrease in entropy with the stability of shape correspondence. The joint sampling of two shapes is improved by evenly sampling high-curvature points in Sahillioğlu and Yemez (2012a). Ovsjanikov et al. (2011), however, propose condition number based sampling whose slightly different goal is to identify those samples such that if their matches are known then the shape correspondence problem becomes the easiest. Our alternative solution that is based on geometrically clear arguments handles the sample selection part of the correspondence computation process. Samples are selected adaptively based on the currently available correspondence.

## 3 PROBLEM STATEMENT AND OVERVIEW

Our goal is to establish a sparse one-to-one correspondence between two isometric (or nearly isometric) shapes, each discretized

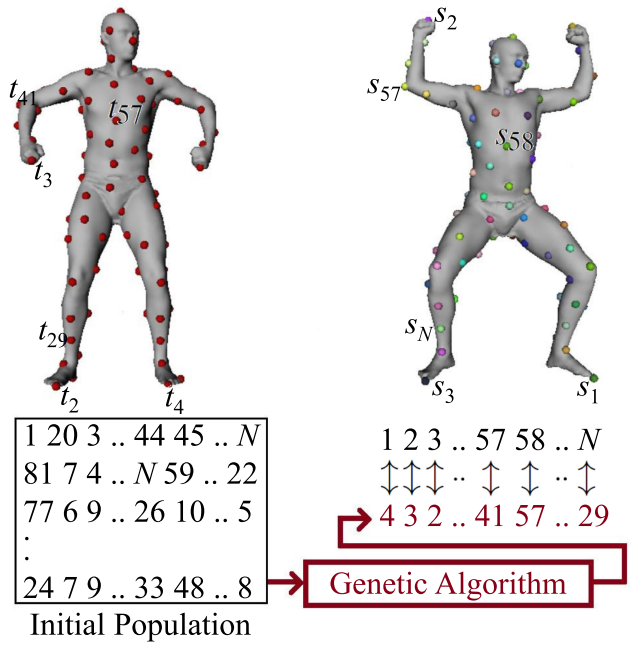



Fig. 1. Overview of our genetic algorithm.

as a mesh structure consisting of vertices, edges, and polygonal faces. We match a subset of vertices represented by a set of evenly spaced (Eldar et al. 1997) samples $S = \{s_i\}$ and $T = \{t_j\}$, respectively. As we are looking for a bijection, we have $|S| = |T| = N$.

We use the following isometric distortion measure to quantify the quality of a bijection $\phi : S \to T$:

$$\mathcal{D}_{\text{iso}}(\phi) = \frac{1}{|\phi|} \sum_{(s_i,t_j)\in\phi} d_{\text{iso}}(s_i, t_j), \tag{1}$$

where $d_{\text{iso}}(s_i, t_j)$ is the contribution of the constituent match $(s_i, t_j)$ to the overall isometric distortion:

$$d_{\text{iso}}(s_i, t_j) = \frac{1}{|\phi'|} \sum_{(s_l,t_m)\in\phi'} |d_{\text{g}}(s_i, s_l) - d_{\text{g}}(t_j, t_m)|, \tag{2}$$

where $d_{\text{g}}(.,.)$ is the geodesic distance between two vertices on a given surface and $\phi' = \phi - \{(s_i, t_j)\}$ in the most general setting. Both $d_{\text{iso}}$ and $\mathcal{D}_{\text{iso}}$ take values in the interval $[0, 1]$, since the function $d_{\text{g}}$ is normalized with respect to the maximum geodesic distance over the surface. $\mathcal{D}_{\text{iso}}$ is a variant of the measures used in Bronstein et al. (2006) and Sahillioğlu and Yemez (2011). The optimal bijection $\phi^*$ we are seeking minimizes $\mathcal{D}_{\text{iso}}$ in the huge space of all $N!$ possible bijections. We have this search space, since a bijection is merely an assignment of a permutation $\pi$ of the target samples to the fixed source samples. This line of thought would simplify the transition to our genetic algorithm (Section 4), where we seek the optimal permutation $\pi^*$ of indices that will be used as subscripts of $\{t_j\}$, e.g., fixed $s_1, s_2, \ldots, s_N$ is assigned to $t_4, t_3, \ldots, t_{29}$, respectively, and $\pi^* = 4, 3, \ldots, 29$ (Figure 1).

## 4 GENETIC ALGORITHM

### 4.1 Terminology

We first connect the bioinformatics and graphics terminology that will alternate throughout the rest of the article. Each candidate solution to our bijection search problem is called a *chromosome*, which represents a permutation $\pi$ of integers from 1 to $N$ that would be subscripted to the target samples $\{t_j\}$ in an attempt to put them in correspondence with the fixed source samples $\{s_1, s_2, \ldots, s_N\}$ as demonstrated in Figure 1. Each integer in a chromosome is called a *gene*, which basically represents a target sample, e.g., the first gene of the second chromosome in the initial population of Figure 1 represents the sample $t_{81}$. The set of chromosomes make up the *population*, which is initialized as the first *generation*. Through genetic operations, such as *crossovers* and *mutations*, the current population is evolved to a different (and better) one, called the next generation. Eventually the fittest chromosome of the final generation (rendered in red in Figure 1) represents the solution to our problem.

### 4.2 Main Design Decisions

First of all, we need a meaningful measure to evaluate the fitness of a given chromosome as it will guide the evolutionary process. We define the fitness of a chromosome representing a permutation $\pi$ as $\mathcal{F}(\pi) = 1 - \mathcal{D}_{\text{iso}}(\phi_\pi)$, where $\phi_\pi : S \to T$ is the bijection that maps $s_i$ to $t_{\pi[i]}$ with $\pi[i]$ being the $i$th integer in $\pi$ for $1 \le i \le N$.

The crossover operation takes two chromosomes and mixes them together into a newborn child chromosome. The mutation operation, however, takes one chromosome and converts it into a new one. The resulting new chromosomes are ensured to be fitter than the input chromosomes (detailed in Section 4.4). With this in mind, we mix the healthy sections of two bijections (Figures 2(a) and 2(b)) into a better bijection, or equivalently a fitter chromosome (Figure 2(c)), which is refined further by individual updates (Figure 2(d)).

Only three genetic algorithm parameters are in use: crossover rate $f_{\text{xover}}$, mutation rate $f_{\text{mutation}}$, and population size $P$. Despite the diversity of our test meshes, we always used the same values; the first two are fixed to 0.85, and the last one to $10N$. Unlike many genetic algorithms, user does not need to tune up many parameters, which we consider as an advantage.

### 4.3 Initial Population

The population is initialized into the first generation of $P$ chromosomes based on our descriptor $\mathbf{g}$ defined at each sample as a vector of geodesic distances to a few special samples that are already accurately matched—this initial bijection between few special samples is denoted by $\varphi$ and obtained as explained in Section 4.5. Specifically, $\mathbf{g}_i^{\text{s}} = [d_{\text{g}}(a_1, s_i)\ d_{\text{g}}(a_2, s_i)\ ..\ d_{\text{g}}(a_{|\varphi|}, s_i)]$ and $\mathbf{g}_j^{\text{t}} = [d_{\text{g}}(b_1, t_j)\ d_{\text{g}}(b_2, t_j)\ ..\ d_{\text{g}}(b_{|\varphi|}, t_j)]$, where $\{(a_k, b_k)\}$ are the constituent matches of $\varphi$ with $1 \le k \le |\varphi| \ll N$.

Having computed $\mathbf{g}$, we define a set of initial match candidates for each source sample $s_i$ by inserting into $\mathbf{c}_i^{\text{s}}$ all the target samples $\{t_j\}$ that satisfy $d_{\mathbf{c}} < \tau$, where

$$d_{\mathbf{c}}\left(\mathbf{g}_i^{\text{s}}, \mathbf{g}_j^{\text{t}}\right) = \max_k \left|\mathbf{g}_i^{\text{s}}[k] - \mathbf{g}_j^{\text{t}}[k]\right|, \tag{3}$$

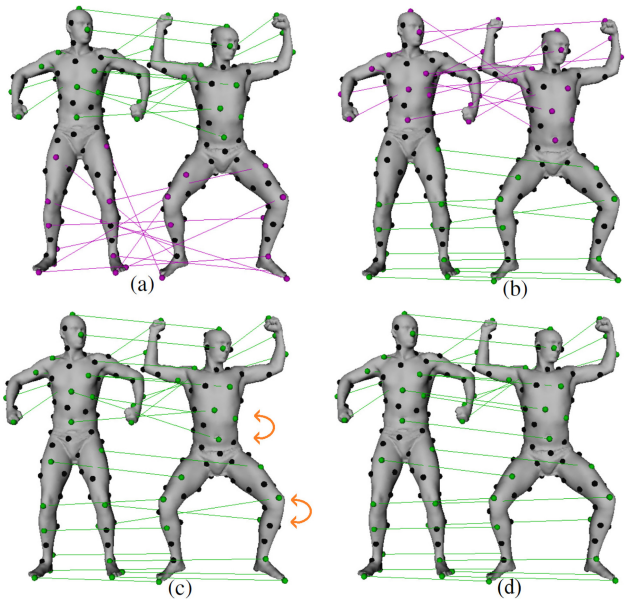



Fig. 2. Healthy, i.e., less distorted, sections of two bijections (greens at (a) and (b)) are crossed-over into a better bijection (c), whose smaller problems (orange highlights) are refined via mutations (d). Matches of the black samples are not shown for visual convenience.

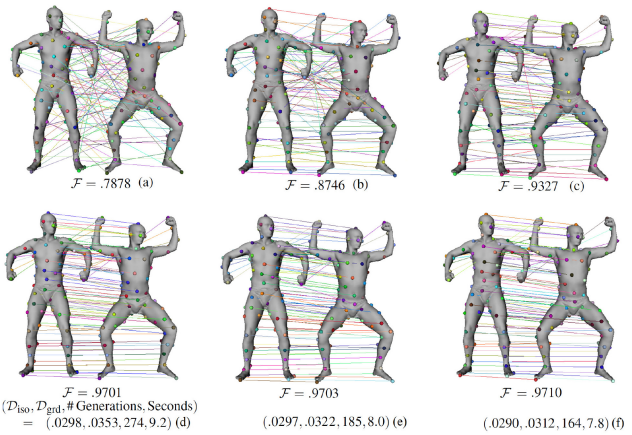



Fig. 3. The fittest member of the initial population of 1,000 chromosomes (a to c) and the resulting bijections (d to f). Initialization is done at random (a and d), by ranking of average geodesic distances (b and e), and by our scheme (c and f). 100 samples are matched. Corresponding samples (spheres) and the connecting line are colored the same for each match. This visualization scheme applies to the subsequent figures as well.

where $\mathbf{g}[k]$ is the $k$th entry in $\mathbf{g}$. We use $\tau = 0.125$, which corresponds to, e.g., half of the toe to knee normalized geodesic distance on a human model. This enables at worst a knee matching for a toe sample in the initial population. We have on average $|\mathbf{c}_i^{\mathrm{s}}| = 10$. Each initial chromosome is then filled by picking a random element for its $i$th entry from $\mathbf{c}_i^{\mathrm{s}}$, i.e., $\pi[i] = t_j \in \mathbf{c}_i^{\mathrm{s}}$. To preserve bijection, we prevent duplicates in $\pi$ by marking the newly picked sample as unavailable for the upcoming picks. Due to this marking, some samples, about 6%, may not find an available initial match candidate in which case we simply use the first available target sample.

