ARC-Flow : Articulated, Resolution-Agnostic, Correspondence-Free Matching and Interpolation of 3D Shapes Under Flow Fields

Adam Hartshorne Allen Paul Tony Shardlow Neill D. F. Campbell
University of Bath
ath35, ap2746, tjs42, nc537 @bath.ac.uk

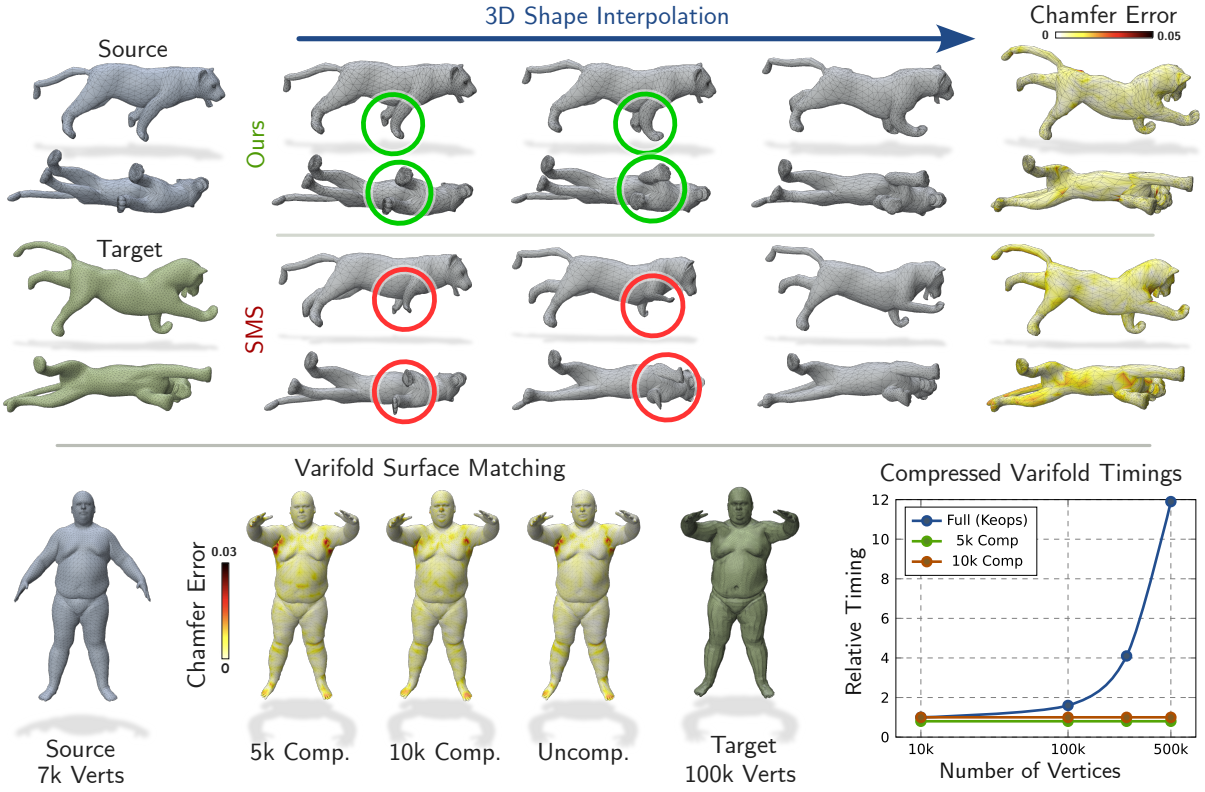

Figure 1. Top: 3D shape interpolation. *Our method obtains a more reliable interpolation than existing state-of-the-art approaches (e.g. Spectral Meets Spatial [12]) even under substantial non-isometric deformation. Samples taken at* $t = 0.25, 0.5, 0.75$ *and* $1.0 =: T$ *along the deformation path.* ***Bottom: Scaling to high-resolution meshes.*** *Using Varifold compression we obtain dramatic computational savings, while maintaining similar perceptual quality, allowing us to scale to high resolution meshes.*

Abstract

This work presents a unified framework for the unsupervised prediction of physically plausible interpolations between two 3D articulated shapes and the automatic estimation of dense correspondence between them. Interpolation is modelled as a diffeomorphic transformation using a smooth, time-varying flow field governed by Neural Ordinary Differential Equations (ODEs). This ensures topological consistency and non-intersecting trajectories while accommodating hard constraints, such as volume preservation, and soft constraints, e.g. physical priors.

Correspondence is recovered using an efficient Varifold formulation, that is effective on high-fidelity surfaces with differing parameterisations. By providing a simple skeleton for the source shape only, we impose physically motivated constraints on the deformation field and resolve symmetric ambiguities. This is achieved without relying on skinning weights or any prior knowledge of the skeleton's target pose configuration. Qualitative and quantitative results demonstrate competitive or superior performance over existing state-of-the-art approaches in both shape correspondence and interpolation tasks across standard datasets.

Acknowledgements: We are grateful for support from the EPSRC SAMBa Centre for Doctoral Training in Statistical Applied Mathematics (EP/L015684/1), the EPSRC CAMERA Research Centre (EP/M023281/1 and EP/T022523/1), UKRI Strength in Places Fund MyWorld Project (SIPF00006/1) and the Royal Society.

1. Introduction

Recovering maps between 3D shapes is a fundamental problem in computer vision and graphics since it facilitates shape analysis and generation. In particular, modelling realistic non-rigid deformations of 3D shapes is a recognisable problem at the heart of numerous tasks, including animation, shape comparison and style transfer. Although shape matching and interpolation are closely related, they are often treated separately or sequentially, e.g. in learning statistical shape models.

Aligning a common template to a set of shapes, such as 3D human point clouds in different poses, is also a long-standing problem crucial for geometric learning. Despite extensive research, establishing accurate correspondences between non-rigidly deformed shapes remains challenging in real-world scenarios due to non-isometric deformations, variations in discretisation, and symmetries, e.g. those in bipeds and quadrupeds. Furthermore, there is often a desire to also transfer a skeletal structure from a template to different sample poses [48] for applications such as rigging.

While articulated objects undergo shape changes due to articulation, these changes are not arbitrary. Interpolation involves finding a continuous deformation path from a source shape to a target shape, ensuring that the trajectory represents a realistic articulation of the underlying physical object. Moreover, the shape must remain topologically valid and self-intersection-free throughout the trajectory.

