Geometric Analysis of Shape Variability of Lower Jaws of Prehistoric Humans

Jing Ren, Peter Wonka

King Abdullah University of Science and Technology (KAUST)

Gowtham Harihara, Maks Ovsjanikov

LIX, Ecole Polytechnique, UMR CNRS

Abstract

In this document we describe our method and the results obtained for comparing jaws of prehistoric humans. Our main goal was twofold: 1) establish a methodology for comparing the structure of 3D shapes of scans of jaws using geometric data analysis techniques, and 2) use this methodology for comparing and clustering individual objects according to their geometric similarity. Moreover, we also applied geometric modeling techniques to establish a "clean" version of the dataset, without significant artefacts present in the original data, such as large missing parts. We then applied our analysis techniques both on the original and the clean dataset in order to validate our comparison results. For our core task of shape comparison, we used recent state-of-the-art shape matching methods and we present similarity results using different comparison metrics.

Keywords: Geometry Processing, Shape Matching.

1. Introduction and Context

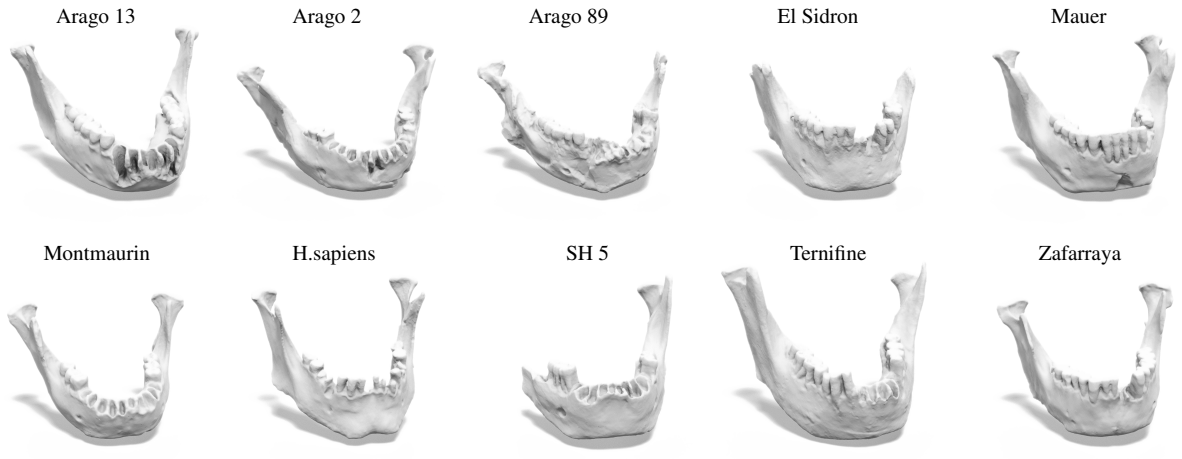

Figure 1: Overview of the 10 jaws.

We are given ten 3D scans of jaws of prehistoric humans, shown in Fig. 1. Given this dataset, our main goal is to establish *geometric similarity metrics* for comparing different shapes in a robust and accurate manner. This is challenging for several reasons.

First, we note that the dataset contains a lot of noise. For example, some jaws have some teeth while others do not. Furthermore, the shape "SH 5" has a very significant missing part. The shapes "Arago 89" and "El Sidron" have

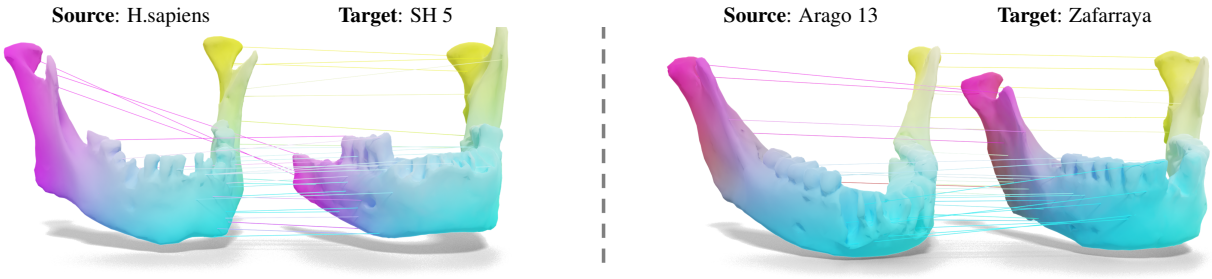

Figure 2: Visualization of the correspondences between shape pairs.

smaller but still significant parts missing. The shape "Arago 89" has very severe artefacts, potentially due to strong damage to the object itself.

Second, the definition of a *meaningful similarity metric* is a challenging problem in itself, lacking a single clearly specified objective. This is due to several factors: we are not given any ground truth annotations of shapes or parts that are either clearly similar or clearly dissimilar; further, there is no canonical deformation model between the shapes; finally these issues are significantly exacerbated by the partiality (missing parts) and artefacts in the data.

Third, the dataset is very small, compared to commonly used datasets and benchmarks in geometric data analysis. This makes it very difficult to draw conclusions from individual observations, and completely precludes any use of learning-based methods. This limited size also has an adverse effect on the robustness of our results. For example, changing a single parameter can lead to significantly different clustering results. Finally, the very significant geometric variability, coupled with the small size of the dataset makes shape clustering particularly challenging.

Below we describe our general approach for dealing with these challenges and results obtained using several existing techniques. At the same time, we emphasize that these results are not conclusive, and have a rather suggestive nature. As we demonstrate, current geometric data analysis techniques are mature and can be adapted to deal with strong variability and used to define several robust and geometrically meaningful similarity metrics. However, the limited size of the dataset is a fundamental challenge that can only be overcome with more example shapes.

Nevertheless, several interesting similarity results emerge from our analysis, as described below.

2. Methodology

Our goal is to find the differences (or similarities) between the given ten jaws. Our general pipeline consists of the following steps:

1. For each pair of shapes, find correspondences between points on their surfaces, using a non-rigid shape matching method.
2. Compute the optimal rigid alignment of each pair of shapes, using the correspondences from step 1. Use this alignment to define the Rigid Alignment Measure for each pair of shapes.
3. Compute the optimal non-rigid deformation, trying to *morph* between each pair of shapes, using the correspondences from step 1. Use this deformation to define the Shape Distortion Measure for each pair of shapes.
4. For each of the two dissimilarity metric above, compute a hierarchical clustering, and a 2D embedding of the dataset, that preserves this metric as well as possible.