Thanks to our robust evolution scheme (Section 4.4), starting with a completely random population will also lead to a good final generation, but not as fast and accurate as our initialization. We also use the compatibility of the rankings (Ganapathi-Subramanian et al. 2016) of the average geodesic distance descriptors $g_i^{\mathrm{s}} = \frac{\sum_{j \neq i} d_{\mathrm{g}}(s_i, s_j)}{N-1}$ and again could not achieve the performance of our initialization scheme (Figure 3).

## 4.4 Evolution of Population

We develop our method in the spirit of the microbial genetic algorithm (Harvey 2009) with our contributions to make the process work well in the context of 3D isometric shape correspondence.

In the evolution of the populations, we stick to the survival-of-the-fittest paradigm in that the best existing chromosomes are mixed together to breed new chromosomes that are also expected to do well (crossover operation). We also add some new genetic material by updating individual chromosomes (mutation operation). Please see Figure 6 for the fully detailed yet compact pseudocode of our genetic algorithm.

We start by evaluating the fitness $\mathcal{F}$ of each chromosome $C_i$ and store it in $C_i^{\mathcal{F}}$ to avoid recomputations. Based on the descendingly sorted $C_i^{\mathcal{F}}$ values, we divide the current population into two parts, a good part possessing the first $h$ chromosomes and a bad part covering the rest (typically $h = P/2$). We then replace some chromosomes in the bad part, induced by $f_{\mathrm{xover}}$, by the resulting child of the two parents from the good part. The parent with a higher $\mathcal{F}$ is marked as the winner and transfers multiple parts from its chromosomes onto the loser parent, hence transferring its good genes. The untouched part of the loser is tried to be maintained as much as possible, since loser is after all from the good part and should possess acceptable genes. Duplicated genes are also prevented in the updated loser, which is returned as the resulting child chromosome. We implement this crossover mechanism with the full details in Figure 7 and demonstrate it in Section 4.4.1 and Figure 5. The new population is then exposed to a mutation operation during which two bad genes of the input chromosome are swapped such that the exchange makes $\mathbf{g}$ descriptors of the corresponding source and target samples compatible. Mutation is also done on some chromosomes (fittest is exempt) induced by $f_{\mathrm{mutation}}$. Specifically, our mutation operation checks whether the geodesic vector $\mathbf{g}$ of the $i$th gene/sample of the input chromosome is compatible with the $\mathbf{g}$ of the $i$th source sample. Two geodesic vectors are compatible if all the corresponding entries in the vectors differ by at most $\tau$. In case of incompatibility, we look for a candidate gene whose swap with the $i$th gene brings compatibility to the $i$th descriptors (Figure 7). The relative advantages of the crossover and mutation operations are visualized in Figure 4.

Note that this process brings elitism for free, which means that at least one best chromosome is copied without changes to the new population of the next generation, so the best in the breed remains in population and can survive to end of run. In other words, elitism enables a guaranteed monotonic increase in $\mathcal{F}$, as shown in Figure 9. We achieve elitism as shown in the line 21 of Figure 6

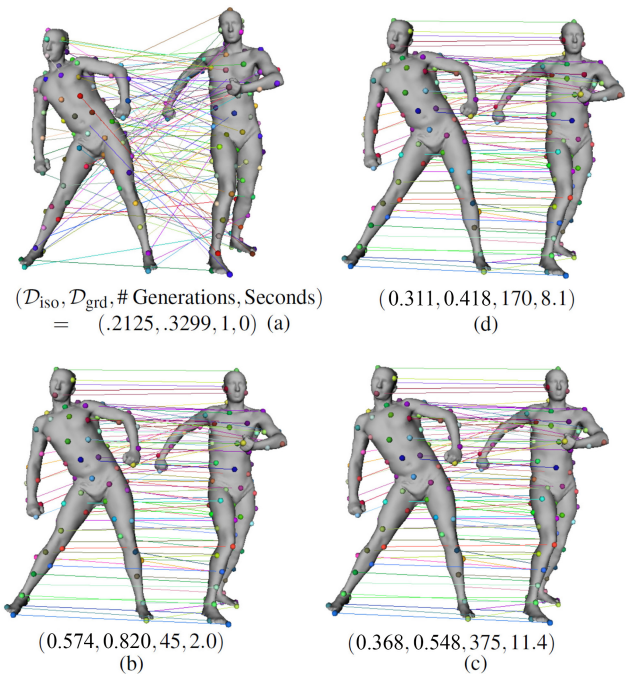



Fig. 4. The fittest member of the random initialization (a) and the resulting bijections when only mutations enabled (b), only crossovers enabled (c), both enabled (d). Population size $P$ = 1,000, number of samples $N$ = 100.

where the crossover operation always replaces a chromosome $C_i$ from the bad part that certainly excludes the best/fittest one. Similarly, mutation operation is guaranteed not to update the best one by starting the loop from index 2, not 1, in line 24.

Finally, while replacing a bad chromosome via the crossover of two good parent chromosomes, we repeat the crossover operation five times and use the fittest of the child chromosomes, which renders crossovers more accurate at the expense of increased completion time per generation. The increased crossover accuracy helps the evolution converge faster with less number of generations.

*4.4.1 Execution Trace of a Crossover.* To clarify our crossover algorithm, we run it through the example case in Figure 7, where colored substrings **3 12 17 9**, **4 7 6**, and **11 16** in the winner chromosome will replace **6 10 9 16**, **3 1 2**, and **13 14** in the loser chromosome, respectively, without creating any duplicates in the resulting child chromosome. The main idea for a duplication-free crossover is as follows. When the first character of the winner's substring collection $T$ = **3 12 17 9 4 7 6 11 16**, namely **3**, is transferred to the loser (at index 7), it will not cause any duplication, because there is already a **3** in the loser's $D$ = **6 10 9 16 3 1 2 13 14**, which is scheduled to be deleted, i.e., to be replaced by **4** due to **4 7 6→3 1 2**. Since the transfer of **3** to the loser index 7 will be compensated by the deletion of **3** from the loser index 2, we need no further action (Figure 5(a)).

When the next character from $T$, namely **12**, is transferred to the loser (at index 8), it will cause a duplication, because there is no other 12 to be deleted from the loser, i.e., loser's **12** does not belong to any colored substring to be deleted, namely, **12** $\notin D$.

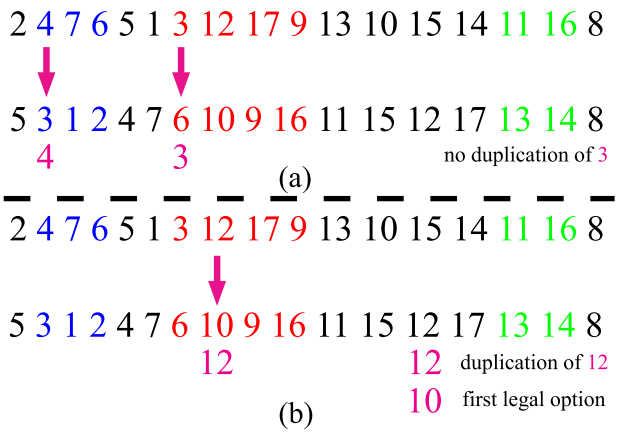
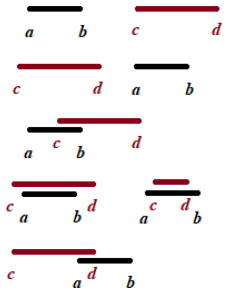



Fig. 5. Main idea of our duplication-free crossover operation.

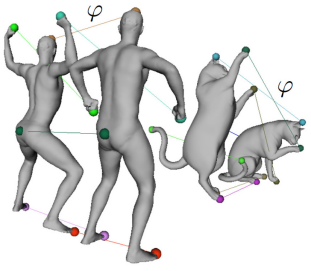
Since the transfer of **12** to the loser index 8 will cause a duplication with the **12** at index 13, we will replace the **12** at index 13 using the first legal value in $D$. The first option **6** is not legal, because it exists in $T$, meaning that it will be transferred to loser (at index 4 due to **4 7 6→3 1 2**), which in turn would cause a different duplication involving indices 4 and 13. The next option **10** is legal as it does not exist in $T$, meaning that it will never be transferred to the loser after getting deleted by **3 12 17 9→6 10 9 6** (Figure 5(b)). We finally set **10** as illegal for the upcoming iterations.