Contributions: This work encompasses the following:

- A novel approach for the simultaneous recovery of physically plausible trajectories and automatic inference of shape correspondence.
- Shape transformation using smooth diffeomorphic fields, ensuring topological consistency and non-intersecting trajectories while accommodating constraints (e.g. volume preservation) and physically inspired priors.
- In exchange for providing a simple skeletal structure to augment the source mesh, our method can model rigid (bone) and conformal (soft tissue) deformations within the enclosed volume. These physically inspired soft constraints enhance interpolation quality and help mitigate symmetric ambiguity.
- Shapes are matched using a Varifold metric, a technique from geometric measure theory, that enables comparison independent of shape fidelity (e.g. mesh resolution) or parameterisation. A recently proposed compression technique enhances computational and memory efficiency, enabling effective scaling to densely sampled shapes.

2. Related Work

This section reviews the most relevant methods to the proposed approach, with Table 1 summarising the key attributes of leading techniques.

Method	Unsup.	No DS	No PM	Scalable
Div-Free [19]	✗	✓	–	✓
Ham. Dyn. [18]	✗	✓	–	✓
NeuroMorph [21]	✓	✗	✗	✗
SMS [12]	✓	✗	✗	✗
ESA [28]	✓	✓	✓	✗
Ours	✓	✓	✓	✓

Table 1. Comparison with leading methods. *Operating in an unsupervised manner (**Unsup.**), doesn't requiring a dataset of samples for training (**No DS**), avoids using a permutation matrix (**No PM**), and scales to high-resolution surfaces (**Scalable**).*

Shape Matching: Shape registration [67, 72], a fundamental problem in computer vision, aims to determine spatial correspondences between pairs of shapes. Use cases range from pose estimation for noisy point clouds to the non-parametric registration of high-resolution medical images.

The Functional Mapping framework, proposed by Ovsjanikov *et al*. [50], efficiently learns mappings between isometrically deformed shapes using the eigenfunctions of the Laplace-Beltrami operator. Dense point-wise correspondences are then recovered via a nearest neighbour search in functional space using Iterative Closest Point (ICP). Due to its simplicity, generalisability and efficiency this framework, has been widely extended to handle non-isometric mappings [20, 46, 55, 57], improve matching accuracy [22, 56], and tackle partial [43, 58] and multi-shape [16, 24, 33] matching. While deep learning-based approaches [26, 42, 60] have also been proposed to remove reliance on hand-crafted feature descriptors.

However, two-stage "match and refine" approaches often yield sub-optimal dense correspondences. Post-processing techniques [20, 47, 74] attempt to improve the final point-wise maps, while Cao *et al*. [11] directly enforce the functional map to be associated with a point-wise map and optimise both simultaneously. Nevertheless, mapping in the spectral domain does not guarantee smooth point-wise correspondences, and solving for a permutation matrix makes it difficult to handle shapes with differing resolutions.

Embedding shapes into measure spaces provides a way to address these issues. The understanding of surface geometry in the context of geometric measure theory and calculus of variations has been widely studied in mathematics, leading to the development of parametrisation-invariant shape matching metrics e.g. Currents [70], Varifolds [9, 10, 14, 35] and Normal Cycles [61]. In particular, the Varifold representation has previously been used in the context of human shapes [3, 27, 54].

Despite invariance to parameterisation and robustness to noise, they scale poorly due to the use of dense kernels, limiting their adoption (quadratic scaling). The Keops library [13, 23] has been the de-facto approach to address these

issues, using Map-Reduce to minimise memory allocation by decomposing large matrices into sub-problems. This enables parallelised GPU computations via specialised CUDA kernels, but this does not eliminate the calculation of all pairwise distances. In contrast to previous computationally prohibitive approaches to measure-based shape compression [17, 25, 31], Paul et al. [52] have recently developed a practical solution utilising Ridge Leverage Score (RLS) sampling, which produces an accurate lower-dimensional representation with substantially improved computational efficiency. Our work leverages this method to enable the scaling to high-resolution target shapes.

Shape Interpolation: Non-rigid interpolation techniques have been extensively explored to generate physically plausible morphing paths between the source and target with applications in shape exploration and deformation transfer [40, 66]. Target alignment is typically balanced against adherence to quality metrics e.g. distortion minimisation and preservation of local geometric features.

Common approaches include posing interpolating trajectories as geodesics in higher-dimensional shape spaces [8, 29, 30, 75], employing local distortion deformation measures like as-rigid-as-possible (ARAP) [65] or PriMo [6]. Alternative methods include cage-based deformations [34], continuum mechanics-based approaches [29, 30], and interpolating intrinsic quantities before reconstructing the 3D shape [2]. However, direct vertex offset predictions can lead to geometric artefacts e.g. self-intersection, while coarse control cages may overly restrict the deformation process.

Eisenberger *et al*. [18, 19] formulate the problem as a time-dependent gradient flow, combining divergence-free vector fields for volume preservation with anisotropic ARAP constraints; computational complexity is controlled by learning fields from a data subsample, e.g. a few thousand points.

Despite these innovations, many approaches rely on consistent surface parameterisation between source and target, known correspondences a-priori, and computing expensive constraints at each interpolation step.

Combined Approaches: NeuroMorph [21] produces continuous interpolations between meshes while establishing point-to-point correspondence, using two unsupervised neural networks to learn the displacement field and permutation matrix. Cao *et al*. [12] improved upon this by harmonising spectral and spatial maps, leading to more accurate and smoother point-wise correspondences under large non-isometric deformations. However, these deep learning-based methods are constrained by their reliance on large, diverse shape datasets for time-intensive feature mapping and their use of permutation matrices, which impede scalability and practical applications. Furthermore, modelling deformations as an offset vector applied directly to the vertices doesn't guarantee smooth trajectories, the non-intersection of surfaces or volume preservation by construction.

Other recent advances have drawn inspiration from geometric measure theory and the Large Deformation Diffeomorphic Metric Mapping (LDDMM) model. Bauer *et al*. [3] proposed a framework for surface registration using the Square Root Normal Field (SRNF) pseudo-distance. Building on this, Hartman *et al*. [28] employed Varifolds to interpolate between unregistered triangulated surfaces. However, this latter approach not only inherits the scaling limitations associated with Varifolds but also relies on learning complex geodesics defined by intricate partial differential equations (PDEs) to predict the interpolation.

3. Varifolds in a Nutshell

Varifold matching [14] is a technique often used for surface matching in Computational Anatomy [78] and has its origin in geometric measure theory [1]. The goal is to establish a metric for the distance between two shapes without requiring correspondence. We seek to provide a high-level intuition and details of the resulting computation but an in-depth treatment is left to further literature [52].

