# BaRe-ESA: A Riemannian Framework for Unregistered Human Body Shapes
# - Supplementary Material -

Emmanuel Hartman[1], Emery Pierson[2,4], Martin Bauer[1,3], Nicolas Charon[3], Mohamed Daoudi[4,5]
Florida State University, Tallahassee, Florida, USA [1],
University of Vienna, Vienna, Austria[2]
University of Houston, Houston, Texas, USA[3]
Univ. Lille, CNRS, Centrale Lille, Institut Mines-Télécom, UMR 9189 CRIStAL, Lille, F-59000, France[4],
IMT Nord Europe, Institut Mines-Télécom, Univ. Lille, Centre for Digital Systems, Lille, F-59000, France[5]

## 1. Formulas and implementations of mesh invariant similarity metrics

In the paragraphs below, we add a few details about the similarity metrics used in the registration procedure and for the evaluation and comparison of the different methods.

First, we remind that the Hausdorff distance between two shapes $[q_0]$ and $[q_1]$ is given by the formula:

$$d_H([q_0],[q_1]) = \max\left\{ \sup_{x_0\in[q_0]} \inf_{x_1\in[q_1]} \|x_0-x_1\|, \right. \\ \left. \sup_{x_1\in[q_1]} \inf_{x_0\in[q_0]} \|x_1-x_0\| \right\}$$

In our numerical experiments, we use the approximate implementation provided by libigl [8]. Note that this metric is typically very sensitive to outliers.

In contrast, the Chamfer distance [4, 6] provides a smoother version of the above and, given two point clouds $[q_0]$ and $[q_1]$, is defined as:

$$d([q_0],[q_1]) = \frac{1}{N_0} \sum_{x_0\in[q_0]} \inf_{x_1\in[q_1]} \|x_0-x_1\| \\ + \frac{1}{N_1} \sum_{x_1\in[q_1]} \inf_{x_0\in[q_0]} \|x_1-x_0\|.$$

We use the Pytorch implementation of Thibault Groueix[1]. One of the downsides of this metric when applied to discrete surfaces, however, is that it is not necessarily robust to local changes of mesh density (and thus not truly mesh invariant) since it is designed as a distance between point clouds.

To address that particular issue, as a final measure of reconstruction quality and fidelity metric for the latent code retrieval approach, we rely on the varifold distance introduced in [1, 9]. Specifically, assuming the discrete surface $[q_0]$ is given by the reunion of the triangles $\{T_i\}_{i=1,\dots,F}$ and $[q_1]$ as the reunion of triangles $\{T'_j\}_{j=1,\dots,F'}$, the discrete approximation of the squared varifold distance writes:

$$d^{\text{Var}}([q_0],[q_1])^2 = \sum_{i,j=1}^{F} k(x_i,n_i,x_j,n_j)a_i a_j \\ -2\sum_{i,j=1}^{F,F'} k(x_i,n_i,x'_j,n'_j)a_i a'_j + \sum_{i,j=1}^{F'} k(x'_i,n'_i,x'_j,n'_j)a'_i a'_j$$

where $x_i, n_i, a_i$ (resp. $x'_i, n'_i, a'_i$) denote the barycenter, unit normal vector and area of triangle $T_i$ (resp. $T'_i$). Here $k$ is a positive definite kernel function on $\mathbb{R}^3 \times S^2$. While several different families of kernels are possible (see discussion in [9]), in all the experiments of this paper, we specifically take $k(x,n,x',n') = e^{-\frac{|x-x'|^2}{\sigma^2}}(n\cdot n')^2$ where $\sigma$ can be interpreted as a spatial scale of sensitivity of the metric which is chosen to be quite small ($\sigma = 0.025$) in our examples. In this work, we adapted the Python implementation used in $H2\_SurfaceMatch$[2] which itself relies on the *PyKeops* library [5] for efficient evaluation and automatic differentiation of kernel functions on the GPU.