Efficient linear-time implementation of this main idea is provided in Figure 7, which can directly be used for reproduction purposes. This algorithm requires substrings to be nonoverlapping, which we guarantee by checking certain conditions for the query interval $[a, b]$ vs. each of the existing accepted intervals $[c, d]$ via the following quick macro that is based on the figure at right (last three rows show overlap cases):




$$\text{Overlaps if } c \le b \le d \ \vee \ c \le a \le d \ \vee \ (a < c \wedge d < b) \quad (4)$$

## 4.5 Initial Bijection




While evaluating $\mathcal{F}$, we set $\phi' = \varphi$ to bring down the time complexity of the frequently used $\mathcal{D}_{\text{iso}}$ computation from $O(N^2)$ to $O(|\varphi|N)$, which is linear in $N$ as $|\varphi| \ll N$. To this end, we create the initial bijection $\varphi$ between a few special samples, namely the extremities and the center, that can be robustly matched. For this sampling, we automatically stop the Farthest Point Sampling (FPS) (Eldar et al. 1997) when the distance of the next sample to the closest existing sample is one third of the maximum geodesic distance over the surface, i.e., the geodesic distance

Input: $S$ and $T$ s.t. $|S| = |T| = N$, samples on two meshes
Output: $\phi : S \rightarrow T$, one-to-one correspondence
$U$ = **initPopulation**() //$U = \{C_1, C_2, .., C_P\}$, $P$ is population size
For *generation* = 1 to max # generations
  $\mathcal{F}^*$ = **getFittest**($U$) //Evaluates all $\{C_i\}$ via $\mathcal{F}$ ($C_i^{\mathcal{F}}$ made ready)
  If ($\mathcal{F}^*$ is fixed for the last $L_1$ generations || //$L_1 = 100$
    no swap mutations in the last $L_2$ generations || //$L_2 = 10$
    $\mathcal{F}^* > 1 - \epsilon$) //$\epsilon = .001$
   Break; //Converged!
  **evolvePopulation**()
For $i = 1 : N$ //$N$ is # of genes (= samples on mesh)
  $\phi(s_i) = t_{C_1[i]}$ //fittest is maintained as the first chromosome $C_1$
Return $\phi$

**evolvePopulation**(Population $U$)
  Descending sort on $U$ s.t. $C_i^{\mathcal{F}} > C_j^{\mathcal{F}}$ $\forall i < j$, i.e., $C_1$ is the fittest
  $G = \{C_1, C_2, .., C_h\}$, $B = \{C_{h+1}, .., C_{N_p}\}$ //Good and bad parts,
  For each $C_i \in B$ //where $h = P/2$
   If **rand**() $< f_{\text{xover}}$ //**rand**() returns a number in [0, 1]
    Let $C_j$ and $C_k$ be random chromosomes from $G$ s.t. $C_j^{\mathcal{F}} > C_k^{\mathcal{F}}$
    $C_i$ = **xover**($C_j$, $C_k$) //$C_i \in B$ updated by the newborn child of
      //2 good parents. Elitism for free as $C_i$ can't
      //initially be $C_1$, the fittest chromosome
  For $i = 2 : P$
   If **rand**() $< f_{\text{mutation}}$
    **mutate**($C_i$) //$C_i \in U$ is updated. Elitism for free as the fittest $C_1$
      //is excluded from consideration ($i \geq 2$)

Fig. 6. Our genetic algorithm for isometric shape correspondence.

between the first two FPS samples. We start FPS from an extreme point, which is obtained as the farthest point to an arbitrary initial vertex. This scheme automatically and robustly samples the extremities and the center, e.g., $|\varphi| = 6, 7$, and 9 for humanoids, quadrupeds, and centaurs, respectively.

We then create an initial population of 200 random chromosomes each of which is considerably smaller than our original chromosomes of size $N$, namely, each one represents a permutation of size $|\varphi| \ll N$. This initial population is evolved through the same algorithm in Section 4.4 leading to perfect bijections up to symmetric flips. Note that the flipping problem can be alleviated using the collection information (Sahillioğlu and Yemez 2014), if any, or using a symmetry-robust approach such as Liu et al. (2012) and Zhang et al. (2013). A more practical solution than using an external method is re-running our fast genetic algorithm multiple times with the tracked symmetric flips as suggested in Sahillioğlu and Yemez (2013b). Note that two random genes are swapped during mutation in the absence of the **g** descriptor at this stage. Please see the wrapped figure for two initial bijections $\varphi$.

Note that the accuracy of $\varphi$ is crucial for our algorithm, especially in the evaluation of $\mathcal{F}$. We guarantee it (up to symmetric flips) by our planned sampling. The other task of $\varphi$ is in the creation of the initial match candidates (Section 4.3), which are not required to be highly accurate. We also use $\varphi$ during mutation (Figure 7).

## 5 ADAPTIVE SAMPLING ALGORITHM

Besides the main genetic algorithm contribution of this article, we also propose the adaptive sampling algorithm in the sequel, which can improve any correspondence by relocating the target samples

Sample run with 3 substrings to crossover ($N = 17$, indices start at 1):
W: **2 4 7 6 5 1 3 12 17 9 13 10 15 14 11 16 8** //Winner
L: **5 3 1 2 4 7 6 10 9 16 11 15 12 17 13 14 8** //Loser
$s[1] = 7$, $e[1] = 10$ $s[2] = 2$, $e[2] = 4$ $s[3] = 15$, $e[3] = 16$ //start-end

**xover**(Chromosome $W$, Chromosome $L$) //Winner and Loser
  $R = L$ //Result $R$ is initially set to loser $L$
  For $i = 1 : x$ //Has $x$ nonoverlapping random substrings ($x = 3$ here)
   For $j = s[i] : e[i]$ //Set auxiliary arrays $T$ (transfer) and $D$ (delete)
    $T[y] = W[j]$ and $D[y++] = L[j]$ //$y = 1$ initially
    //$T$ = **3 12 17 9 4 7 6 11 16** and $D$ =**6 10 9 16 3 1 2 13 14** now
  //Mark indices in $R$ which would have undesired duplicates if we had
  //directly replaced the loser substrings with the winner substrings ($L'$)
  $m = 1$
  For $i = 1 : |T|$
   If $T[i] \notin D$ //if $T[i] \in D$ then no danger of duplication; no marking
    For $j = 1 : N$
     If $T[i] = R[j]$
      $M[m++] = j$ //marked indices stored at $M$ (mark)
      //$M$ = 13 14 5 6 11 now. These indices point to:
      //$L'$ = **5 4 7 6 4 7 3 12 17 9 11 15 12 17 11 16 8**
  //Update marked indices of $R$ using $D[i] \notin T$ (prevents duplications)
  $m = 1$
  For $i = 1 : |D|$
   If $D[i] \notin T$ //if $D[i] \in T$ then danger of duplication; no update to $R$
    $R[M[m++]] = D[i]$ //first legal option is used
    //$R$ = **5 3 1 2 2 13 6 10 9 16 14 15 10 1 13 14 8** now
  //Copy substrings from $W$ to $R$ safely (no danger of duplication)
  For $i = 1 : x$
   For $j = s[i] : e[i]$
    $R[j] = W[j]$
    //$R$ = **5 4 7 6 2 13 3 12 17 9 14 15 10 1 11 16 8** now
  Return $R$ //Note that loser genes (black) are also preserved in $R$

**mutate**(Chromosome $C$)
  For $i = 1 : N - 1$
   If $d_c(\mathbf{g}_i^s, \mathbf{g}_{C[i]}^t) > \tau$ //See Eq. 3 for $d_c$ ($\tau = .125$)
    //Geodesic vectors **g** are incompatible; swap $C[i]$ with a good $C[j]$
    Repeat $j$ = **rand**($i + 1$, $N$) //random int in $[i + 1, N]$. Left of $i$
    //is already fixed in previous iterations. Pick a good one from right
    Until $d_c(\mathbf{g}_{C[j]}^t, \mathbf{g}_i^s) \leq \tau$
    Swap $C[i]$ and $C[j]$

Fig. 7. Crossover and mutation algorithms that complete Figure 6.

with the current correspondence in mind. In particular, we will improve the bijections produced by our genetic algorithm in Section 4 and Sahillioğlu and Yemez (2012a).

Given a map, not necessarily a bijection, between source and target sample sets $\phi : S \rightarrow T = \{(s_i, t_j)\}$, we aim to compute a new target sample set $\hat{T}$, where $|\hat{T}| = |T|$ and $(s_i, \hat{t}_j)$ is a better correspondence than $(s_i, t_j)$. The first constraint is satisfied naturally by moving $t_j$ to another vertex on the mesh, hence $|\hat{T}| = |T|$. For the latter, we minimize the following energy function whose pairwise term tries to improve the correspondence and singleton term acts as a regularization energy that preserves evenly-spaced sampling:

$$\mathcal{E}(\hat{t}, \phi) = \sum_{(s_i, t_j) \in \phi} \sum_{(s_l, t_m) \in \phi} |d_g(s_i, s_l) - d_g(\hat{t}_j, \hat{t}_m)| + \alpha ||\hat{r} - r_s||, \quad (5)$$

where $r_s$ is the radius of the evenly spaced sampling on the source mesh. The new radius on the target mesh is computed based on the new samples $\{\hat{t}_j\}$ and is asked to look like the radius of the

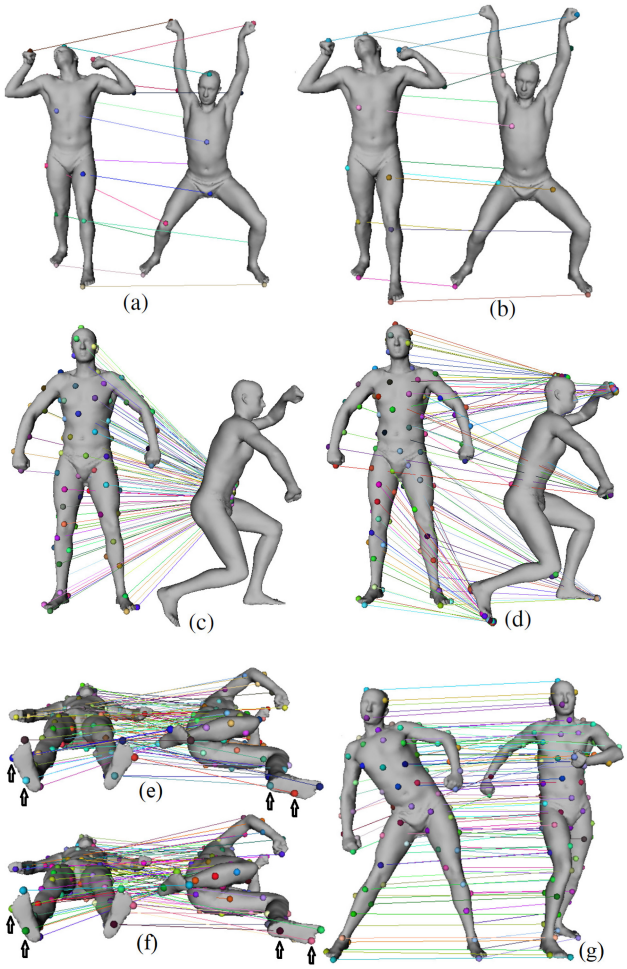



Fig. 8. Improvement of the adaptive sampling (b) to the mapping at (a). Interesting behaviors of the adaptive sampling energy in (c) and (d). Improvements at denser maps, sole without (e) and with (f) adaptive sampling. Map in Figure 4(d) is improved at (g).

source samples as the idea is to move target samples such that they fit much better to the fixed source samples. Radius is given by $\sum_i d_g(s_i, s_c)/|\phi|$, where $s_c$ is the closest sample to $s_i$. Since geodesic distance $d_g$ is normalized into $[0, 1]$, so is radius. We use $\alpha = 0.5$.