We consider a vector field $\vec{\mathbf{v}}(\mathbf{x}) : \mathbb{R}^3 \to \mathbb{R}^3$ and a surface $\boldsymbol{\mathcal{X}}$, and define a scalar measure as the integral of the vector field over the surface, often termed the current, as

$$\mu_{\boldsymbol{\mathcal{X}}}(\vec{\mathbf{v}}) := \int_{\boldsymbol{\mathcal{X}}} \vec{\mathbf{v}}(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x})\, dS_{\boldsymbol{\mathcal{X}}}(\mathbf{x})\,, \tag{1}$$

where we have taken the inner product between the vector field and the unit surface normal $\hat{\mathbf{n}}(\mathbf{x})$ with $dS_{\boldsymbol{\mathcal{X}}}(\mathbf{x})$ as the elemental surface area. We limit the vector fields to be a smooth space $\mathbb{V}$ defined as a vector Reproducing Kernel Hilbert Space (RKHS) with kernel $\kappa(\mathbf{x}, \mathbf{x}')$ that defines the spatial correlation across the vector field. This allows us to consider the "best" vector field that gives the highest measure (under the dual RKHS norm) as

$$\|\mu_{\boldsymbol{\mathcal{X}}}\|_{\mathbb{V}^*} := \sup_{\vec{\mathbf{v}} \in \mathbb{V}, \|\vec{\mathbf{v}}\| \leq 1} \left| \mu_{\boldsymbol{\mathcal{X}}}(\vec{\mathbf{v}}) \right| ; \tag{2}$$

intuitively this is the smoothest field that passes through the surface with the vectors most aligned with the surface normals. The reproducing property of the RKHS provides a closed form solution to Eq. (2) through the inner product $\|\mu_{\boldsymbol{\mathcal{X}}}\|^2_{\mathbb{V}^*} = \langle \mu_{\boldsymbol{\mathcal{X}}}, \mu_{\boldsymbol{\mathcal{X}}} \rangle_{\mathbb{V}^*}$ where the product $\langle \mu_{\boldsymbol{\mathcal{X}}}, \mu_{\boldsymbol{\mathcal{Y}}} \rangle_{\mathbb{V}^*}$ is

$$\int_{\boldsymbol{\mathcal{X}}} \int_{\boldsymbol{\mathcal{Y}}} \kappa(\mathbf{x}, \mathbf{y}) \langle \hat{\mathbf{n}}_{\boldsymbol{\mathcal{X}}}(\mathbf{x}), \hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}}(\mathbf{y}) \rangle\, dS_{\boldsymbol{\mathcal{X}}}(\mathbf{x})\, dS_{\boldsymbol{\mathcal{Y}}}(\mathbf{y}). \tag{3}$$

We can now define a distance between two shapes using norm as $d(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}) := \|\mu_{\boldsymbol{\mathcal{X}}} - \mu_{\boldsymbol{\mathcal{Y}}}\|^2_{\mathbb{V}^*}$ where

$$\begin{aligned} d(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}) &= \langle \mu_{\boldsymbol{\mathcal{X}}} - \mu_{\boldsymbol{\mathcal{Y}}}, \mu_{\boldsymbol{\mathcal{X}}} - \mu_{\boldsymbol{\mathcal{Y}}} \rangle_{\mathbb{V}^*} \\ &= \langle \mu_{\boldsymbol{\mathcal{X}}}, \mu_{\boldsymbol{\mathcal{X}}} \rangle_{\mathbb{V}^*} - 2\langle \mu_{\boldsymbol{\mathcal{X}}}, \mu_{\boldsymbol{\mathcal{Y}}} \rangle_{\mathbb{V}^*} + \langle \mu_{\boldsymbol{\mathcal{Y}}}, \mu_{\boldsymbol{\mathcal{Y}}} \rangle_{\mathbb{V}^*}\,. \end{aligned} \tag{4}$$

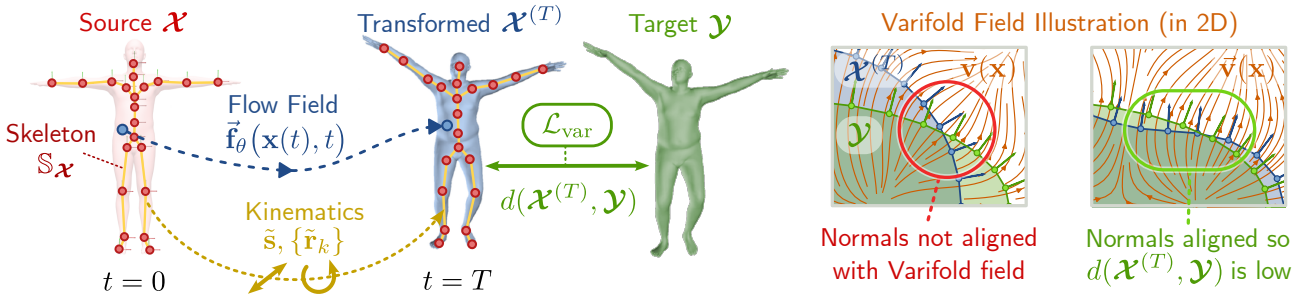

***Figure 2. Left: Method overview**. We transform the source shape to match the target using a diffeomorphic differential vector field; we also recover the forward kinematic transformation of the source skeleton. Matching is performed with a correspondence-free Varifold metric. **Right: Varifold field.** Visualisation of an implicit Varifold field for intuition; computation of $\vec{\mathbf{v}}(\mathbf{x})$ is not required for matching.*

The final step to the full Varifold measure is to generalise and include a kernel over the normal vectors as well, rather than the standard Euclidean dot product in Eq. (3), so that we have a spatial kernel $\kappa_{\mathbf{x}}(\mathbf{x}, \mathbf{x}')$ and a normal kernel $\langle \hat{\mathbf{n}}_{\mathcal{X}}, \hat{\mathbf{n}}_{\mathcal{Y}} \rangle \to \kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\mathcal{X}}, \hat{\mathbf{n}}_{\mathcal{Y}})$; $\langle \mu_{\mathcal{X}}, \mu_{\mathcal{Y}} \rangle$ becomes

$$\int_{\mathcal{X}} \int_{\mathcal{Y}} \kappa_{\mathbf{x}}(\mathbf{x}, \mathbf{y})\, \kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\mathcal{X}}(\mathbf{x}), \hat{\mathbf{n}}_{\mathcal{Y}}(\mathbf{y}))\, dS_{\mathcal{X}}(\mathbf{x})\, dS_{\mathcal{Y}}(\mathbf{y}) \,. \tag{5}$$

Intuition: If we specify Gaussian kernels, that is $\kappa(\mathbf{x}, \mathbf{y}) := \exp(-\frac{1}{2\ell^2}\|\mathbf{x} - \mathbf{y}\|^2)$, then intuition for the metric becomes clear. The product in Eq. (5) makes sense as when points $\mathbf{x}$ and $\mathbf{y}$ from the two shapes are close, i.e. $\|\mathbf{x} - \mathbf{y}\|$ is small, then $\kappa_{\mathbf{x}}(\mathbf{x}, \mathbf{y})$ will be large and we want $\kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\mathcal{X}}(\mathbf{x}), \hat{\mathbf{n}}_{\mathcal{Y}}(\mathbf{y}))$ to be large as well; therefore, $\|\hat{\mathbf{n}}_{\mathcal{X}}(\mathbf{x}) - \hat{\mathbf{n}}_{\mathcal{Y}}(\mathbf{y})\|$ becomes small and the normals will be equal. Conversely, when points from the different shapes are far apart then $\|\mathbf{x} - \mathbf{y}\|$ is large and $\kappa_{\mathbf{x}}(\mathbf{x}, \mathbf{y}) \approx 0$ so the normals can be different; illustrated on the right of Fig. 2. We can pick the lengthscales $\ell_{\mathbf{x}}$ and $\ell_{\mathbf{n}}$ appropriately for the resolution we want the surfaces to match at (they should be commensurate with the surface discretisation). The measure is minimised (at 0) when the two surfaces are the same and, importantly, *at no point have we required known correspondences between the two shapes*.