Below we describe each step in this pipeline. Further technical details are provided in the Appendix.

2.1. Shape Correspondences

We used the method introduced in [1] to compute the point-wise correspondences between every pair of jaws. To avoid errors arising due to the missing data, we manually specified 22 landmark points on each shape (see Fig. 9 in the Appendix), which were used to guide the correspondence. In the Appendix we also present results computed without any landmarks on the cleaned dataset. Note that we use the manually specified landmarks only in this step of the pipeline for computing approximate correspondences. All other steps in our analysis pipeline are completely agnostic to these landmarks and operate on all points of the scans.

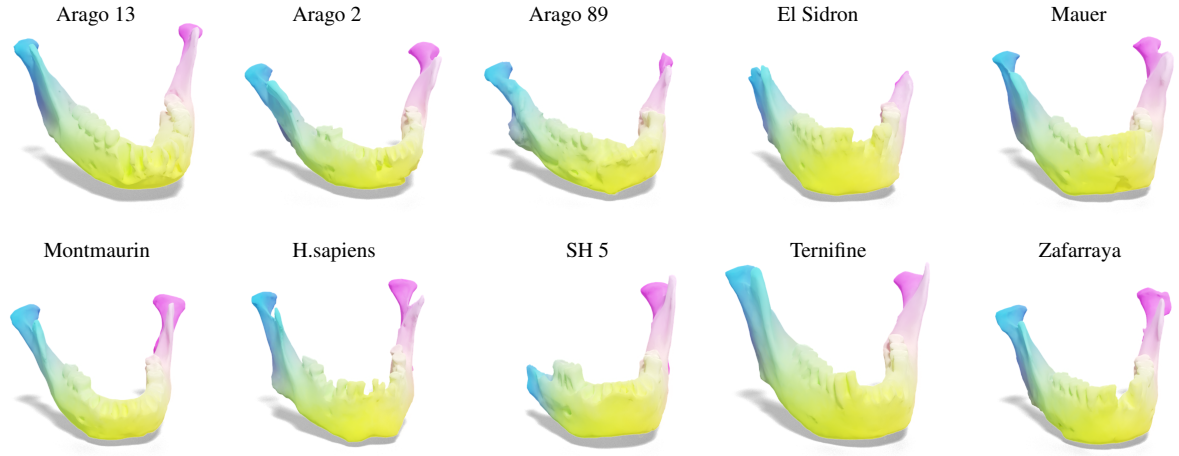

Figure 3: Computed correspondences (mappings) across all the jaws. The correspondences are encoded by color: regions with the same color on different jaws are in correspondence.

Fig. 2 illustrates the computed correspondences two shape pairs. Specifically, if a vertex on the target shape corresponds to a vertex on the source shape (like the vertices connected by a line in the figure), they will be assigned the same color.

Fig. 3 shows the computed correspondences across all the jaws. Specifically, two vertices from different jaws with the same color are corresponding to each other.

In the Appendix we also show the cycle consistency measure associated with the computed correspondences, which highlights the problematic areas on each shape.

2.2. *Rigid Alignment and Metric*

The shapes in the original dataset are not well aligned (e.g., consider "H.sapiens" and "SH 5" shapes shown in Fig. 4), making it hard to compare their geometry. Therefore, we first rigidly align each pair of shapes using the standard Iterative Closest Point (ICP) method [2]. ICP is an iterative method, which computes the optimal rotation and translation to align one shape to another given some initial correspondence. We use the correspondences obtained in the previous step as initialization to ICP.

As an example, the position of "SH 5" after rigid alignment is shown in Fig. 4. Note that the *shape* of "SH 5" is not modified but only its position in space is changed. We provide the exact algorithm used for rigid alignment in the Appendix for completeness.

2.2.1. *Rigid Alignment Measure*

Once each pair of shapes S_1, S_2 is rigidly aligned, we define the *Rigid Alignment Measure* (RAM) metric as follows:

$$\mathrm{RAM}(S_1, S_2) = \sum_{x \in S_1} w_x \|x - NN_{S_2}(x))\|, \text{ where } NN_{S_2}(x) = \arg\min_{y \in S_2} \|x - y\|. \tag{1}$$

In other words, we measure the weighted sum of the distances between each point x on shape S_1 and its nearest neighbor (NN) on S_2. We emphasize that both the points x in the sum and the points y in the nearest neighbor computation involve all points on the shapes, and not only the small set of landmarks. The weight w_x measures the "reliability" of point x in measuring similarity. We use it primarily to discount missing regions, since they should not strongly penalize dissimilarity between shapes that are well-aligned. The exact definition is given in the Appendix.

To obtain a symmetric metric we average across both pairs of shapes. I.e., we define our rigid-based shape distance: $D_R(S_1, S_2) = \mathrm{RAM}(S_1, S_2) + \mathrm{RAM}(\mathrm{S}_2, \mathrm{S}_1)$.

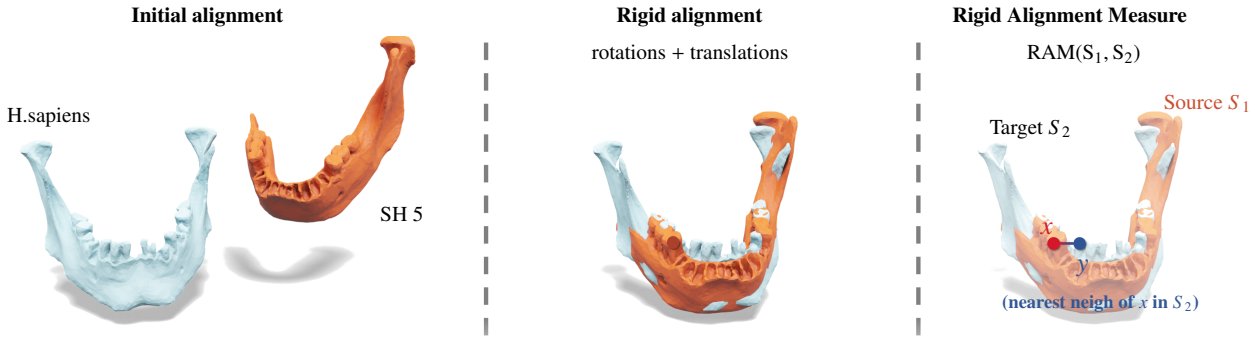

Figure 4: Rigid alignment and metric.