We are inspired by the coordinate descent method in the minimization of $\mathcal{E}$ in that at every iteration of the optimization we check whether moving from $t_j$ to a sample $t_k$ in its one-ring neighborhood improves $\mathcal{E}$ or not. If an improvement is observed, then we set $\hat{t}_j = t_k$. For efficiency, we check local improvements only, that is, while searching the one-ring of $t_j$ for an improvement, we consider only the distances between sample pairs that include $t_j$. We show the advantage of this scheme on a sparse map in Figures 8(a) and 8(b), as well as its interesting behavior when $\alpha = 0$ and $d_g(s_i, s_l) = 0$. In this case, we get an attraction effect, whereas we get a repulsion effect when the absolute value is removed, i.e., negative of the distances is minimized, hence maximizing the distances in between (Figures 8(c) and 8(d)). We also show improvements on denser maps in Figures 8(e)–8(g). Besides, we improve another sample-based method, namely, Sahillioğlu and Yemez (2012a), as shown in Figures 14 and 16.

## 6 COMPUTATIONAL COMPLEXITY

Let $V$ be the number of vertices in the original mesh (source or target, whichever has more vertices) and recall that $P$ and $N$ represent the population size and the number of samples to be matched, respectively. We have $P = O(N)$ as we always use $P = 10N$. Genetic algorithm is initialized with the Farthest Point Sampling of $O(NV \log V)$ time complexity, followed by the creation of the initial match candidates set ($O(N|\varphi|)$ per sample) and the assignment from this set ($O(N)$ per chromosome), hence a total of $O(NV \log V + N|\varphi|N + NP) = O(NV \log V + N^2|\varphi|)$.

Genetic algorithm takes $O(G(N \log N + N(N + N|\varphi|) + N(N|\varphi|) + N(N|\varphi|)))$, where $G$ is the number of generations required to complete the evolutionary process. The first term multiplied by $G$ is the sorting of the population that defines the good and bad parts. The second term crossovers two chromosomes in $O(N)$ (Figure 7) and repeats it five times to use the newborn child chromosome yielding the minimum isometric distortion $\mathcal{D}_{\text{iso}}$, which is computed in $O(N|\varphi|)$. We traverse the entire $O(N)$-size bad part for this operation, hence a total of $O(N(N + N|\varphi|))$. Similarly, in the third term, one chromosome in the $O(N)$-size population visits each of its $N$ genes for a potential mutation swap decided based on the $|\varphi|$-size geodesic vector descriptor comparisons (Figure 7), hence $O(N(N|\varphi|))$. The fourth term is due to the search of the fittest chromosome (fifth line in Figure 6). It essentially checks the $\mathcal{D}_{\text{iso}}$ of all chromosomes in the total time of $O(N(N|\varphi|))$. A crucial observation is on the size of the initial bijection $\varphi$, which is constant as we only use the extremity samples and the center sample for $\varphi$. This fact simplifies the overall complexity into $O(GN^2)$, where $G$ never exceeded 250 in our experiments (Section 7.4).

For the adaptive sampling, we observe that $O(N^2)$-time energy functional in Equation (5) can be evaluated through all vertices, which, by aggregate analysis, yields $O(VN^2)$.

## 7 EXPERIMENTS

### 7.1 Datasets

We tested our method on a comprehensive suite consisting of four standard datasets in comparison with the state-of-the-art techniques. The first dataset is a reconstructed pose sequence of a human actor from the SCAPE set (Anguelov et al. 2005), which contains 71 non-uniformly sampled models with the fixed template connectivity, whereas the second one consists of the high-resolution *Cat*, *Centaur*, *David*, *Dog*, *Gorilla*, *Horse*, *Michael*, *Victoria*, and *Wolf* objects from the TOSCA set (Bronstein et al. 2008), each class representing the motion of an articulated object with 11, 6, 7, 9, 4, 8, 12, and 3 meshes, respectively. We also used the SHREC'11 set (Boyer et al. 2011) with 5 high-resolution meshes in each of the *Noise*, *Shotnoise*, *Isometry*, *Sampling*, *Scaling*, and *Viewing* categories. This dataset shows the robustness of our method to various noises, resolution and scaling differences, as well as non-spherical topologies. Finally, we tested our method on the FAUST dataset (Bogo et al. 2014), which has 100 high-resolution scans of

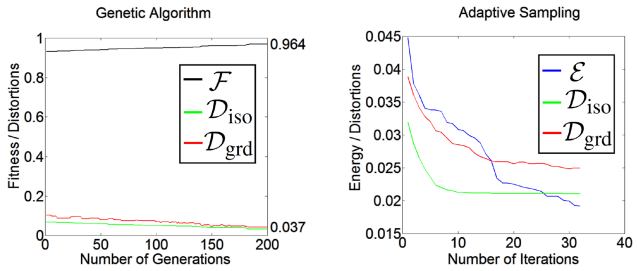



Fig. 9. Plots of various functions while matching a SCAPE pair.

10 human actors and a low-resolution template model registered to each of these scans. This dataset verifies the applicability of our method to real 3D scan data as well as to near isometric pairs by matching not only the intra-subject pairs (different poses of the same actor) but also the inter-subject pairs (different actors).

## 7.2 Evaluation Metrics

In addition to the visuals in Figures 9–19, we also quantify the quality of our resulting genetic maps (Section 4) with and without adaptive sampling (Section 5) using our isometric distortion measure $\mathcal{D}_{\text{iso}}$ as well as the following ground-truth distortion measure:

$$\mathcal{D}_{\text{grd}}(\phi) = \frac{1}{|\phi|} \sum_{(s_i,t_j)\in\phi} d_{\text{g}}(\gamma(s_i), t_j), \tag{6}$$

where $\gamma(s_i)$ the ground-truth correspondence of $s_i$ on the target mesh. Since the geodesic distance $d_{\text{g}}$ is normalized, so is $\mathcal{D}_{\text{grd}}$. Note that $\mathcal{D}_{\text{grd}}$ is used in Figures 3 and 4 as well. To ease comparison to the most of the methods in literature, we use the error-fraction metric in Figures 14 and 16 with the units of the plots and datasets identical to the ones used in (Kim et al. 2011). We match $N = 100$ samples and consider the corresponding subset of matches from the dense methods in our evaluations.

## 7.3 Results

The plots in Figure 9 confirm the reduction of $\mathcal{D}_{\text{grd}}$ as the populations get fitter through new generations. Adaptive sampling takes the final distortion of the genetic algorithm and decreases it further as its $\mathcal{E}$ gets lower. Thanks to the elitism, $\mathcal{F}$ monotonically increases.

We provide further results in Figure 10 to demonstrate our performance under isometric deformations. Figure 11 shows our success on various levels of noise, resolution, and scaling differences, as well as non-spherical topologies. Real 3D scan matching and partial matching examples are shown in Figures 17 and 18, respectively.

*7.3.1 Comparisons.* We compare our method with the Blended Intrinsic Maps (BIM) (Kim et al. 2011), which is able to produce the state-of-the-art dense maps between 3D shapes by combining conformal maps with the interpolated weights varying smoothly over the surface. Approximating geodesic centroids of the blending maps with Euclidean distances may match touching parts of the mesh wrongly (Figure 12(a)). We never encounter such a problem, because we avoid the extrinsic approximations to the intrinsic geodesics used in our algorithm. A hand vertex touching (or close)

Fig. 10. Resulting genetic maps on the TOSCA set. *David* to *Victoria* map shows success for a nearly isometric pair. When isometry deviates significantly, we observe failure, e.g., from the short legs of *Gorilla* to *Victoria*.

to a knee triangle, for instance, does not lead to an erroneous geodesic path from the hand to the knee thanks to the absence of the edge connections between these regions. To emphasize our advantage over BIM on such a common touch scenario, we separate the models that have touching parts in SCAPE and TOSCA *Michael* into the SCAPE-TOUCH and TOSCA-TOUCH sets, respectively. For SCAPE-TOUCH evaluation, we match the single self-touching model (Figure 14(c)) with 20 other SCAPE models. For TOSCA-TOUCH evaluation, we match all of the pairs produced by 20 models, 6 of which exhibit the self-touching situation (Figure 14(d)). We clearly outperform BIM, as well as other competitors, in this scenario (Figure 14(b) and Table 2). In addition to the touching issue, exporting a subset of matches from the dense BIM map may not be as accurate as the explicit computation of those matches in an interpolation-free approach like ours (Figure 12(b)). Note also that BIM needs spherical topology, which makes it fail in the bottom row of Table 1 and in FAUST scans. BIM is also sensitive to triangulation quality (Figure 14) and geometric noise (Table 1). Finally, maps in BIM are not usually onto, which leaves many unmatched target vertices. Note that we suffer from neither of these shortcomings.

In the Expectation-Maximization (EM) algorithm (Sahillioğlu and Yemez 2012a), based on the isometric distortions of the current correspondence (E-step), a better correspondence is obtained by using first a bipartite perfect matching and then a greedy optimization (M-step). Since EM is a sparse correspondence method, just like our method, it can (and does) benefit from the adaptive sampling idea, as shown by EM+AS in Figures 14 and 16. The main drawback of EM is its sensitivity to the initial alignment, which needs to be sufficiently good. Spectral alignment may occasionally give a bad start from which EM cannot recover (Figure 12(c)). Note

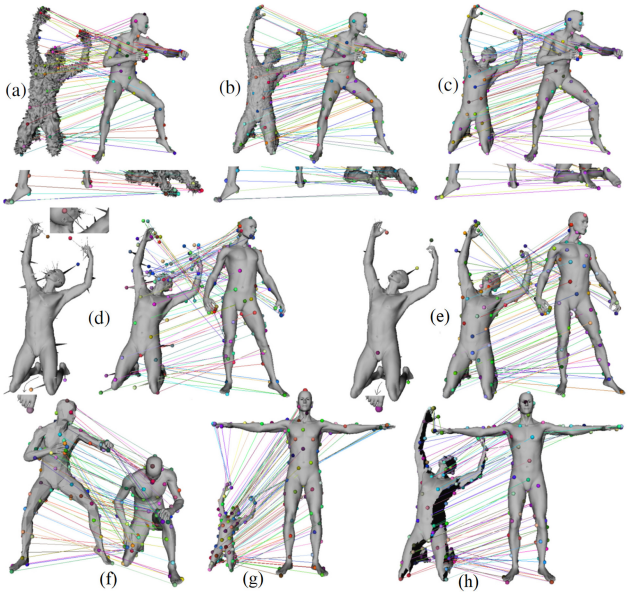



Fig. 11. Highest (a) and lowest (b) levels of *Noise*. No noise for the corresponding *Isometry* pair (c). We require adaptive sampling with a low regularization ($\alpha = 0.1$) to handle the bad sampling of the source in the presence of the highest (d) and lowest (e) levels of *Shotnoise*. Highest levels of *Sampling* (f), *Scaling* (g), and *Viewing* (h).