4. Method

Notation: Shape interpolation continuously deforms a source shape $\mathcal{X}$ to match a target shape $\mathcal{Y}$. Formally, we let $\mathcal{X} = \{V_{\mathcal{X}}, N_{\mathcal{X}}, dS_{\mathcal{X}}\}$ consist of a discrete set of vertices on the surface, $V_{\mathcal{X}} = \{\mathbf{x}_i\}, i \in [1, I], \mathbf{x}_i \in \mathbb{R}^3$, with associated normal vectors $F_{\mathcal{X}} = \{\hat{\mathbf{n}}_i\}$ and surface areas $dS_{\mathcal{X}} = \{s_i\}$; similarly for $\mathcal{Y}$. These could be from a triangular mesh but our approach allows for more general surface representations. In addition, *for the source shape alone*, we provide a simple internal skeleton $\mathbb{S}_{\mathcal{X}} = \{\mathbf{b}_j, e_k\}$ comprising an acyclic graph of internal vertices $\mathbf{b}_j \in \mathbb{R}^3$, $j \in [1, J]$, connected by edges $e_k = (j, j')$, $k \in [1, K]$.

Overview: Our method comprises four key components:

1. Deformation via a diffeomorphic flow field to guarantee **preservation of topology**; the vector field is constructed to be divergence free and therefore **preserves volume**.
2. A varifold metric is used to ensure the **deformed surface matches the target without known correspondence**.
3. We supply the source with a simple, articulated internal skeleton and then force our field to **infer the new skeletal pose automatically whilst promoting local rigidity**.
4. The surface and soft tissue surrounding the skeleton are encouraged to deform in a conformal manner by **incorporating physically inspired priors**.

This section discusses the components of our model, illustrated in Fig. 2, used to learn the deformation.

4.1. Diffeomorphic Flow

The deformation is modelled as a diffeomorphism, represented by a time-varying vector flow field $\vec{\mathbf{f}}(\mathbf{x}, t) : \mathbb{R}^{3\times 1} \to \mathbb{R}^3$, that deforms the surface under an Ordinary Differential Equation (ODE); the dynamics are provided by a neural network as a NeuralODE [15]. The surface is pushed (or "evolved") under this flow at all points in a continuous manner, independent of mesh resolution or parameterisation. As the vector field is continuous and differentiable, the "streamlines" of the field will not cross; the surface is guaranteed not to self-intersect and the topology (e.g. number of holes) is preserved. Equally, differential properties of the surface or volume (e.g. the normal vector to the surface) can also be propagated through the field.

Deformation: A surface position becomes a function of time, $\mathbf{x}(t)$, where we start at $t = 0$ on the source shape, $\mathbf{x}(0) \in V_{\mathcal{X}}$, and at some fixed time $t = T$ we want the point to lie on the surface of the target $\mathcal{Y}$. Therefore the shape evolves under an initial value ODE

$$\frac{d}{dt}\mathbf{x}(t) = \vec{\mathbf{f}}(\mathbf{x}(t), t), \quad \text{s.t.} \quad \mathbf{x}(t = 0) \in V_{\mathcal{X}} \,. \tag{6}$$

The estimate of a point $\mathbf{x}^{(0)} := \mathbf{x}(t = 0)$ on the source moving to the point $\mathbf{x}^{(T)} := \mathbf{x}(t = T)$ on the target surface is the solution to the Initial Value Problem (IVP)

$$\mathbf{x}^{(T)} = \mathbf{x}^{(0)} + \int_0^T \vec{\mathbf{f}}_\theta(\mathbf{x}(t), t)\, dt =: \mathsf{ODE}_{\mathsf{Solve}}\left(\mathbf{f}_\theta, \mathbf{x}_0, T\right) \,, \tag{7}$$

where we have parameterised the flow field by θ, the neural network weights of a NeuralODE, and defined the $\mathsf{ODE}_{\mathsf{Solve}}$ function as the result of a numerical solver that approximates the integral to a given tolerance.

Differential Properties: If we parameterise our NeuralODE appropriately we can guarantee suitable continuity on the vector field. This allows us to transfer differential properties from source to target. For example, if $\mathcal{J}[\cdot]$ is some linear differential operator that yields the surface normal, i.e. $\mathcal{J}[\mathbf{x}] = \mathbf{n}$, we have

$$\mathbf{n}^{(T)} = \mathbf{n}^{(0)} + \int_0^T \mathcal{J}\big[\vec{\mathbf{f}}_\theta(\mathbf{x}(t), t)\big]\, dt\,. \tag{8}$$

Volume Preservation: Many natural shapes, e.g. humans and animals, are composed mostly of incompressible tissues that preserve volume under deformation. We incorporate the constraint by learning a divergence-free vector field [19]. The curl of a vector field is always divergence-free, thus we parameterise the vector flow field as

$$\vec{\mathbf{f}}_\theta(\mathbf{x}, t) := \nabla \times \vec{\mathbf{a}}_\theta(\mathbf{x}, t)\,, \tag{9}$$

where $\vec{\mathbf{a}}_\theta(\mathbf{x}, t) \in \mathbb{R}^3$ is the output of a neural network and the curl operation is calculated by auto-differentiation.

ARC-Net: A SIREN [64] network, with a layer of FINER [44], is used as to specify the vector field $\vec{\mathbf{a}}_\theta(\mathbf{x}, t)$ in our NeuralODE. This enables the network to represent fine details in the underlying signal and capture the derivatives of the target function; this addresses the limitations of conventional ReLU-based MLPs. See the supplementary material for details of the precise architecture used.