2.3. *Non-rigid Deformation*

Our Rigid Alignment Measure captures the Euclidean distances between the points on the rigidly aligned shapes. This quantifies the differences between the positions of points on the shapes, but provides no information on the changes to the shapes themselves.

To measure the changes in shape *structure*, we introduce another measure, based on *non-rigid* shape alignment. Namely, rather than only allowing rotation and translation to align the two shapes we compute a transformation that morphs one shape into another. We then use this transformation to define another shape dissimilarity metric that intuitively measures the elastic deformation energy required to perform this morphing.

In order to compute the optimal non-rigid transformation we use a variant of the ARAP deformation method [3]. Similarly to ICP, this is also an iterative optimization technique, but crucially, it allows arbitrary shape deformations, and not only rigid motions. It also relies on initial correspondences between each pair of shapes, for which we use the maps computed in step 1 of our pipeline.

The details of the exact method we use for non-rigid shape deformation are provided in the Appendix.

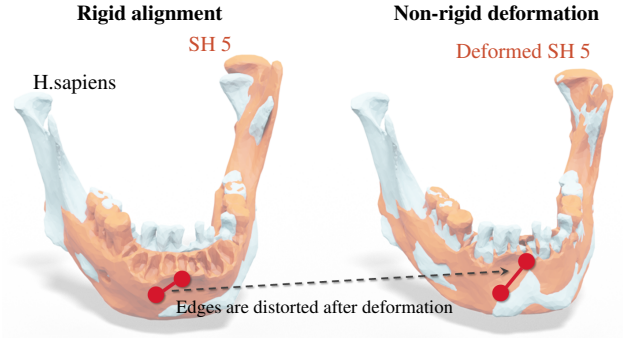

Figure 5: Shape Distortion Measure.

Fig. 5 (right) shows an example of "SH 5" deformed into "H.sapiens". As can be seen from 5 (left), the right ramus and the alveolar part of "SH 5" are far from those of the "H .sapiens". After the non-rigid deformation, these two parts fit more closely. Of course, the non-rigid deformation induces a distortion between the original and the deformed shape. Our goal is to define a measure of this distortion in order to compare each pair of shapes in the dataset.

2.3.1. *Shape Distortion Measure*

Given the source shape S_1 and the target shape S_2, assume that we deform S_1 to fit with S_2. To define our shape distortion measure we consider the relative change of pairwise distances between points on the S_1. Namely if we

denote by V and Z the coordinates of the vertices of S_1 before and after the deformation respectively, our Shape Distortion Measure (SDM) is defined as follows:

$$\mathrm{SDM}(S_1, S_2) = \sum_{v_i, v_j \in S_1} w_{ij} \left(\frac{\|Z_i - Z_j\|}{\|V_i - V_j\|} - 1 \right)^2 . \tag{2}$$

The sum in this expression is over all pairs of vertices on shape S_1, where for efficiency we consider only vertices connected by an edge in the triangle mesh. The weights w_{ij} intuitively measure how reliable the pair v_i, v_j is. The details are provided in the Appendix. Overall, the Shape Distortion Measure quantifies how much stretching or shrinking is necessary to deform S_1 onto S_2. Note that a purely rigid deformation would induce zero error, since in that case the edge lengths would remain the same.

Fig. 5 illustrates how SDM is defined. Specifically, we measure how much each edge gets deformed, and sum over all edges of the mesh to define the Shape Distortion Measure as the overall deformation metric.

Since SDM is not symmetric and because a distance metric is supposed to be symmetric, we use the average of two distortion measures d to define the shape distance:

$$D_E(S_1, S_2) = \frac{1}{2}(\mathrm{SDM}(S_1, S_2) + \mathrm{SDM}(S_2, S_1)). \tag{3}$$

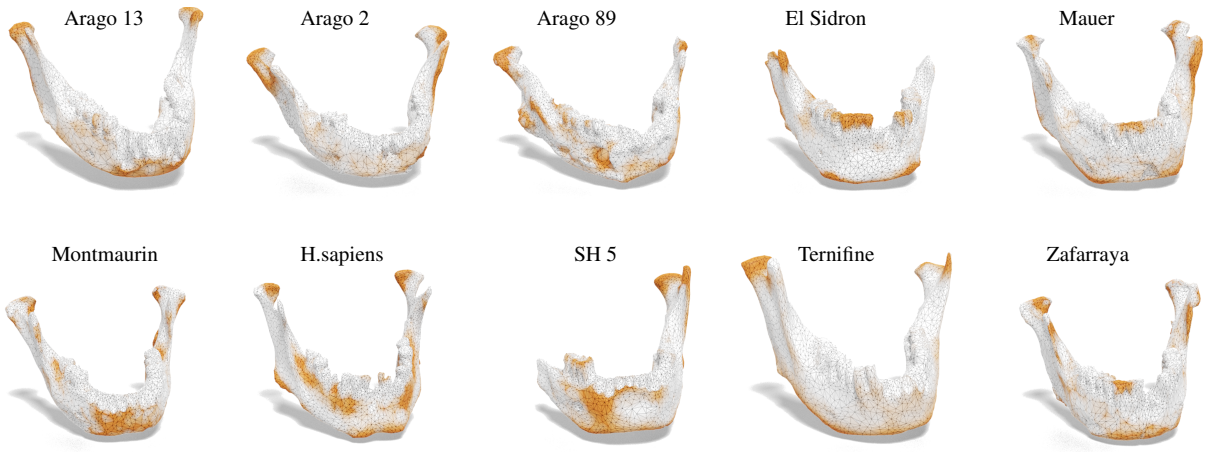

Figure 6: Average Rigid Alignment Measure on each shape. Intuitively, this highlights the regions on each shape that make it different or unique, compared to others in the dataset.