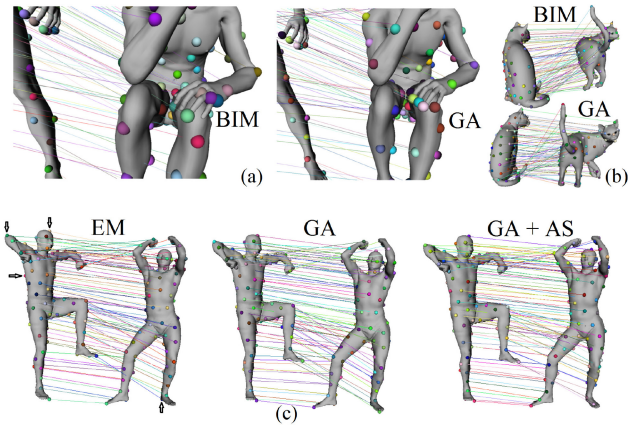



Fig. 12. Hand to knee matching in BIM due to touching surfaces (a). Fingertip matches are also shifted to the palm in BIM. Similar shifts on the cat's back (b). Neither of these problems is observed in our genetic algorithm (GA). GA and GA followed by adaptive sampling (AS) produce better matches (arrows) than EM (c).

that embedding errors are involved in both BIM (extended complex plane embedding) and EM (initial spectral embedding), which we avoid by eliminating such intermediate parameterization domains.

We also compare with the Gromov-Wasserstein (GW) algorithm (Solomon et al. 2016) that uses entropic regularization to map from unstructured data to a regular representation. The optimization procedure that iteratively distributes the distortion works best for isotropic meshes, which is not the case for the non-uniformly sampled SCAPE models (Figure 13(a)). We, as a triangulation-invariant

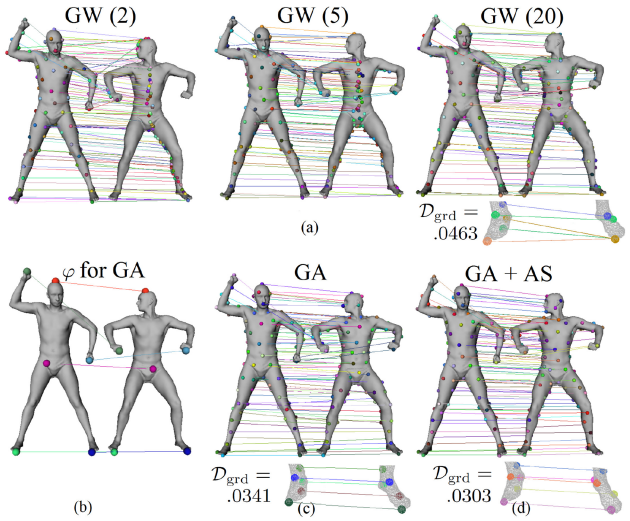



Fig. 13. Number of GW iterations (in parentheses) and the resulting maps (a). Initial bijection $\varphi$ (b) that leads to our GA mapping (c), which is improved further via our adaptive sampling (d). $\mathcal{D}_{\text{grd}}$ values are also provided for three maps. Notice the many-to-one case of GW in the zoomed feet, which may never occur in our bijections.

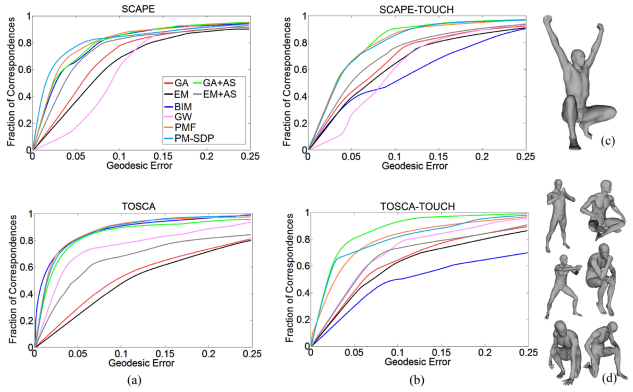



Fig. 14. Error-fraction plots on the SCAPE and TOSCA sets (a) and their subsets (b). GA should be compared with EM and GA+AS with the others.

method, consequently outperform GW (Figures 13(b)–13(d)). Note that GW can also optimize using the barycentric area weights to alleviate the triangulation sensitivity issue, which we enable in our quantitative evaluations (Figures 14 and 16), resulting in similar GW performances for non-uniformly sampled SCAPE and uniformly sampled TOSCA and FAUST. Our performance is still higher on both datasets. We also point out that GW can cope with the lack of isometry and structure to a larger extent than BIM and much larger extent than our method as it distributes the stretch-based distortion more evenly over the surface. Additionally, GW objective is differentiable, a feature that allows it to be used in the gradient-based optimization methods such as a deep network (Ezuz et al. 2017). Similar to BIM, the maps produced by GW are not usually onto (Figure 13(a)), which in turn leaves a significant amount of target vertices unmatched, a problem we avoid in our

Table 1. Quantitative Evaluation of the Maps Produced by Different Approaches

| | GA | EM | GA + AS | BIM | GW |
|---|---|---|---|---|---|
| SCAPE | **(0.027, 0.036)** | (0.032, 0.042) | **(0.015, 0.029)** | (0.025, 0.030) | (0.035, 0.045) |
| TOSCA | **(0.031, 0.041)** | (0.033, 0.043) | (0.016, 0.029) | **(0.026, 0.026)** | (0.027, 0.033) |
| *Dav-Vic* | **(0.030, n/a)** | (0.033, n/a) | **(0.017, n/a)** | (0.030, n/a) | (0.032, n/a) |
| *Gor-Vic* | **(0.061, n/a)** | (0.065, n/a) | **(0.039, n/a)** | (0.046, n/a) | (0.049, n/a) |
| *Noise* | **(0.073, n/a)** | (0.077, n/a) | **(0.070, n/a)** | (0.077, n/a) | (0.075, n/a) |
| *Shotnoise* | **(0.111, n/a)** | (0.136, n/a) | **(0.067, n/a)** | (0.129, n/a) | (0.118, n/a) |
| *Isometry* | **(0.028, n/a)** | (0.028, n/a) | **(0.015, n/a)** | (0.020, n/a) | (0.021, n/a) |
| *Sampling* | **(0.036, n/a)** | (0.037, n/a | **(0.016, n/a)** | (0.036, n/a) | (0.039, n/a) |
| *Scaling* | **(0.027, n/a)** | (0.027, n/a) | **(0.010, n/a)** | (0.017, n/a) | (0.021, n/a) |
| *Viewing* | **(0.049, n/a)** | (0.052, n/a) | **(0.022, n/a)** | (n/a, n/a) | (0.021, n/a) |

Each entry is an ordered pair representing $(\mathcal{D}_{\text{iso}}, \mathcal{D}_{\text{grd}})$. For SHREC'11, we use the highest levels of each category. *Dav-Vic* is short for *David-Victoria*. It is fair to compare GA with EM (columns 2 and 3), and GA+AS with BIM and GW (columns 3, 4, and 5). The best performing method w.r.t. $D_{\text{grd}}$ (or $D_{\text{iso}}$ when $D_{\text{grd}}$ is unavailable) is written in bold.

Table 2. Quantitative Evaluation of the Maps in the Style of Table 1

| | GA + AS | EM+AS | PMF | PM-SDP |
|---|---|---|---|---|
| SCAPE | **(0.015, 0.029)** | (0.030, 0.036) | (0.026, 0.030) | (0.028, 0.031) |
| TOSCA | (0.016, 0.029) | (0.030, 0.039) | (0.020, 0.028) | **(0.026, 0.027)** |
| SCAPE-TOUCH | **(0.015, 0.0281)** | (0.031, 0.036) | (0.016, 0.0284) | (0.024, 0.0282) |
| TOSCA-TOUCH | **(0.025, 0.026)** | (0.029, 0.037) | (0.020, 0.028) | (0.025, 0.027) |

bijections. Finally, note that we rounded the GW maps to permutations by picking the index of the maximum entry for each row in their resulting fuzzy matrix.

In Table 1, we quantify the quality of the maps generated by our genetic algorithm (GA), genetic algorithm followed by our adaptive sampling (GA+AS), BIM, EM, and GW. For the BIM, EM, and GW, we used the source codes provided by their authors. Note that, it is fair to compare GA with EM (as they are restricted to the fixed joint sampling of two meshes), and GA+AS with the others (as they are allowed to pick any mesh vertex in their correspondences).

Recent dense correspondence methods are also employed in our comparison suite using their public source codes. PMF (Vestner et al. 2017) and PM-SDP (Maron et al. 2016) perform slightly better than our method (Figures 14 and 16 and Table 2) at the expense of significantly increased computation time (Table 3). There are also some datasets, such as SCAPE, SCAPE-TOUCH and TOSCA-TOUCH, where we compare favorably. Note that, as we switch from TOSCA to TOSCA-TOUCH (Figure 14 and Table 2), we see that our results improve more than that of PMF and PM-SDP, the purely intrinsic methods that are also robust to self-touching situations. Our improvement can be explained by the exclusion of the *Gorilla*, *David*, and *Victoria* classes in the TOSCA-TOUCH set, which are relatively problematic for our method but not for PMF and PM-SDP.

We finally note our light comparison with the sample-based sparse method of (Tevs et al. 2011), which plans ahead the informative samples for a robust isometric matching. Amongst the four SCAPE maps received from one of the authors, we show in Figure 15 the only one without the symmetric flip problem.

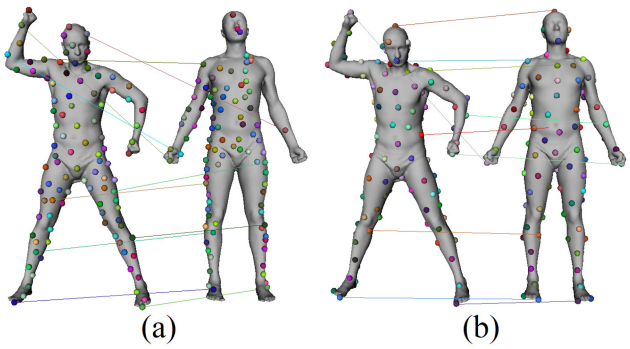



Fig. 15. Map of (Tevs et al. 2011) (a) in comparison with ours (b). The former matches 125 samples with $D_{\text{grd}} = 0.067$. We match the same number of samples with $D_{\text{grd}} = 0.029$. Some matches are highlighted with lines.