ODE Solver: Our approach is solver-agnostic; however, in recent years, probabilistic solvers have emerged as an efficient framework for integrating uncertainty quantification with inference in dynamical systems [5, 37, 38, 62, 68, 69]. These solvers utilise a Gaussian Process (GP) function, which offers greater flexibility compared to traditional Runge-Kutta (RK) based polynomial representations, thereby reducing the number of time steps required to solve the dynamics of an underlying ODE function accurately. We solve the IVP on a fixed time grid using the "KroneckerEK0" formulation with a single derivative [37].

4.2. Varifold Matching

We use a discrete approximation for the integral in Eq. (5),

$$\langle \mu_{\boldsymbol{\mathcal{X}}}, \mu_{\boldsymbol{\mathcal{Y}}} \rangle \approx \sum_{i=1}^{I_{\boldsymbol{\mathcal{X}}}} \sum_{i'=1}^{I_{\boldsymbol{\mathcal{Y}}}} \kappa_{\mathbf{x}}(\mathbf{x}_i, \mathbf{y}_{i'}) \kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\boldsymbol{\mathcal{X}}_i}, \hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}_{i'}}) s_{\boldsymbol{\mathcal{X}}_i}\, s_{\boldsymbol{\mathcal{Y}}_{i'}}\,, \tag{10}$$

to calculate the Varifold metric, Eq. (4), to use as the surface matching loss $\mathcal{L}_{\text{var}}(\theta) := d(\boldsymbol{\mathcal{X}}^{(T)}, \boldsymbol{\mathcal{Y}})$. Here, $\boldsymbol{\mathcal{X}}^{(T)}$ denotes the set of vertices, normals and differential surface areas that are obtained at time $t = T$ from pushing $\boldsymbol{\mathcal{X}}$ through the $\mathsf{ODE}_{\mathsf{Solve}}$ in Sec. 4.1.

Efficient Scaling: The product computation in Eq. (10) is $\mathcal{O}(I_{\boldsymbol{\mathcal{X}}} I_{\boldsymbol{\mathcal{Y}}})$, i.e. quadratic in the mesh resolution; this substantial computational burden has previously limited the use of Varifolds. We build on recent work from Paul *et al*. [52] that dramatically reduces the computational cost and allows scaling to high resolution meshes. The method compresses each densely sampled surface, $\boldsymbol{\mathcal{S}}$, into a lower-dimensional representation, $\boldsymbol{\mathcal{S}}_{\boldsymbol{\mathcal{C}}}$, via Ridge Leverage Score (RLS) Sampling, to yield an accurate approximation of the Varifold representation; high-resolution shapes can be compressed within seconds while maintaining perceptually lossless quality. Post-compression, the same Varifold loss metric, Eq. (4), is computed using the new sample locations, normals, and weights, $\{V_{\boldsymbol{\mathcal{S}}_{\boldsymbol{\mathcal{C}}}}, N_{\boldsymbol{\mathcal{S}}_{\boldsymbol{\mathcal{C}}}}, dS_{\boldsymbol{\mathcal{S}}_{\boldsymbol{\mathcal{C}}}}\}$, resulting in far smaller kernels and much faster calculation (see Fig. 1). See the supplementary material for algorithmic details.

4.3. Skeleton-Driven Transformation

We assume an articulated body has an internal skeletal structure and provide a simple skeleton $\mathbb{S}_{\boldsymbol{\mathcal{X}}} = \{\mathbf{b}_j, e_k\}$, of vertices and edges, for the source shape $\boldsymbol{\mathcal{X}}$. This should deform in a locally rigid manner whilst the surrounding soft tissue and surface deform non-rigidly; we promote this behaviour with an appropriate penalty (loss) on the deformation field, however, we do not know the skeletal pose of the target. To that end, we simultaneously (i) solve for the forward kinematics of the final skeletal pose (the global translation $\tilde{\mathbf{s}} \in \mathbb{R}^3$ and quaternion joint angles $\tilde{\mathbf{r}}_k \in \mathbb{Q}$ between each bone), and (ii) penalise the field for any non-rigid deformation across the skeleton. We apply this penalty throughout the interpolation, i.e. over $t \in (0, T]$, to ensure the skeleton always remains piecewise rigid.

Solving the forward kinematic chain for the skeleton yields the absolute translation and quaternion for each bone

$$\{\mathbf{s}_k^{(T)}, \mathbf{r}_k^{(T)}\} = \mathsf{FwdKinematics}(\mathbb{S}_{\boldsymbol{\mathcal{X}}}, \tilde{\mathbf{s}}, \{\tilde{\mathbf{r}}_k\})\,, \tag{11}$$

where the superscript denotes the final pose (at $t = T$).

Bone Model: We model each bone $e_k = (j, j')$ as a cylinder from $\mathbf{b}_j$ to $\mathbf{b}_{j'}$ with a radius as a proportion, β, of the length. We sample a random point $\mathbf{p}_k^{(0)}(\boldsymbol{\alpha})$ in this cylinder where $\boldsymbol{\alpha} := [\alpha_\tau, \alpha_\rho, \alpha_\psi] \in [0, 1]^2 \times [0, 2\pi]$ are the cylindrical coordinates; α_τ is scaled to cover the length $0 \to \|\mathbf{b}_{j'} - \mathbf{b}_j\|$ and α_ρ the radius $0 \to \beta\,\|\mathbf{b}_{j'} - \mathbf{b}_j\|$. We interpolate the rigid transformation for the point, $\mathbf{p}_k^{(0)}(\boldsymbol{\alpha})$,

$$\mathbf{p}_k(\boldsymbol{\alpha}, t) = \mathbf{s}_k(t) + \mathbf{r}_k(t) \cdot \mathbf{p}_k^{(0)}(\boldsymbol{\alpha}) \cdot \mathbf{r}_k^*(t) \tag{12}$$

using the SLERP [63] transformation with $\mathbf{s}_k(t) := \frac{t}{T}\mathbf{s}_k^{(T)}$ and $\mathbf{r}_k(t) := \exp\left(\frac{t}{T}\ln(\mathbf{r}_k^{(T)})\right)$. Illustrated in Fig. 3, we compare points in the bones pushed through the ODE solver to their corresponding rigid body estimate, over a set of

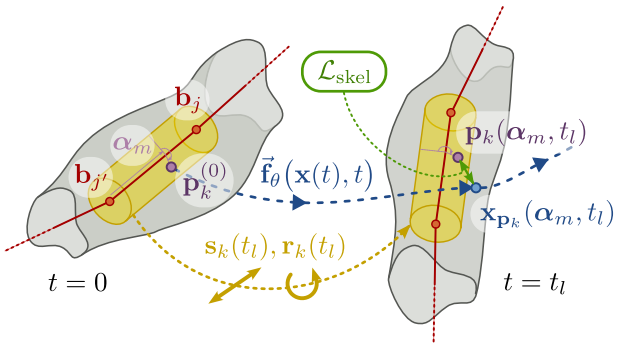

***Figure 3. Skeleton loss**. We encourage the flow to match an estimated rigid transform for points within the cylindrical "bones".*