## 2. Second Order Sobolev Metrics

In Section 3 we outline the desired properties of a metric on the space of immersions that we pullback onto our latent space. Here we give a more in depth formulation for the family of split second order Sobolev metrics introduced in [7] that we use in our model. We begin with a second order

---

[1] `https://github.com/ThibaultGROUEIX/ChamferDistancePytorch`

[2] `https://github.com/emmanuel-hartman/H2_SurfaceMatch`

metric given by

$$\int_{\mathcal{T}} \langle h, h\rangle + g_q^{-1}(dh, dh) + \langle \Delta_q h, \Delta_q k\rangle \,\mathrm{vol}_q \qquad (1)$$

where we view $dh$ as a vector valued one-form, $g_q$ is the pullback of the Euclidean metric on $\mathbb{R}^3$ and $\Delta_q$ denotes the vector Laplace operator induced by the parametrization $q$. Fixing a coordinate view and treating $d_h$ and $g_q$ as $3 \times 2$ and $2 \times 2$ (resp.) matrix fields on $\mathcal{T}$, one can then express $\mathrm{vol}_q$ as $\sqrt{|g_q|}$ and the first order term is computed as $g_q^{-1}(dh, dh) = \mathrm{tr}(dh \cdot g_q^{-1})$. However, using the construction of [10], we may further decompose the vector valued one-forms by $dh = dh_m + dh_+ + dh_\perp + dh_0$. The exact formulas for these terms can be found in [10]. Each of these components are orthogonal with respect to the metric $g_q^{-1}$. The associated Riemannian energies of the first three can be roughly interpreted from the point of view of linear elasticity [2] as measuring the change of metric tensor under constant volume form (shearing), the change in volume density (stretching) and the change of normal vector direction (bending) respectively. The last term doesn't have such a clear interpretation but is necessary to recover the standard first-order Sobolev norm. Thus we can eventually decompose Equation (1) and introduce nonnegative weighting coefficients, producing the six parameter family of metrics given by

$$\begin{aligned} G_q(h,k) = \int_{\mathcal{T}} \Big( & a_0 \langle h, k\rangle + a_1 g_q^{-1}(dh_m, dk_m) \\ & + b_1 g_q^{-1}(dh_+, dk_+) + c_1 g_q^{-1}(dh_\perp, dk_\perp) \\ & + d_1 g_q^{-1}(dh_0, dk_0) + a_2 \langle \Delta_q h, \Delta_q k\rangle \Big) \,\mathrm{vol}_q \,. \end{aligned} \qquad (2)$$

The weighting of these terms have very natural geometric interpretations which can be adjusted to make the metric more suited for different applications. The zeroth-order term weighted by $a_0$ penalizes how far the surface is moved weighted by the volume form of the surface. The second-order term weighted by $a_2$ penalizes tangent vectors that increase the local curvature of the surface. The interpretation of the first order terms weighted by $a_1, b_1, c_1$ and $d_1$ was discussed above. In the application to human motions, we typically choose $a_1$ and $b_1$ to be the largest coefficients so that our metric penalizes non-isometric motions. In computations, the different terms in the metric are discretized on triangular meshes based on the principles of discrete differential geometry. A review of relevant methods from this field can be found e.g. in [3] and the details of the specific quantities we use are presented in [7].

## 3. Computational cost

As stated in the paper, our pipelines are optimization based. We provide a substantial comparison for the different approaches.

| Method | Training | Retrieval | Interpolation | |
|---|---|---|---|---|
| LIMP | 1.5w | <1s | <1s | |
| 3D-Coded | 12h | 160s | <1s | 160s |
| ARAPReg | 2w | 160s | <1s | 160s |
| BaRe-ESA | <1h | 160s | 91s | 160s |

Table 1: Computation costs for different methods. For the interpolation, the results are as follow: we display on the left the costs in the case latent codes are available, and the cost in the case they're not.

All the other approaches require significant training costs compared to BaRe-ESA which requires less than one hour, cf Table 1. On the other hand, BaRe-ESA, ARAPReg and 3d-Coded require additional optimization for the latent code retrieval, which we found takes approximately the same time for all three methods. The optimization cost is driven by the mesh invariant costs – varifold or Chamfer – which have $n^2$ complexity, where $n$ is the number of vertices. LIMP is the only method that doesn't require optimization, but the network behaves notably bad when the poses are unseen as showed in the experiments. For the interpolation problem our method requires approximately 90 seconds if the latent codes are already available, whereas it takes approximately the same time as one latent code retrieval if they are not available. All timing results were obtained using a standard home PC with a Intel 3.2 GHz CPU and a GeForce GTX 2070 1620 MHz GPU.