3. Global Dataset Analysis

Given the measures of similarity defined above, we use the the pairwise dissimilarity between each pair of shapes for our global analysis.

Specifically we analyze the dataset in three ways:

1. We visualize the average rigid alignment measure by color-coding them on the points of each shape. This intuitively highlights the points and regions that make each shape unique, compared to others in the dataset.
2. We compute a 2D visualization of the dataset, where each point corresponds to a particular shape. This helps to understand how different shapes relate to each other.
3. We compute a hierarchical cluster tree using our two dissimilarity metrics. This helps get a sense what pairs or groups of shapes are related.

Note that methods 2. and 3. above depend on the choice of dissimilarity. Therefore, we present two separate results for each (using our RAM and SDM-based shape metrics).

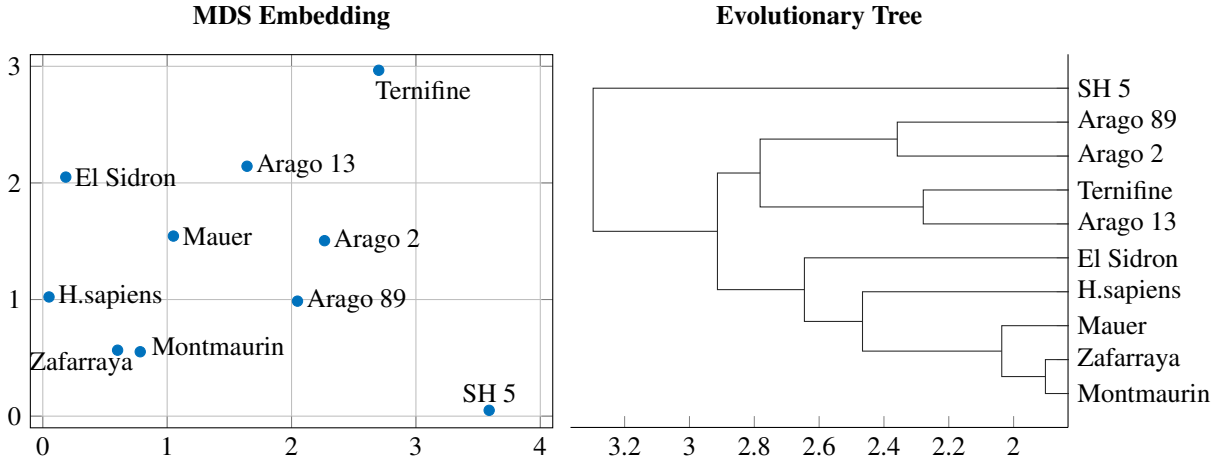

Figure 7: Global dataset analysis using Rigid Alignment Measure.

3.1. Average Shape Dissimilarity

First, for each shape, we compute the average of its rigid alignment measure to the other nine shapes and visualize the error per vertex as shown in Fig. 6. The more highlighted a region is (i.e., corresponding to the brighter red color), the farther it is from the other shapes in the dataset. Note that the rigid alignment measure is weighted by the correspondence accuracy. Therefore, for the regions that correspond to the missing part might not be highlighted since they are assigned with a small weight.

Fig. 6 shows the average dissimilarity on all shapes in our dataset. We note several regions that are commonly highlighted: the mental protuberance, the condyle and the coronoid process. This corresponds to the fact that these regions have a lot of variability in the dataset. Thus, the unique features of each shape are often a combination of particular structure of these three features. Second, on some shapes such as "H.sapiens" and "Montmaurin", parts of the body are highlighted as well. In those cases, these regions seem to unique structure due to their width and relative positions.

Finally, we note that "Arago 89" and "SH 5" are shapes with strong damage. Therefore, we do not assign significant confidence to the results on these shapes.

3.2. Shape Relations and Clustering

Our next goal is to analyze the relative global similarity of the shapes in the dataset. For this, we use the two metrics we defined (Rigid Alignment Measure and Shape Distortion Measure) first to establish a representation of the overall similarity of shapes in the dataset. Specifically, for each dissimilarity metric, we first compute a 10x10 matrix, that captures the dissimilarity for each pair of shapes. We use these distance matrices in two ways: first to obtain a 2D embedding using Multi-dimensional scaling (MDS) [4], which visually presents the dataset in two dimensions, while trying to respect the computed pairwise dissimiarity measures. Second, we build an evolutionary tree using a hierarchical clustering algorithm.

Specifically, we assign each jaw a 2D-coordinate such that the Euclidean distance between two points in Fig. 7 (or Fig. 8) approximates the shape distance between the two shapes, where the shape distance is derived from the Rigid Alignment Measure or Shape Distortion Measure. Therefore, the further two points in Fig. 7 and 8 are, the dissimilar the representing two jaws are, and vice versa.

From the MDS embeddings we can see that for both metrics, the three Arago jaws are in close-by locations. Also "H.sapiens", "Zafarraya", and "Montmaurin" are relatively close to each other forming in a cluster. We also note that the "Ternifine" is far from other shapes for both metrics. By considering Fig. 6 we observe that the overall shape of "Ternifine" jaw is very different from the others, due its unique features in the coronoids and the lower base.