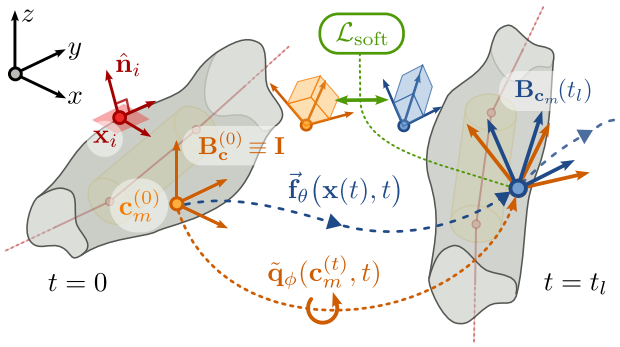

***Figure 4. Soft tissue loss**. We penalise arbitrary deformation of the soft tissue (and surface) using physically inspired priors to minimise shear/strain energy throughout the flow.*

times $\{t_l\} \in (0, T]$ and samples $\{\boldsymbol{\alpha}_m\}$, to provide the loss

$$\mathcal{L}_{\text{skel}}(\theta, \tilde{\mathbf{s}}, \{\tilde{\mathbf{r}}_k\}) := \sum_{k,l,m} \left\| \mathbf{p}_k(\boldsymbol{\alpha}_m, t_l) - \mathbf{x}_{\mathbf{p}_k}(\boldsymbol{\alpha}_m, t_l) \right\|^2 ,$$

$$\mathbf{x}_{\mathbf{p}_k}(\boldsymbol{\alpha}_m, t_l) := \mathsf{ODE}_{\mathsf{Solve}}(\mathbf{f}_\theta, \mathbf{p}_k^{(0)}(\boldsymbol{\alpha}_m), t_l) \, . \tag{13}$$

4.4. Soft Tissue Transformations

The surface and soft tissues will deform non-rigidly around the bones (under the constraint of volume preservation); however, they are not a free-flowing fluid and will resist non-rigid deformation with shear or strain energies. We model this as a physically inspired penalty that resists arbitrary transformations; this is related to existing physical based priors such as "as-rigid-as-possible" [65] or "as-conformal-as-possible" [77].

Inspired by continuum mechanics [73], we consider how the local basis vectors around a point (i.e. an elemental cube) deform. We sample a point $\mathbf{c}^{(0)}$ in the soft tissue and consider three unit differential axis-aligned basis vectors $\hat{\boldsymbol{\delta}}_{\mathbf{c}_x}^{(0)}, \hat{\boldsymbol{\delta}}_{\mathbf{c}_y}^{(0)}, \hat{\boldsymbol{\delta}}_{\mathbf{c}_z}^{(0)}$ centred on $\mathbf{c}^{(0)}$. This evolves as

$$\mathbf{c}(t) := \mathsf{ODE}_{\mathsf{Solve}}\big(\mathbf{f}_\theta, \mathbf{c}^{(0)}, t\big), \; t \in (0, T] \, , \tag{14}$$

with $\hat{\boldsymbol{\delta}}_{\mathbf{c}_x}(t), \hat{\boldsymbol{\delta}}_{\mathbf{c}_y}(t), \hat{\boldsymbol{\delta}}_{\mathbf{c}_z}(t)$ found using the ODE solver with the Jacobian of the neural vector field as for the surface normals in Eq. (8). The best rigid estimate minimises

$$\min_{\tilde{\mathbf{q}} \in \mathbb{Q}} \| \mathbf{B}_{\mathbf{c}}^{(0)} - \tilde{\mathbf{q}}^* \cdot \mathbf{B}_{\mathbf{c}}(t) \cdot \tilde{\mathbf{q}} \|^2 \, , \tag{15}$$

where $\mathbf{B}_{\mathbf{c}}(t) := [\hat{\boldsymbol{\delta}}_{\mathbf{c}_x}(t), \hat{\boldsymbol{\delta}}_{\mathbf{c}_y}(t), \hat{\boldsymbol{\delta}}_{\mathbf{c}_z}(t)]$ is the transformed basis and $\mathbf{B}_{\mathbf{c}}^{(0)} := [\hat{\boldsymbol{\delta}}_{\mathbf{c}_x}^{(0)}, \hat{\boldsymbol{\delta}}_{\mathbf{c}_y}^{(0)}, \hat{\boldsymbol{\delta}}_{\mathbf{c}_z}^{(0)}] \equiv \mathbf{I}$ is the identity basis.
Q-NET: Rather than have to solve for the optimal quaternion $\tilde{\mathbf{q}}$ in Eq. (15) for every point, we use a network to learn the optimal quaternion as a (smooth) function of space and time $\tilde{\mathbf{q}}_\phi(\mathbf{x}, t) : \mathbb{R}^{3\times 1} \rightarrow \mathbb{Q}$. This comprises as small specialist MLP, parameterised by weights ϕ. Specifically, we use the method of Peretroukhin *et al*. [53] who represent rotations through a symmetric matrix that defines a Bingham distribution over unit quaternions.

Tissue/Surface Loss: The soft tissue loss (Fig. 4) is

$$\mathcal{L}_{\text{soft}}(\theta, \phi) := \sum_{l,m} \|\tilde{\mathbf{B}}_{m,l} - \mathbf{I}\|^2 + \frac{1}{3} \sum_{n=1}^{3} \left(\left\| [\tilde{\mathbf{B}}_{m,l}]_n \right\| - 1 \right)^2 ,$$

$$\tilde{\mathbf{B}}_{m,l} := \tilde{\mathbf{q}}_\phi^*(\mathbf{c}_m^{(t_l)}, t_l) \cdot \mathbf{B}_{\mathbf{c}_m}(t_l) \cdot \tilde{\mathbf{q}}_\phi(\mathbf{c}_m^{(t_l)}, t_l) \, . \tag{16}$$

We evaluate over time samples $\{t_l\}$ for points randomly sampled in the soft tissue $\{\mathbf{c}_m^{(0)}\}$ as well as points on the surface $\{\mathbf{x}_i\}$. For surface points, $\mathcal{L}_{\text{surf}}(\theta, \phi)$ is the same as $\mathcal{L}_{\text{soft}}(\theta, \phi)$ with $\{\mathbf{c}_m^{(0)}\} \rightarrow \{\mathbf{x}_i\}$ but we align the first component of the basis vector with the surface normal and only consider the transformation in the 2D basis of the surface.