Entropy-based planning of the former is likely to produce unevenly-spaced sampling, whereas we do provide even spacing and perform better.

## 7.4 Timings

On an 8GB 3.4GHz 64-bit PC, the average execution time of our implementation over 50 random SCAPE pairs (12.5K vertices each) is 4s for the computation of $N = 100$ samples plus 9s of the genetic algorithm initialized by a population of size 1,000 plus 5s for the adaptive sampling, hence a total of 13s when the optional adaptive sampling is discarded. BIM creates its dense map in 72s, whereas EM produces the sparse map of the same size ($N = 100$) in 5s. *Dog* with 25K vertices is matched in 9s (initial sampling) + 9s (genetic algorithm) + 12s (adaptive sampling), and in 234s with BIM, and 10s with EM. *David* with 52K vertices takes only 21 + 14 + 45s to complete with our method. SCAPE, *Dog*, and *David* requires, respectively, 200, 200, and 250 generations on average to complete.

Note that, our genetic algorithm is independent of the size of the mesh vertices as it stresses only on the samples (Section 6). For a sparse correspondence, this feature makes it preferable over dense pipelines, which process *all* vertices in the form of some $V \times V$ matrices, e.g., kernel density matrix in PMF (Vestner et al. 2017), metric measure matrix in GW (Solomon et al. 2016), positive semi-definite matrix in PM-SDP (Maron et al. 2016), per-triangle second-order cone constraints in SM (Aigerman et al. 2015), as well as smaller but still computationally expensive blending matrix in BIM (Kim et al. 2011) and basis functions matrix in FM (Ovsjanikov et al. 2012). A dense matching algorithm naturally provides the sparse correspondence as a subset of its result set, but at the expense of more time and space demands, e.g., using their public code one GW iteration of cubic complexity takes 337s for a SCAPE pair in our PC and it requires at least 20 of them (Figure 13(a)). Similarly, the public codes of PMF and PM-SDP require 1050 and 1710s, respectively, in our PC for the same pair, and SM reports 1380s of computation for a total of 11K vertices, which is about half the amount of vertices for a SCAPE pair. Recall our corresponding execution time of 13 seconds (see Table 3 for a summary). Note also that, we run GW on the decimated TOSCA and SHREC'11 models of 15K vertices due to its speed problems.

Table 3. Execution Times of Various Algorithms to Match 100 Sample Points on Two Shapes Having 12.5K Vertices Each

| GA | GA + AS | EM | BIM | GW | PMF | PM-SDP | SM |
|---|---|---|---|---|---|---|---|
| 13 | 18 | 5 | 72 | 6,740 | 1,050 | 1,710 | >2,760 |

First three columns are for the sparse correspondence methods, whereas the rest is for the dense matching algorithms. Times are given in seconds.

Table 4. Performance of Various Algorithms in the Form of ($\mathcal{D}_{\text{iso}}$, $\mathcal{D}_{\text{grd}}$, Seconds) Triplets while Matching 100 Samples on FAUST

| | GA + AS | OTE | OTE by GA Init |
|---|---|---|---|
| Intra-subject | (0.014, 0.027, 12.7) | (0.033, 0.042, 0.16) | (0.032, 0.043, 0.30) |
| Inter-subject | (0.018, 0.032, 12.5) | (0.035, 0.044, 0.15) | (0.033, 0.044, 0.28) |

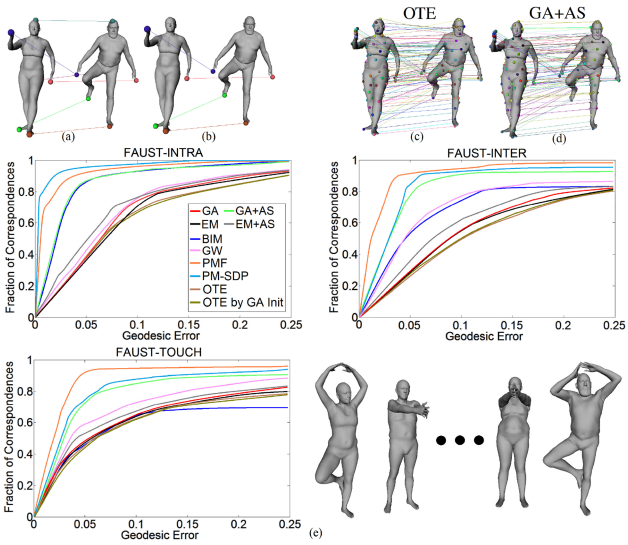



Fig. 16. Our initial bijection on a FAUST inter-subject pair (a) provides the four landmark matches (b) for the OTE algorithm (c). Our result for the same pair is more accurate (d). Error-fraction plots over 45 intra-subject, inter-subject, and self-touching pairs are also provided (e).

## 7.5 Applications

Among many graphics tasks that may benefit from our fast and robust genetic algorithm, we demonstrate three adaptations, namely, dense matching, registration, and partial matching, with proof-of-concept examples validating the stability of the proposals.

*7.5.1 Initializing a Dense Pipeline.* We take the dense matching algorithm (Aigerman and Lipman 2015), OTE for short, which computes harmonic parameterizations of the spherical meshes into the Euclidean orbifolds efficiently by solving a sparse linear system. Using their public code, we first compare our sparse matches with the corresponding subset in their dense maps on the FAUST dataset. We then replace the manual selection of four landmarks in this approach by the four landmarks computed with our method, hence making OTE fully automatic. The computation based on our landmark matching, which are the first four matches of our initial bijection (Section 4.5), reveals almost the same output as the original OTE does, since they also manually select similar landmarks (see Figure 13 of OTE). We should, however, note that our automatization works only for the isometric (or nearly isometric) shape pairs, narrowing down the large application scope of OTE. An isometric dense matching pipeline such as FM, however, may be initialized with our sparse maps without any limitations on its application domain. Other non-isometric pipelines that require initial sparse maps include PMF, SM, and the recent Ezuz and Ben-Chen (2017).

The execution time of OTE is 0.16s on 7K-vertices models of FAUST. Automatic landmark sampling and matching by our algorithm brings 0.14s of extra time. We finally note that OTE, while being very fast for a dense map producer, is not as accurate as our algorithm (Figure 16 and Table 4). Their accuracy loss is mostly due to the inclusion of an extra embedding domain, which we avoid. Our comparative FAUST performance is also given in Figure 16. Our method is on a par with BIM for intra-subjects. For inter-subjects as well as FAUST-TOUCH, however, we perform better than BIM, where the latter is a subset of FAUST consisting of 10 self-touching models. All other methods in our test suite are clearly outperformed on FAUST-INTRA, -INTER, and -TOUCH.

Yet another important dense pipeline is the registration of high-resolution real-world 3D scan data. Landmarks in correspondence can alleviate the stability problems of the registration algorithms (Allen et al. 2003; Maron et al. 2016; Pauly et al. 2005; Sahillioğlu and Kavan 2015). With this motivation, we evaluate our sparse landmark correspondence on the FAUST benchmark, which provides challenging real-world scans. Promising results increase the impact of our method. Note that all methods discussed thus far are too slow to handle the 190K-vertices scans of FAUST, except BIM and OTE, which are also not applicable to this matching scenario due to the non-sphere topology of the input with holes. Our algorithm takes merely 6s for the initial sampling plus 0.14s for the computation of the initial bijection, which is already sufficient for the most of the registration algorithms (Figure 17(a)). A sparse map of size 100 takes 96 seconds of initialization plus 20 seconds of genetic algorithm (Figure 17(b)). Over 40 FAUST scan pairs, we obtain an average $\mathcal{D}_{\text{iso}}$ of 0.035. Note also our promising results on the related SHREC'11 *Viewing* dataset (Figure 11(h)), which synthetically represents scans with larger holes.

Note that, the learning-based methods such as Wei et al. (2016) can handle scan matching in a more stable manner than our purely geodesic-based approach. Our relative instability can be explained with an example concerning two samples in the scan mesh of Figure 11(h), namely, the cyan sample in the chest and the purple one around the armpit. Due to the missing black region in between, the geodesic distance between these two will be very high compared to the one computed on a full model. This problem does not, however, apply to all the pairs, e.g., the geodesic between the left and the right pinky fingers is not affected by the missing region. Consequently, we expect performance degradation proportional to the amount of missing data, as verified by our tests on *Viewing* in Table 5.

*7.5.2 Partial Isometric Matching.* We take the partial isometric matching algorithm (Sahillioğlu and Yemez 2012b), PIM for short, which performed well on a recent partial matching contest (Cosmo et al. 2016). By minimizing a novel scale-invariant

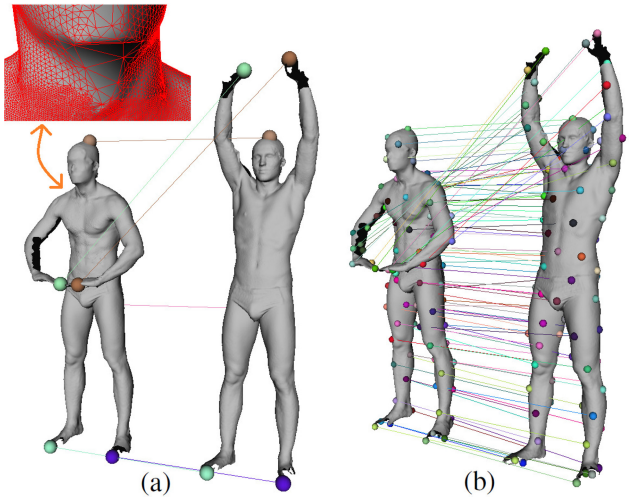



Fig. 17. Our initial bijection (a) and sparse correspondence (b), either of which can be incorporated into the registration process of these scans. Neither the holes nor the nonuniform triangulation (zoomed) prevents our algorithm from producing these plausible maps.