Finally, we also compute the hierarchical clustering based on our two measures of dissimilarity (RAM and SDM). Fig. 7 and 8 (right) show the clustering computed from the corresponding to the two measures. We use standard

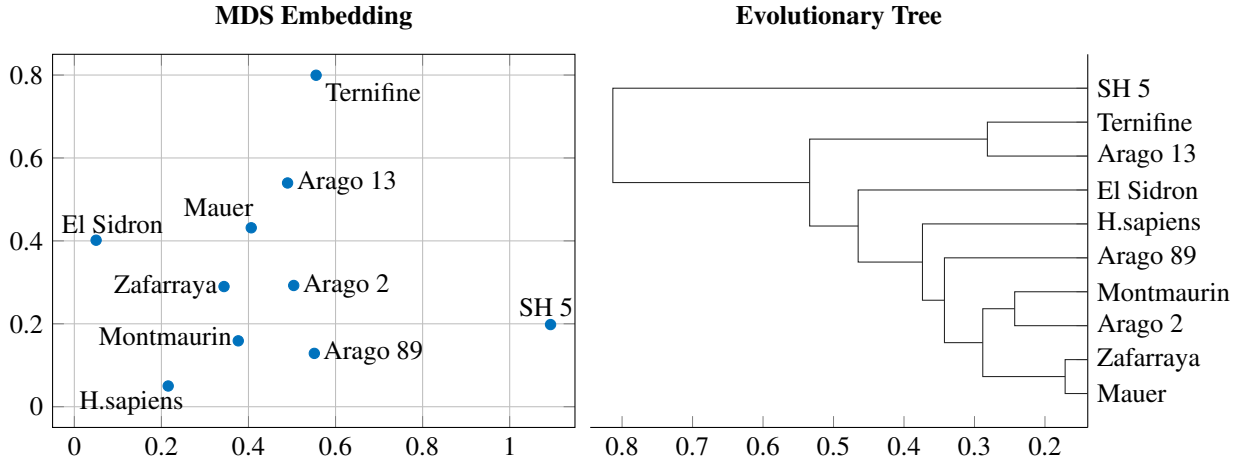

Figure 8: Global dataset analysis using Shape Distortion Measure.

single-linkage clustering (with the average distance criterion). We observe that the clustering generally respects the conclusions drawn from our other analysis. Specifically "SH 5" and "Ternifine" are far from the other shapes, while Arago are similar to each other and the remaining shapes form a very loose general cluster.

References

[1] J. Ren, A. Poulenard, P. Wonka, M. Ovsjanikov, Continuous and orientation-preserving correspondences via functional maps, ACM Transactions on Graphics (TOG) 37 (6).

[2] P. J. Besl, N. D. McKay, Method for registration of 3-d shapes, in: Sensor fusion IV: control paradigms and data structures, Vol. 1611, International Society for Optics and Photonics, 1992, pp. 586–606.

[3] O. Sorkine, M. Alexa, As-rigid-as-possible surface modeling, in: Proceedings of the fifth Eurographics symposium on Geometry processing, Eurographics Association, 2007, pp. 109–116.

[4] T. F. Cox, M. A. Cox, Multidimensional scaling, Chapman and hall/CRC, 2000.

[5] W. Kabsch, A solution for the best rotation to relate two sets of vectors, Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography 32 (5) (1976) 922–923.

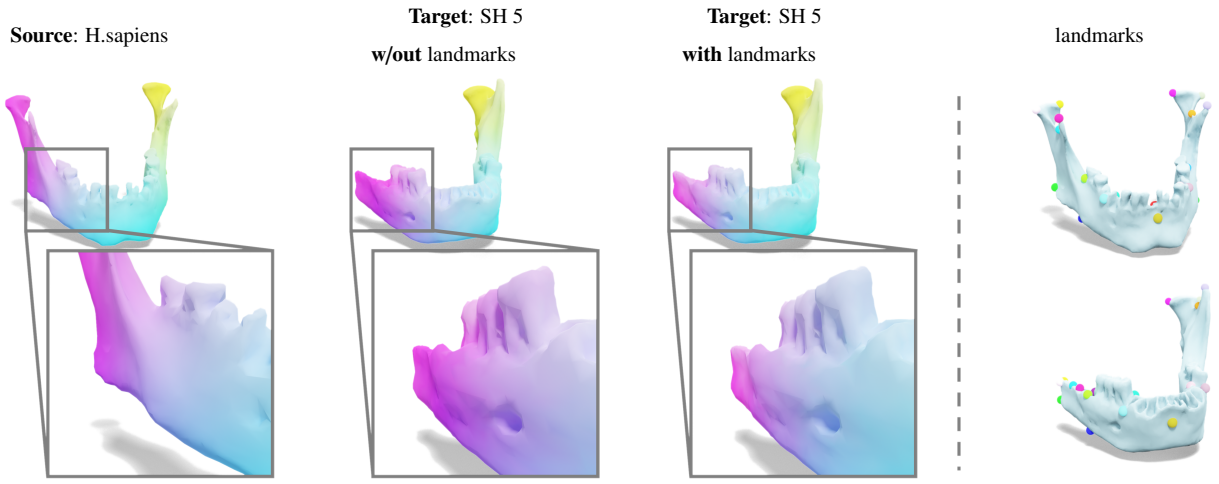

Figure 9: Here we show two computed maps between the shape "H.sapiens" and "SH 5", where the corresponding regions are in the same color. The second column visualizes the map computed without any landmarks. We can see that due to the partiality of the shape "SH 5", the left ramus of "H.sapiens" is wrongly mapped to left body of "SH 5". To avoid it, we manually selected 22 landmarks (as shown on the rightmost column), and compute the point-wise base on these landmarks. The obtained map shown in the third column is more accurate.

4. Appendix

4.1. Landmarks for correspondences

Figure 9 (right) shows the landmark points we used for computing pairwise correspondences. Figure 9 (left) also shows the correspondences obtained without using landmarks via color-coding. Note that while generally reasonable, due to strong partiality, the correspondences without landmarks can have large distortion. In Section 4.5 below shows the results of our analysis pipeline without using any landmarks on the remodeled dataset, where artefacts and partiality are corrected.