## 4. Description of state-of-the-art methods

We propose a detailed description of the state-of-the-art method we use as baselines. We selected deep learning methods that builds a flat latent space for human shape deformations. They describe as follows:

- Learning Latent Shape Representations with Metric Preservation (LIMP) is a deep learning method modeling deformations of shapes using a variational auto encoder with geodesic constraints. The encoder part use a PointNet architecture, which makes it invariant to parameterization. The decoder part is a Multi Layer Perceptron. The geometric constraints are used a loss functions during the training process. The latent space is divided in an extrinsic part and an intrinsic part and the loss is applied on the interpolation in those dimensions. The intrinsic part is constrained using the computation of full geodesic matrix, which make the training process particularly heavy.
- As-Rigid-As-Possible Regularization (LIMP) is a deep learning method modeling deformations of shapes using an auto-decoder architecture. The latent codes and

the decoder are learned altogether. During the training, an As-Rigid-As-Possible loss is imposed such that the decoder directions are similar to the ARAP ones. This procedure also makes the training procedure heavy. In order to make it parameterization invariant, we replace the $L^2$ metric by the varifold distance, as an alternative to our Riemannian latent space.

- 3D correspondences by deep deformation (3D Coded) is a deep learning method modeling deformations of shapes using a variational auto encoder. Similarly to LIMP, the encoder part use a PointNet architecture, which makes it invariant to parameterization. The decoder uses a Multi Layer Perceptron to deform a template mesh, but no constraint is imposed on the interpolation of latent variables. By taking advantage of a high number of training samples ($> 200000$), they obtained state-of-the-art results for human shape correspondence.

In the paper, all those methods are trained using the same training set as Bare-ESA, from Dynamic FAUST and reported parameters from the respective papers.

## 5. Comparison to the framework of [7]

In Figure 1 we compare BaRe-ESA to the unrestricted method of [7]. Note, that BaRE-ESA is significantly cheaper to compute as we reduced the dimension of the minimization problem – the latent space dimension will be in the order of 100s, while the dimension of the unrestricted method is on the order of 10000s. More importantly, one can observe that BaRe-ESA leads to significantly more natural deformations, cf. the movement of the arms in Fig. 1.

## 6. Algorithmic details

We provide below the pseudo-code of our algorithm for latent code retrieval of a scan.

## 7. Comparison against the linear model

We provide in this section a comparison between the full Bare-ESA model, and the simpler basis restricted linear model (no Riemannian metric). To demonstrate the pertinence of our model, we perform the comparison on the interpolation and extrapolation experiments. We display our results in Table 2, where we observe a significant gap between linear model and BaRe-ESA respective errors. As expected, the linear model generates non natural deformations, resulting in higher error compared to the real human motions.

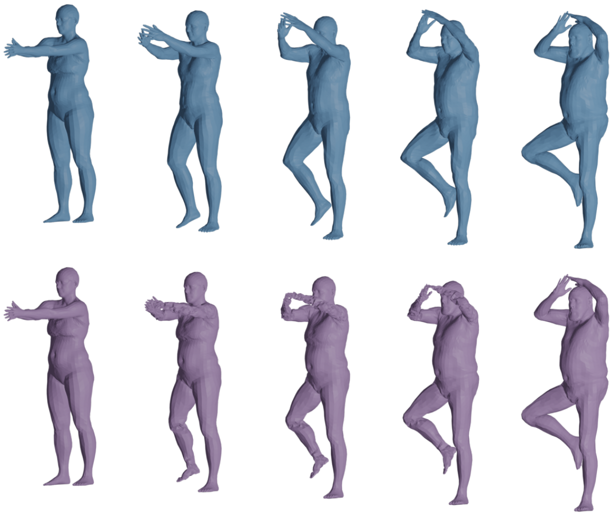

Figure 1: First line: optimal deformation calculated using the basis informed ESA of the present article. Second line: optimal deformation calculated using a standard $H^2$-matching.