Table 5. Scan Matching Performance of Our Genetic Algorithm under Different Percentages of Missing Data

| | Level 1 | Level 2 | Level 3 | Level 4 | Level 5 |
|---|---|---|---|---|---|
| % of hole | 13 | 14 | 25 | 27 | 28 |
| $\mathcal{D}_{\text{iso}}$ | 0.033 | 0.034 | 0.040 | 0.042 | 0.044 |

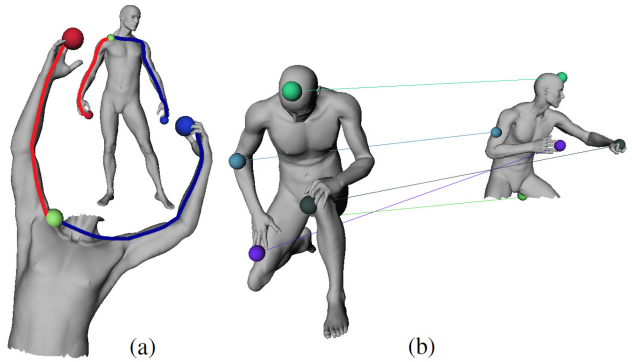



Fig. 18. Distortion measure of PIM is inspired by the fact that the ratios between blue geodesics and red geodesics are the same as three matches involved are all good (a). Our genetic algorithm efficiently minimizes this distortion to match an arbitrarily scaled partial model from SHREC'11 *Partial* with a full SHREC'11 *Isometry* model. Note that the touching surfaces around hands and knees do not degrade our performance (b).

isometric distortion measure that is based on the preservation of geodesic distance ratios (Figure 18(a)), PIM solves the partial correspondence problem in a setting where one of the shapes to be matched is an arbitrarily scaled isometric part of the other. The measure that is cubic in the map size is evaluated $\binom{|F|}{|P|}|P|!$ times via combinatorial search over all possible mappings, where $F$ and $P$ are the sample sets over the full and partial shapes, respectively.

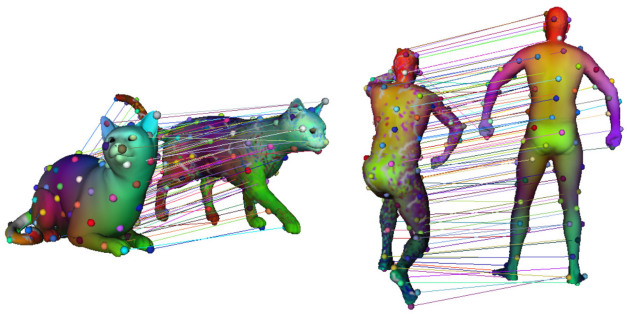


Fig. 19. Our resulting sparse map (spheres and lines) is interpolated trivially into a dense one by simply assigning a source vertex (non-sphere) to a target vertex with the most similar geodesic distances to the samples of the existing sparse map. Result is not as smooth as the dense map of the BIM.

For tractability, PIM uses $|F| = 10$ and $|P| = 5$, which, using their public code, takes 0.6 seconds of search time on a pair of SHREC'11 models, each with 50K vertices. To ensure that $F$ includes all the samples in $P$, it is suggested to increase $|F|$ to 20, in which case the search time becomes 37 seconds. Replacing the combinatorial search with our genetic search reaches the same optimum in 0.2 and 1.9s, respectively, as the latter explores the space of permutations more wisely without visiting every possibility (Figure 18(b)). During our genetic optimization, we evolve chromosomes consisting of $|F|$ genes, $|P|$ of which represent the samples on the partial shape and the remaining $|F| - |P|$ are dummy entries corresponding to the unmatched samples on the full shape.

The 599 shapes in Cosmo et al. (2016) are based on the TOSCA dataset, some are cut with a plane (cuts set) and some are punctured (holes set). We use PIM to match 100 random cuts-meshes and 100 random holes-meshes with their full base models. We observe that the minimization of the PIM's distortion with the original scheme of Sahillioğlu and Yemez (2012b) and with our genetic algorithm gives very close results on average, which in turn fairly lends the qualitative and quantitative (ground-truth) evaluations in Cosmo et al. (2016) to this article. The only significant difference is the execution time, which favors the genetic algorithm.

## 8 CONCLUSION

We presented a fast and robust 3D isometric shape correspondence algorithm, every step of which is geometrically intuitive and easily comprehensible, e.g., the standard C library is enough to implement every step without including an external support such as a linear algebra library. We minimized the isometric distortion without using an intermediate parameterization domain; hence, our solution is free of embedding errors. The minimization is carried out with genetic optimization, a framework that fits well for the essential permutation creation task of all the correspondence problems. To this end, we provided a genetic algorithm geared towards the isometric correspondence problem. Our method requires no initial input matches, is insensitive to surface triangulation, self-touching situations, and mild geometric noise, and can also be applied to shapes with arbitrary genus and resolution.

As a second contribution, we provided an adaptive sampling strategy that is able to improve a given correspondence by

iteratively moving the matched samples on one side of the correspondence.

The comprehensive evaluation on four standard benchmarks in comparison with state-of-the-art methods validated our technique. Specifically, our algorithm compares favorably with BIM, while being significantly faster. We outperform EM, which is faster, as well as GW and OTE, which are more versatile than our method. Other methods in our comparison suite, namely, PMF, PM-SDP, and SM, are much slower as they are designed for dense maps.

## 9 LIMITATIONS

Failure cases arise under topological noise and significant deviation from isometry. This is mainly because of the drastic and unpredictable geodesic distance changes that make two shapes unmatchable under our geodesic preservation rules (fitness in Section 4.2).

Partial matching support is brought only after altering the distortion measure and introducing the dummy entries. In the original setting, however, we cannot decide a consistent maximum geodesic distance across the partially similar shapes, which breaks down our scale normalization operation. If the partially similar shapes are assumed to be at the correct scale beforehand (a strong assumption), or if the fitness is computed based on a scale-invariant isometric distortion measure (as done in Section 7.5.2), then partial similarity will not be a limitation any more.

## 10 FUTURE WORK

Genetic optimization is a convenient tool to establish isometric correspondences, as shown in this article. We believe that it can be adapted to other scenarios such as dense, non-isometric, partially-isometric, and collectionwise-consistent correspondences. We indeed took the initiative for the dense matching (Section 7.5.1) and the partially-isometric matching (Section 7.5.2) extensions. Our work can also benefit from the exact geodesics (Xu et al. 2015) that would replace the currently active Dijkstra's shortest paths restricted to the mesh edges. Finally, a new interpolation scheme can be developed to achieve a smoother dense map based on a sparse one (Figure 19).

## ACKNOWLEDGMENTS

I thank the anonymous reviewers for their valuable comments and Prof. Wand for the help with his paper's experiments.

## REFERENCES

Y. Aflalo, A. Dubrovina, and R. Kimmel. 2016. Spectral generalized multi-dimensional scaling. *Int. J. Comput. Vision* 118, 3 (2016), 380–392.

N. Aigerman, S. Kovalsky, and Y. Lipman. 2017. Spherical orbifold tutte embeddings. *ACM Trans. Graph.* 36, 4 (2017), 90.

N. Aigerman and Y. Lipman. 2015. Orbifold tutte embeddings. *ACM Trans. Graph.* 34, 6 (2015), 190–1.

N. Aigerman and Y. Lipman. 2016. Hyperbolic orbifold tutte embeddings. *ACM Trans. Graph.* 35, 6 (2016), 217.

N. Aigerman, R. Poranne, and Y. Lipman. 2015. Seamless surface mappings. *ACM Trans. Graph.* 34, 4 (2015), 72.

B. Allen, B. Curless, and Z. Popovic. 2003. The space of human body shapes: Reconstruction and parameterization from range scans. *ACM Trans. Graph.* 22, 3 (2003), 587–594.

D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. 2005. Scape: Shape completion and animation of people. *ACM Trans. Graph.* 24, 3 (2005), 408–416.

F. Bogo, J. Romero, M. Loper, and M. Black. 2014. FAUST: Dataset and evaluation for 3D mesh registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3794–3801.

D. Boscaini, J. Masci, E. Rodolà, and M. Bronstein. 2016. Learning shape correspondence with anisotropic convolutional neural networks. In *Advances in Neural Information Processing Systems*. 3189–3197.

J. Bowers, R. Wang, L. Wei, and D. Maletz. 2010. Parallel Poisson disk sampling with spectrum analysis on surfaces. *ACM Trans. Graph.* 29, 6 (2010), 166.

E. Boyer, A. M. Bronstein, M. M. Bronstein, B. Bustos, et al. 2011. SHREC 2011: Robust feature detection and description benchmark. In *Proceedings of the Eurographics Workshop on 3D Object Retrieval*.

A. M. Bronstein, M. M. Bronstein, and R. Kimmel. 2006. Generalized multidimensional scaling: A framework for isometry invariant partial surface matching. *Proc. Natl. Acad. Sci. U.S.A.* 103, 5 (2006), 1168–1172.

A. M. Bronstein, M. M. Bronstein, and R. Kimmel. 2008. *Numerical Geometry of Non-Rigid Shapes*. Springer.

M. M. Bronstein and I. Kokkinos. 2010. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1704–1711.

Q. Chen and V. Koltun. 2015. Robust nonrigid registration by convex optimization. In *Proceedings of the International Conference on Computer Vision*. 2039–2047.

C. K. Chow, H. T. Tsui, and T. Lee. 2004. Surface registration using a dynamic genetic algorithm. *Pattern Recogn.* 37, 1 (2004), 105–117.

Luca Cosmo, Emanuele Rodolà, Michael M. Bronstein, Andrea Torsello, Daniel Cremers, and Yusuf Sahillioğlu. 2016. SHREC'16: Partial matching of deformable shapes. *Proc. 3DOR* 2, 9 (2016), 12.

A. D. Cross, R. C. Wilson, and E. R. Hancock. 1997. Inexact graph matching using genetic search. *Pattern Recogn.* 30, 6 (1997), 953–970.

K. Delibasis, P. Asvestas, and G. Matsopoulos. 2010. Multimodal genetic algorithms-based algorithm for automatic point correspondence. *Pattern Recogn.* 43, 12 (2010), 4011–4027.

J. Duda and M. Jakiela. 1997. Generation and classification of structural topologies with genetic algorithm speciation. *J. Mech. Design* 119, 1 (1997), 127–131.

Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi. 1997. The farthest point strategy for progressive image sampling. *IEEE Trans. Image Process.* 6 (1997), 1305–1315.

D. Ezuz and M. Ben-Chen. 2017. Deblurring and denoising of maps between shapes. *Comput. Graph. Forum* 36, 5 (2017), 165–174.

D. Ezuz, J. Solomon, V. Kim, and M. Ben-Chen. 2017. GWCNN: A metric alignment layer for deep shape analysis. *Comput. Graph. Forum* (2017).

N. Fish, O. van Kaick, A. Bermano, and D. Cohen-Or. 2016. Structure-oriented networks of shape collections. *ACM Trans. Graph.* 35, 6 (2016), 171.