4.2. Cycle consistency of computed correspondences

We report the cycle consistency of the computed maps, which is a commonly-used measure of accuracy in the absence of ground truth correspondences. Specifically, given three shapes S_i, S_j, and S_k, and the computed maps T_{ij}, T_{jk}, and T_{ki}, we compute the composite map $T_{ki} \circ T_{jk} \circ T_{ij}$. This is a map from S_i to itself. The cycle consistency error

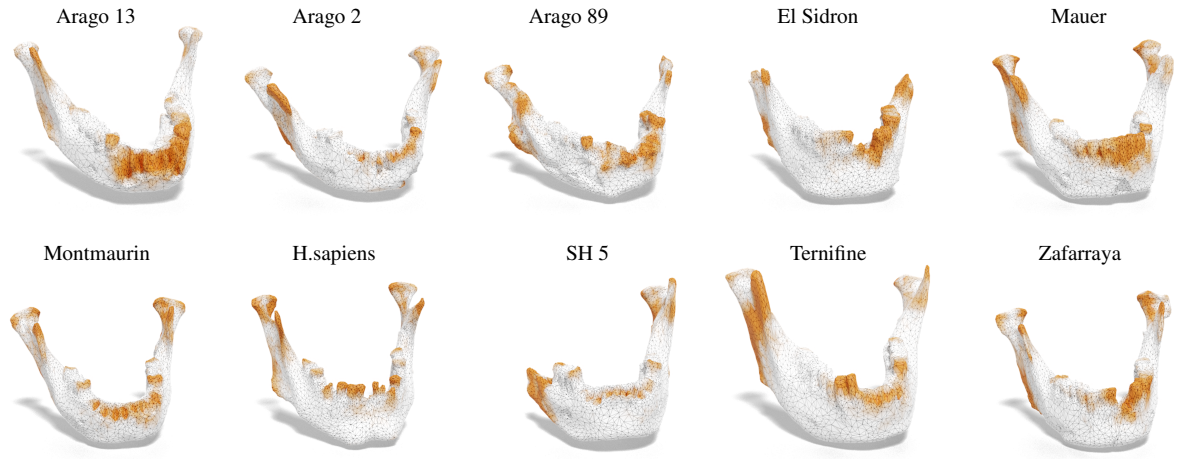

Figure 10: Cycle consistency of the compute maps visualized in Fig 3.

measures the difference between this composite map and the identity map. By averaging this metric over all possible choices of (k, j), we obtain an error defined on each shape S_i. Following this idea, we computed and visualized the cycle consistency error on the complete dataset on each of the jaw in Fig. 10. The regions with large inconsistency across the dataset are colored orange. We can see that the inconsistent regions are mainly the teeth and the ramus of the jaws, which corresponds to regions that are commonly missing.

To alleviate this issue, we remodeled the jaws so that the teeth are removed, and the missing parts of the ramus are completed. More details are given in Section 4.5 below.

4.3. Rigid Alignment Algorithm

Algorithm 1 describes the procedure used for computing the optimal rigid alignment between a source shape X_1 and a target shape X_2 given some initial correspondence T_{12}. Note that the optimal rotation and translation are given in closed form [2, 5] with matrix Singular Value Decomposition.

Algorithm 1 Rigid Alignment

Input X_1, X_2, T_{12}
Output X_1^*: rigid transformation of X_1
$X_1^{\text{prev}} \leftarrow X_1$
$R, t \leftarrow \arg\min_{R,t} \|X_1^{\text{prev}} * R + t - X_2(T_{12}, :)\|$ ▷ closed-form solution
$X_1^{\text{curr}} \leftarrow X_1^{\text{prev}} R + t$
while $\|X_1^{\text{prev}} - X_1^{\text{curr}}\| \geq \epsilon$ **do**
    $T_{12} \leftarrow \text{knnsearch}(X_2, X_1^{\text{curr}})$
    $X_1^{\text{prev}} \leftarrow X_1^{\text{curr}}$
    $R, t \leftarrow \arg\min_{R,t} \|X_1^{\text{prev}} * R + t - X_2(T_{12}, :)\|$
    $X_1^{\text{curr}} \leftarrow X_1^{\text{prev}} R + t$
return $X_1^* \leftarrow X_1^{\text{curr}}$

4.3.1. Rigid Alignment Weights

Algorithm 2 Correspondence Weights

Input X_1, X_2, T_{12}
Output w_{12}: weights for the given correspondences
$X_1^* \leftarrow$ Rigid Alignment(X_1, X_2, T_{12}) ▷ Algorithm 1
$nn_{12} \leftarrow \text{knnsearch}(X_2, X_1^*)$, $nn_{21} \leftarrow \text{knnsearch}(X_1^*, X_2)$
$T_{11} \leftarrow nn_{21} \circ nn_{12}$, $T_{22} \leftarrow nn_{12} \circ nn_{21}$
$e_1 \leftarrow d_{S_1}(T_{11}, Id)$, $e_2 \leftarrow d_{S_2}(T_{22}, Id)$
$e \leftarrow e_1 + e_2(T_{12})$
$w_{12} \leftarrow \exp(-e)$
$w_{12} \leftarrow \frac{w_{12} - \min(w_{12})}{\max(w_{12}) - \min(w_{12})}$
return w_{12}

Recall that our *Rigid Alignment Dissimilarity* (RAD) metric is defined as:

$$\text{RAD}(\text{S}_1, \text{S}_2) = \sum_{x \in S_1} w_x \|x - NN_{S_2}(x))\|, \text{ where } \text{NN}_{\text{S}_2}(\text{x}) = \arg\min_{\text{y} \in \text{S}_2} \|\text{x} - \text{y}\|. \tag{4}$$

The weight w_x which measures the reliability of the vertex x in our similarity metric is defined in Algorithm 2. Fig. 11 illustrates the errors e_1, e_2, e that are used in Algorithm 2 to define the correspondence weights. Note that these weights also provide confidence for per-vertex deformation in the next step (see Sec. 4.4).

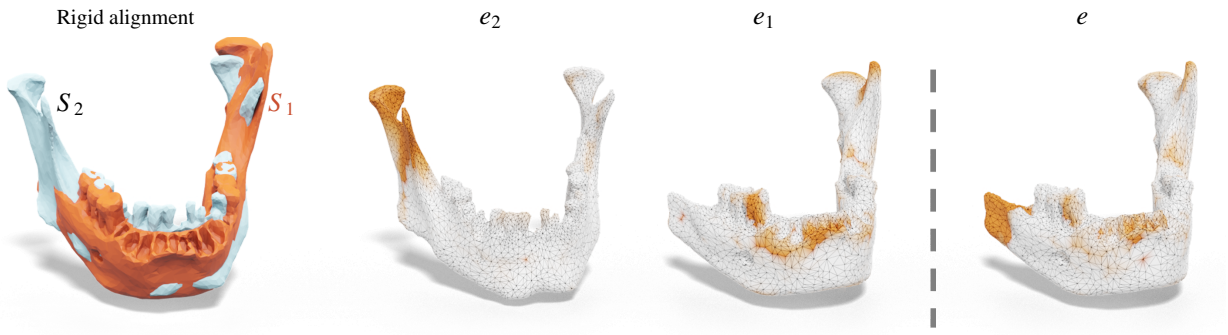

Figure 11: Illustration of the errors used to define the correspondence weights in Algorithm 2.The highlighted regions with color orange suggest that these vertices do not have trustful correspondences and should be assigned with relatively **small** weight for deformation and similarity measurement.