4.5. Full Loss Function

We jointly optimise the parameters of the ARC-Net, θ, defining the time-varying vector flow field, the Q-Net, ϕ, predicting rotation of conformal samples, and the translation $\tilde{\mathbf{s}}$ and joint rotations $\{\tilde{\mathbf{r}}_k\}$ of the forward kinematic skeleton. The full loss is given as

$$\mathcal{L}_{\text{full}}(\theta, \phi, \tilde{\mathbf{s}}, \{\tilde{\mathbf{r}}_k\}) := \mathcal{L}_{\text{var}}(\theta) + \lambda_1 \mathcal{L}_{\text{skel}}(\theta, \tilde{\mathbf{s}}, \{\tilde{\mathbf{r}}_k\}) + \lambda_2 \mathcal{L}_{\text{soft}}(\theta, \phi) + \lambda_3 \mathcal{L}_{\text{surf}}(\theta, \phi) \, , \tag{17}$$

where $\lambda_1, \lambda_2, \lambda_3 \geq 0$, are weighting parameters for each term; details are provided in the supplemental material.
Implementation Details: We use the JAX framework [7] and the Equinox library [36] for building the neural network architecture. In addition, the ProbDiffEq [39] library provides the probabilistic ODE solver. The loss function is optimised using VectorAdam [41], which adapts the basic Adam algorithm to be rotation-equivariant by accounting for the vector structure of optimisation variables. Additional details can be found in the supplementary material.

5. Experiments and Discussion

We evaluate against four existing approaches: SMS [12] (dense correspondence and interpolation, requires training data); ESA [28] (Varifold-based interpolation with-

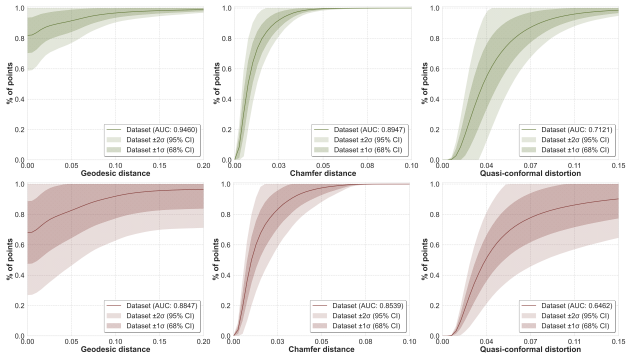

Figure 5. Interpolation results for DFAUST vs SMS [12]. *Mean and confidence intervals for the three metrics are shown; top row has our results and bottom show SMS. Our method improves across all metrics and also has narrower error bars indicating more consistent performance. Please zoom for details.*

out correspondence); Hamiltonian Dynamics [18] (high-quality interpolations, requires known dense correspondence); and Divergence Free [19] (volume preservation via divergence-free Eigen-basis). We use three standard datasets: DFAUST [4], MANO [59], and SMAL [79]; SMAL samples are from a modified version of the dataset [32]. To standardise across methods, all examples are normalised to fit a length-one cube. We use K-medoids [51] to select a diverse range of poses from each dataset and sample random pairs. We pick 80 shapes for training (for SMS only) and 20 for testing. For DFAUST, we select random pairs of poses across multiple subjects but ensure a pairing is of the same individual. We remove dataset mesh connectivity bias by remeshing each shape using ACVD [71].

Metrics: We quantify performance using three standard metrics via Area Under the Curve (AUC) values: *Geodesic Distance* between predicted and ground truth correspondences (assesses point-to-point mapping accuracy); *Chamfer Distance* between predicted and target surfaces (evaluates overall shape similarity); and *Conformal Distortion* resulting from the interpolation (measures preservation of local geometric properties). Further details, comprehensive results and analyses are in the supplementary material.

Shape Interpolation: Figure 5 shows the interpolation results for our method against the state-of-the-art SMS method for DFAUST showing the mean and confidence intervals for all metrics. Our method improves under all metrics with notably lower variance indicating more consistent performance as well. Equivalent plots for other comparisons are available in the supplemental material with the results summarised as AUC values for all comparators and datasets in the top half of Tab. 2. Our approach obtains superior dense correspondence recovery (geodesic distance) without sacrificing the quality of surface fit to the target mesh (chamfer distance) and whilst still conferring the superior theoretical properties of a diffeomorphic transformation. Equally, the results on local surface geometry are improved (conformal distortion). Qualitative interpolation results against SMS on SMAL are shown in the top of Fig. 1 where we see an example with significant non-rigid deformation that SMS is unable to model.

Ablation Study: Since the comparator methods do not require the provision of a source skeleton, Tab. 2 also includes comparisons showing the effects of omitting the skeleton and soft constraints. The big reduction in the conformal metric without the soft constraints strongly suggests that our physical differential priors are a key factor in recovering good local surface geometry. The benefit of the skeleton (without the soft constraints) is harder to infer from the reductive numbers in the table but centres on robustness; without the skeleton the Varifold can fall into local minima and fail to capture correct correspondences in challenging circumstances (e.g. fingers).

Surface Fidelity and Scaling: The bottom of Fig. 1 shows that we can match surfaces with vastly different resolutions (e.g. 7k matched to 500k vertices). Varifold compression allows us to perform this in a computationally efficient manner; we can see the Chamfer error results for 10k compressed mesh are visually indistinguishable from the full resolution but obtained at a huge reduction in computational (and memory) cost (plot on the right of Fig. 1).

Challenging Examples: In Fig. 6 we visualise the high quality interpolation path, correspondences and low cham-

Method	MANO			DFAUST			SMAL		
	Geodesic ↑	Chamfer ↑	Conformal ↑	Geodesic ↑	Chamfer ↑	Conformal ↑	Geodesic ↑	Chamfer ↑	Conformal ↑
Ours	**0.89** ± 0.17	**0.83** ± 0.09	**0.80** ± 0.13	**0.95** ± 0.09	**0.89** ± 0.06	**0.71** ± 0.17	**0.93** ± 0.09	**0.87** ± 0.05	**0.69** ± 0.08
SMS [12]	0.83 ± 0.23	0.80 ± 0.11	0.58 ± 0.17	0.88 ± 0.31	0.85 ± 0.11	0.65 ± 0.26	0.82 ± 0.12	0.79 ± 0.05	0.56 ± 0.12
ESA [28]	0.79 ± 0.35	0.61 ± 0.41	0.63 ± 0.24	0.76 ± 0.49	0.69 ± 0.33	0.70 ± 0.23	0.82 ± 0.18	0.66 ± 0.16	0.57 ± 0.15
Ham. Dyn. [18]	n/a	0.82 ± 0.05	0.62 ± 0.23	n/a	0.82 ± 0.25	0.68 ± 0.28	n/a	0.76 ± 0.21	0.59 ± 0.16
Div. Free [19]	n/a	0.50 ± 0.28	0.53 ± 0.28	n/a	0.62 ± 0.23	0.63 ± 0.28	n/a	0.58 ± 0.13	0.49 ± 0.19
OURS + ✗ SOFT ✗ SKEL	0.84 ± 0.17	0.77 ± 0.19	0.22 ± 0.32	0.92 ± 0.13	0.81 ± 0.09	0.33 ± 0.37	0.88 ± 0.18	0.77 ± 0.06	0.23 ± 0.24
OURS + ✗ SOFT ✓ SKEL	0.85 ± 0.20	0.78 ± 0.12	0.21 ± 0.33	0.91 ± 0.18	0.82 ± 0.10	0.34 ± 0.38	0.90 ± 0.15	0.77 ± 0.06	0.23 ± 0.25