V. Ganapathi-Subramanian, B. Thibert, M. Ovsjanikov, and L. Guibas. 2016. Stable region correspondences between non-isometric shapes. *Comput. Graph. Forum* 35, 5 (2016), 121–133.

N. Gelfand, L. Ikemoto, S. Rusinkiewicz, and M. Levoy. 2003. Geometrically stable sampling for the ICP algorithm. In *Fourth International Conference on 3-D Digital Imaging and Modeling*. 260–267.

C. Goh, E. Teoh, and K. C. Tan. 2008. Hybrid multiobjective evolutionary design for artificial neural networks. *IEEE Trans. Neural Netw.* 19, 9 (2008), 1531–1548.

I. Harvey. 2009. The microbial genetic algorithm. In *Proceedings of the European Conference on Artificial Life*. 126–133.

Q. Huang, B. Adams, M. Wicke, and L. Guibas. 2008. Non-rigid registration under isometric deformations. *Computer Graphics Forum* (2008), 1149–1458.

V. Jain and H. Zhang. 2006. Robust 3D shape correspondence in the spectral domain. In *Proceedings of the IEEE International Conference on Shape Modeling and Applications (SMI'06)*. 118–129.

M. Kazhdan, J. Solomon, and M. Ben-Chen. 2012. Can mean-curvature flow be modified to be non-singular? *Comput. Graph. Forum* 31, 5 (2012), 1745–1754.

K. Khoo and P. Suganthan. 2001. Multiple relational graphs mapping using genetic algorithms. *Evolution. Comput.* 2 (2001), 727–733.

V. Kim, Y. Lipman, and T. Funkhouser. 2011. Blended intrinsic maps. *ACM Trans. Graph.* 30, 4 (2011).

A. Kovnatsky, M. Bronstein, X. Bresson, and P. Vandergheynst. 2015. Functional correspondence by matrix completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 905–914.

Zorah Lahner, Emanuele Rodolà, Michael Bronstein, Daniel Cremers, Oliver Burghard, Luca Cosmo, Andreas Dieckmann, Reinhard Klein, and Yusuf Sahillioğlu. 2016. SHREC'16: Matching of deformable shapes with topological noise. In *Proceedings of the Eurographics Workshop on 3D Object Retrieval*.

Y. Lipman and T. Funkhouser. 2009. Möbius voting for surface correspondence. *ACM Trans. Graph.* 28, 3 (2009).

O. Litany, T. Remez, E. Rodolà, A. M. Bronstein, and M. M. Bronstein. 2017a. Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the International Conference on Computer Vision*.

O. Litany, E. Rodolà, A. Bronstein, and M. Bronstein. 2017b. Fully spectral partial shape matching. *Comput. Graph. Forum* 36, 2 (2017), 247–258.

O. Litany, E. Rodolà, A. Bronstein, M. Bronstein, and D. Cremers. 2016. Non-rigid puzzles. *Comput. Graph. Forum* 35, 5 (2016), 135–143.
T. Liu, V. G. Kim, and T. Funkhouser. 2012. Finding surface correspondences using symmetry axis curves. *Comput. Graph. Forum* 31, 5 (2012).
H. Maron, N. Dym, I. Kezurer, S. Kovalsky, and Y. Lipman. 2016. Point registration via efficient convex relaxation. *ACM Trans. Graph.* 35, 4 (2016), 73.
H. Maron, M. Galun, N. Aigerman, M. Trope, N. Dym, E. Yumer, V. Kim, and Y. Lipman. 2017. Convolutional neural networks on surfaces via seamless toric covers. *ACM Trans. Graph.* 36, 4 (2017), 71.
R. Myers and E. R. Hancock. 2001. Least-commitment graph matching with genetic algorithms. *Pattern Recogn.* 34, 2 (2001), 375–394.
D. Nehab and P. Shilane. 2004. Stratified point sampling of 3D models. In *Proceedings of the Eurographics Symposium on Point-Based Graphics*. 49–56.
D. Nogneng and M. Ovsjanikov. 2017. Informative descriptor preservation via commutativity for shape matching. *Comput. Graph. Forum* 36, 2 (2017), 259–267.
M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas. 2012. Functional maps: A flexible representation of maps between shapes. *Proc. SIGGRAPH* (2012).
M. Ovsjanikov, Q. Huang, and L. Guibas. 2011. A condition number for non-rigid shape matching. *Comput. Graphics Forum* 30, 5 (2011), 1503–1512.
M. Ovsjanikov, Q. Mérigot, F. Mémoli, and L. Guibas. 2010. One point isometric matching with the heat kernel. *Comput. Graph. Forum* 29, 5 (2010), 1555–1564.
M. Pauly, N. Mitra, J. Giesen, M. Gross, and L. Guibas. 2005. Example-based 3D scan completion. In *Symposium on Geometry Processing*. 23–32.
M. Pilat and C. Jacob. 2008. Creature academy: A system for virtual creature evolution. *Proceedings of the IEEE World Congress on Computational Intelligence*. 3289–3297.
J. Pokrass, A. Bronstein, M. Bronstein, P. Sprechmann, and G. Sapiro. 2013. Sparse modeling of intrinsic correspondences. *Comput. Graph. Forum* 32, 2 (2013), 459–468.
E. Rodolà, A. Bronstein, A. Albarelli, F. Bergamasco, and A. Torsello. 2012. A game-theoretic approach to deformable shape matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 182–189.
E. Rodolà, S. Bulo, T. Windheuser, M. Vestner, and D. Cremers. 2014. Dense non-rigid shape correspondence using random forests. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4177–4184.
E. Rodolà, M. Moeller, and D. Cremers. 2017. Regularized pointwise map recovery from functional correspondence. *Comput. Graph. Forum* 36, 8 (2017), 700–711.
Yusuf Sahillioğlu and Ladislav Kavan. 2015. Skuller: A volumetric shape registration algorithm for modeling skull deformities. *Med. Image Anal.* 23, 1 (2015), 15–27.
Yusuf Sahillioğlu and Yücel Yemez. 2010. 3D shape correspondence by isometry-driven greedy optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 453–458.
Yusuf Sahillioğlu and Yücel Yemez. 2011. Coarse-to-fine combinatorial matching for dense isometric shape correspondence. *Comput. Graph. Forum* 30, 5 (2011), 1461–1470.
Yusuf Sahillioğlu and Yücel Yemez. 2012a. Minimum-distortion isometric shape correspondence using EM algorithm. *IEEE Trans. PAMI* 34, 11 (2012), 2203–2215.
Yusuf Sahillioğlu and Yücel Yemez. 2012b. Scale normalization for isometric shape matching. *Comput. Graph. Forum* 31, 7 (2012).
Yusuf Sahillioğlu and Yücel Yemez. 2013a. Partial 3D correspondence from shape extremities. *Comput. Graph. Forum* 33, 6 (2013), 63–76.
Yusuf Sahillioğlu and Yücel Yemez. 2013b. Coarse-to-fine isometric shape correspondence by tracking symmetric flips. *Comput. Graph. Forum* 32, 1 (2013), 177–189.
Yusuf Sahillioğlu and Yücel Yemez. 2014. Multiple shape correspondence by dynamic programming. *Comput. Graph. Forum* 33, 7 (2014), 121–130.
L. Silva, O. Bellon, and K. Boyer. 2005. Precision range image registration using a robust surface interpenetration measure and enhanced genetic algorithms. *IEEE Trans. PAMI* 27, 5 (2005), 762–776.
K. Sims. 1994. Evolving virtual creatures. *Proc. SIGGRAPH* (1994).
J. Solomon, G. Peyre, V. Kim, and S. Sra. 2016. Entropic metric alignment for correspondence problems. *ACM Trans. Graph.* 35, 4 (2016), 72.
K. O. Stanley and R. Miikkulainen. 2002. Evolving neural networks through augmenting topologies. *Evolution. Comput.* 10, 2 (2002), 99–127.
P. Suganthan. 2002. Structural pattern recognition using genetic algorithms. *Pattern Recogn.* 35, 9 (2002), 1883–1893.
S. Tao, Z. Huang, B. Zuo, Y. Peng, and W. Kang. 2012. Partial retrieval of CAD models based on the gradient flows in Lie group. *Pattern Recogn.* 45, 4 (2012), 1721–1738.
A. Tevs, A. Berner, M. Wand, I. Ihrke, and H.-P. Seidel. 2011. Intrinsic shape matching by planned landmark sampling. *Comput. Graph. Forum* 30, 2 (2011), 543–552.
O. van Kaick, H. Zhang, and G. Hamarneh. 2013. Bilateral maps for partial matching. *Comput. Graph. Forum* 32, 6 (2013), 189–200.
O. van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or. 2011. A survey on shape correspondence. *Comput. Graph. Forum* 30, 6 (2011), 1681–1707.
M. Vestner, R. Litman, E. Rodolà, A. Bronstein, and D. Cremers. 2017. Product manifold filter: Non-rigid shape correspondence via kernel density estimation in the product space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6681–6690.
Y. Wang. 2006. Feature point correspondence between consecutive frames based on genetic algorithm. *Int. J. Roboti. Automat.* 21, 1 (2006), 35.
L. Wei, Q. Huang, D. Ceylan, E. Vouga, and H. Li. 2016. Dense human body correspondences using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1544–1553.
Z. Xi, Y. Kim, J. Kim, and J. Lien. 2016. Learning to segment and unfold polyhedral mesh from failures. *Comput. Graph.* (2016), 139–149.
C. Xu, T. Y. Wang, Y.-J. Liu, L. Liu, and Y. He. 2015. Fast wavefront propagation (FWP) for computing exact geodesic distances on meshes. *IEEE Trans. Visual. Comput. Graph.* 21, 7 (2015), 822–834.
K. Xu, H. Zhang, D. Cohen-Or, and B. Chen. 2012. Fit and diverse: Set evolution for inspiring 3D shape galleries. *ACM Trans. Graph.* 31, 4 (2012), 57.
S. Yamany, M. Ahmed, and A. Farag. 1999. A new genetic-based technique for matching 3-D curves and surfaces. *Pattern Recogn.* 32, 10 (1999), 1817–1820.
Z. Zhang, K. Yin, and K. Foong. 2013. Symmetry robust descriptor for non-rigid surface matching. *Comput. Graph. Forum* 32, 7 (2013), 355–362.
C. Zhu, R. Yi, W. Lira, I. Alhashim, K. Xu, and H. Zhang. 2017. Deformation-driven shape correspondence via shape recognition. *ACM Trans. Graph.* 36, 4 (2017), 51.

Received July 2017; revised June 2018; accepted June 2018