4.4. Details on non-rigid shape deformation

Given two shapes S_1 and S_2, where S_1 has vertex positions stored in X, and S_2 has vertex positions stored in Y. The goal is to assign the vertices in S_1 with new positions Z, such that the deformed S_1 looks similar to S_2.

Assume that we are given initial correspondences T_{12} between S_1 and S_2. Vertex positions Z are the variables we want to solve for. Following [3] we define the following energy:

$$E(Z, R_{\text{global}}, t_{\text{global}}, R^{(i)}_{\text{local}} | X, Y, T_{12}, w_{12}) = a_1 E_1 + a_2 E_2 + a_3 E_3 \tag{5}$$

where

$$E_1 = \left\| (Z - Y(T12, :)) \text{diag}(w_{12}) \right\|_F^2 \tag{6}$$

$$E_2 = \left\| Z - (R_{\text{global}} X + t_{\text{global}}) \right\|_F^2 \tag{7}$$

$$E_3 = \sum_i \sum_{j \in \mathcal{N}(i)} \left\| (Z_i - Z_j) - R^{(i)}_{\text{local}} (X_i - X_j) \right\|_F^2 \tag{8}$$

Specifically, the first term shows that after the deformation of S_1 into target shape S_2, the vertex positions Z is close to the vertex positions Y of the target shape according to the given correspondences T_{12}. The weight w_{12} assigns a confidence weight to each correspondence in T_{12}, which means for the vertices with a large weight, we trust them more for deformation. Algorithm 2 explains how to compute the weights from the given correspondences T_{12}.

The second term of Eq. 5 is the global rigidity, that the global deformation is close to rigid deformation. Therefore, we can obtain Z by just applying some rotation R_{global} and some translation t_{global}. The third term is the local rigidity introduced in [3], to ensure that the deformation of each triangle is close to rigid transformation.

By minimizing the total energy $E(Z)$ over iterations, we can obtain the deformation of S_1 into target shape S_2. Algorithm 3 shows how to iteratively update the deformed positions Z and other variables.

4.5. Mesh cleaning

We remodeled the original meshes and got some clean meshes as shown in Fig. 12. Specifically, we removed all the teeth from the meshes. For the partial shapes, we used its symmetric shape to complete the missing parts. Also the surfaces are smoothed to remove the noises. Note that the two heads of the shape “El Sidron” are still missing, since we do not have any information about this part from the original shape itself.

Since the missing parts are completed, we do not need landmarks to guide the map computation. We simply apply the same method as used in the original dataset but without any landmarks. Fig. 13 shows the computed maps on the 10 remodeled meshes. Again, the regions with the same color correspond to each other according to our compouted maps. Similarly, Fig.14 and 15 visualize the overall similarity of the remodelled shapes using the metrics Rigid Alignment Measure and Shape Distortion Measure respectively.

Algorithm 3 Shape Deformation

Input X_1, X_2, T_{12} maxIter
Output Z: deformed positions of X_1
$Z \leftarrow$ Rigid Alignment(X_1, X_2, T_{12}) ▷ Algorithm 1
for $1 \leq$ iter $\leq$ maxIter **do**
  $w_{12} \leftarrow$ Correspondence Weights (X_1, X_2, T_{12}) ▷ Algorithm 2
  $R_{\text{global}}, t_{\text{global}} \leftarrow \underset{R_{\text{global}}, t_{\text{global}}}{\arg\min}\ E(R_{\text{global}}, t_{\text{global}} \,|\, X_1, X_2, T_{12}, w_{12}, Z)$ ▷ Close-form solution
  $R^{(i)}_{\text{local}} \leftarrow \underset{R^{(i)}_{\text{local}}}{\arg\min}\ E(R^{(i)}_{\text{local}} \,|\, X_1, X_2, T_{12}, w_{12}, Z)$
  $Z \leftarrow \underset{Z}{\arg\min}\ E(Z \,|\, R_{\text{global}}, t_{\text{global}}, R^{(i)}_{\text{local}}, X_1, X_2, T_{12}, w_{12})$
  $T_{12} = \text{knnsearch}(X_2, Z)$
return Z

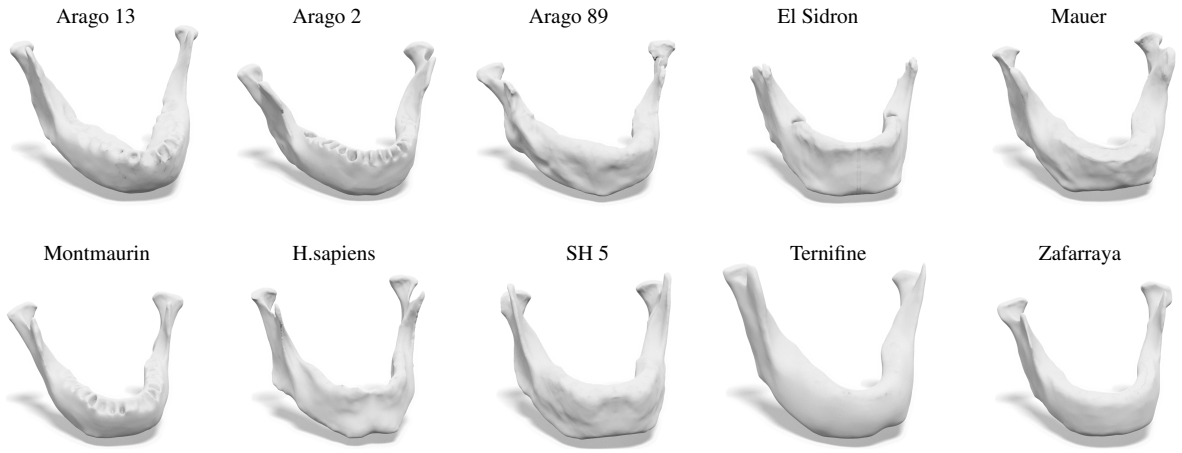

Figure 12: Remodeled meshes.