**Algorithm 1:** Latent code retrieval of a scan

```
Input: The target scans $q_0$;
$a_0, a_1, b_1 c_1, d_1, a_2$ the parameter of the Sobolev elastic metric;
$(\lambda_k, \sigma_k)_{k=0}^{p}$ the balancing weight and the spatial support of the varifold distance at each refinement step
Output: $f_{\text{geo}}$: the geodesic connecting $q$ as the coefficients $\alpha$ and the representative $[q_0]$ in the template space.
Initialize $\alpha_{ij} = 0$ and the path as
  $\bar{q} + \sum_{i=1}^{m} \alpha_i^j h_i + \sum_{i=m+1}^{m+n} \alpha_i^j k_{i-m}\cdot$ ;
for $k \leftarrow 0$ to $p$ do
    Define the energy functional $E(\alpha) = \int_0^1 \overline{G}_\alpha(\partial_t \alpha, \partial_t \alpha) dt + \lambda_k \Gamma(F(\alpha)(1), q_0)$ in an automatic differentiation framework (PyTorch here), that computes the gradient value $\nabla_\alpha E$ along the functional value;
    Minimize $E$ with respect to $\alpha$ with a gradient descent algorithm (SciPy BFGS or L-BFGS-B), outputing optimal $\alpha_{out}$ coefficients based on initialization $\alpha$;
    Set $\alpha = \alpha_{out}$;
end
Set $[q_0]$ to be the endpoint of the final geodesic;
return $\alpha$ and $[q_0]$
```

|  | Linear Model |  |  | Bare-ESA |  |  |
|---|---|---|---|---|---|---|
|  | Hausdorff | Chamfer | Varifold | Hausdorff | Chamfer | Varifold |
| Interpolation | 0.18 | 0.07 | 0.03 | **0.07** | **0.04** | **0.02** |
| Extrapolation | 0.39 | 0.44 | 0.05 | **0.16** | **0.10** | **0.02** |

Table 2: Mean errors of Linear Model and Bare-ESA for interpolation and extrapolation experiments. The Hausdorff, Chamfer and varifold distance are computed against ground truth sequences.

## 8. Precise requirements of BaRe-ESA

In our experiments we used registered 4D data to construct the body pose basis by performing PCA on the tangent vectors of curves in the space of human body surfaces that represent natural human motions. To construct the body pose basis we compute geodesics between humans in a similar pose and apply the same PCA approach to the tangent vectors of the resulting curves in the space of human body surfaces. We do not consider the use of 4D data for constructing a pose basis as a limitation of our method, but rather an advantage, since it allows us to utilize information from actual human motions to inform the construction of the pose change basis. By contrast, most alternative methods do not utilize the 4D data in this way. Moreover, we can use the framework of [7] to construct geodesics as synthetic training data in applications where 4D data is not available. In the same time, the assumption on the existence of humans in a similar body pose could be dropped relatively easily, as one could use the previously obtained body pose basis to create such a training set if needed. This would significantly change the training effort required by our approach.

## References

[1] Nicolas Charon and Alain Trouvé. The varifold representation of nonoriented shapes for diffeomorphic registration. *SIAM journal on Imaging Sciences*, 6(4):2547–2580, 2013. 1

[2] Nicolas Charon and Laurent Younes. Shape spaces: From geometry to biological plausibility. *Handbook of Mathematical Models and Algorithms in Computer Vision and Imaging: Mathematical Imaging and Vision*, pages 1929–1958, 2023. 2

[3] Keenan Crane. Discrete differential geometry: An applied introduction. *Notices of the AMS, Communication*, pages 1153–1159, 2018. 2

[4] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 1

[5] Jean Feydy, Joan Glaunès, Benjamin Charlier, and Michael Bronstein. Fast geometric learning with symbolic matrices. *Advances in Neural Information Processing Systems*, 33, 2020. 1

[6] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 230–246, 2018. 1

[7] Emmanuel Hartman, Yashil Sukurdeep, Eric Klassen, Nicolas Charon, and Martin Bauer. Elastic shape analysis of surfaces with second-order sobolev metrics: a comprehensive numerical framework. *arXiv preprint arXiv:2204.04238*, 2022. 1, 2, 3, 4

[8] Alec Jacobson, Daniele Panozzo, et al. libigl: A simple C++ geometry processing library, 2018. https://libigl.github.io/. 1

[9] Irène Kaltenmark, Benjamin Charlier, and Nicolas Charon. A general framework for curve and surface comparison and registration with oriented varifolds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3346–3355, 2017. 1

[10] Zhe Su, Martin Bauer, Eric Klassen, and Kyle Gallivan. Simplifying transformations for a family of elastic metrics on the space of surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 848–849, 2020. 2