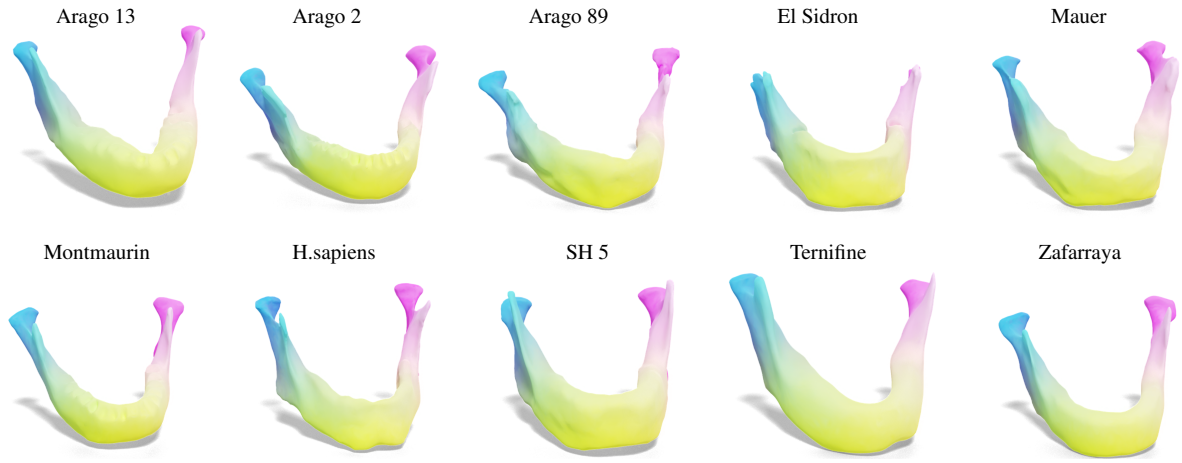

Figure 13: Computed point-wise correspondences on the remodeled dataset without any landmarks.

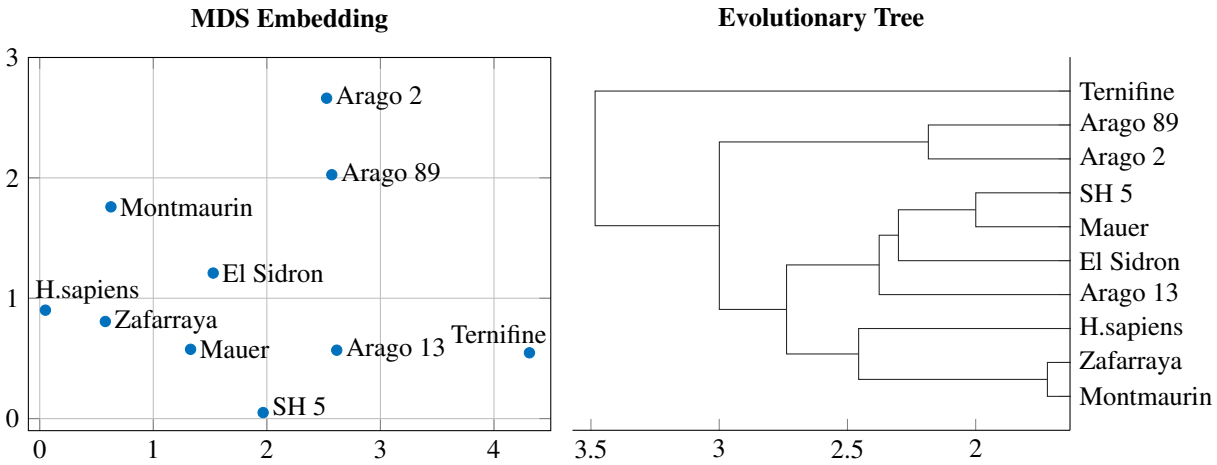

Figure 14: Global analysis of the Remodeled Dataset using Rigid Alignment Measure.

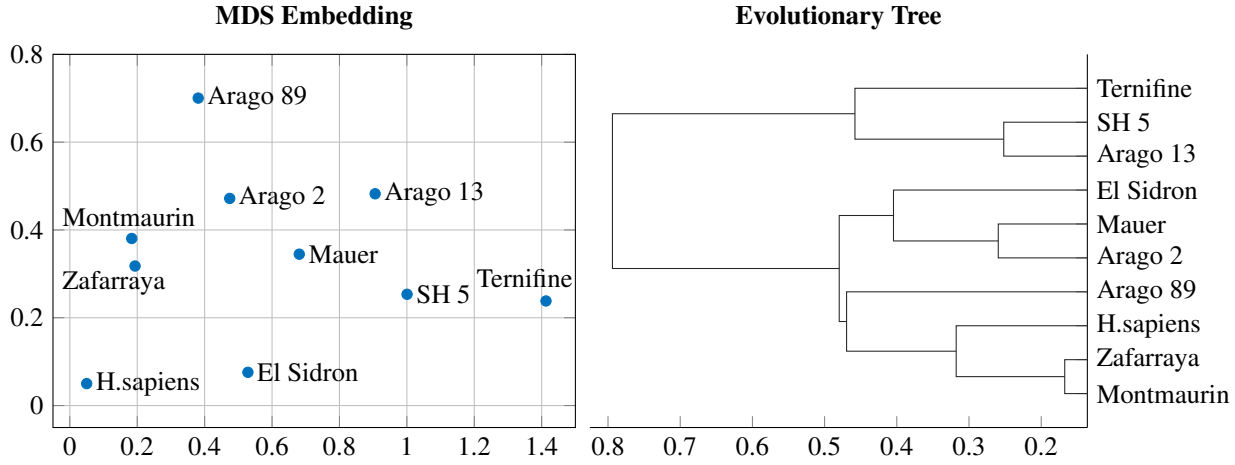

Figure 15: Global analysis of the Remodeled Dataset using Shape Distortion Measure.