Table 2. Area Under the Curve (AUC) metrics across all datasets. *AUC metrics are calculated with a threshold of 0.20, 0.1 and 0.15 for Geodesic, Chamfer and Conformal metrics respectively. The top shows comparisons with other methods, the bottom shows an ablation study with the soft tissue (SOFT) and skeleton terms (SKEL) removed.*

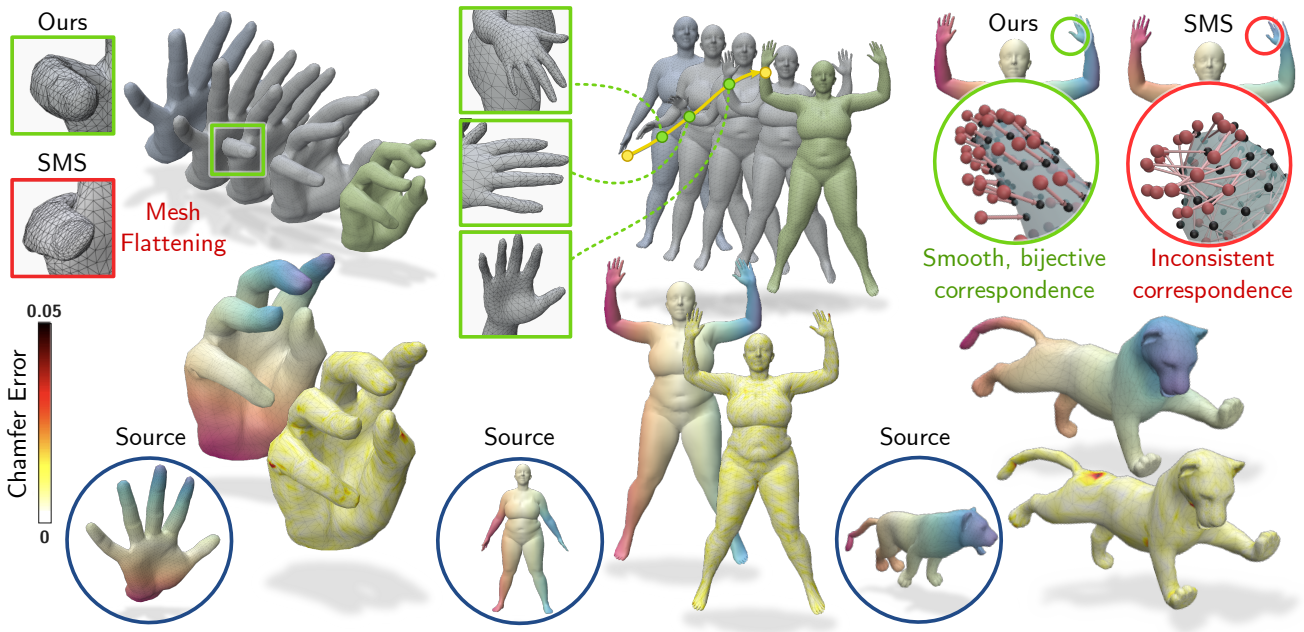

Figure 6. Challenging examples from each dataset. Top Left: Sample interpolations. *Even under large deformations, we do not suffer from flattening or "rubber hand" syndrome due to the articulated structure and the soft tissue priors.* ***Top Right: Correspondence detail.*** *We guarantee smooth, bijective correspondences by construction in contrast to inconsistent correspondences from permutation matrices.* ***Bottom: Correspondence and chamfer distances.*** *Accurate correspondences are recovered with low chamfer errors.*

fer errors on the most challenging examples (test set pairs that contain the largest average deformation) from each dataset (the SMAL interpolation is in Fig. 1). The offset correspondence visualisation (top right) shows the smooth bijjective correspondences guaranteed by construction from our method whereas permutation matrices can yield inconsistent correspondences (e.g. SMS).

Timing: Our method, without Varifold compression, yields runtimes comparable to ESA, requiring minutes to compute high-quality trajectories and dense correspondences between dataset samples. In contrast, SMS requires a minimum of 24 hours compute on high-performance GPU hardware to train its deep Functional Mapping network, though subsequent inference is rapid (under one minute). The Hamiltonian and Div-Free methods offer the fastest runtime for interpolation however they work from known correspondences making direct comparisons unsuitable. We note that SMS cannot train on high resolution meshes (due to the permutation matrix) whereas the Varifold compression removes this barrier for our approach; ESA has costly dense pairwise computations that also scale poorly.

Failure Cases: Whilst we found our method to be more robust than the comparator methods in general, there are potential failure cases as shown in Fig. 7. If the Varifold lengthscale is set incorrectly (given the mesh resolution) then artefacts can be introduced (as on the left). We cannot recover from topology errors in the source or target mesh (since this violates the diffeomorphic deformation; we show an example where the initial mesh had legs self-intersecting.

6. Conclusion

We have presented a novel unsupervised framework that learns realistic interpolation trajectories between different 3D shapes and computes accurate shape correspondences in an automatic manner and without relying on any prior knowledge. The effectiveness of the proposed approach was demonstrated using qualitative examples and quantitative analysis in various experiments, including human body shape and pose data as well as human hand scans.

Overall, we believe that our method will be a valuable contribution for bringing shape matching to practical applications in challenging real-world settings especially the ability to scale to high resolution surfaces and operate under arbitrary surface representations.

As future work, we plan to explore the initialisation and optimisation of the source skeleton, particularly using automatic skeleton generation techniques [45, 49, 76], and examine the impact of different skeletal structures.

Figure 7. Failure Cases. *Illustrations of failures when modelling assumptions are not met; e.g. the Varifold lengthscale is too small for mesh resolution or the input is topologically inconsistent.*

References

[1] William K Allard. On the first variation of a varifold. *Annals of mathematics*, 95(3):417–491, 1972. 3

[2] Seung-Yeob Baek, Jeonghun Lim, and Kunwoo Lee. Isometric shape interpolation. *Computers & Graphics*, 46:257–263, 2015. 3

[3] Martin Bauer, Nicolas Charon, Philipp Harms, and Hsi-Wei Hsieh. A numerical framework for elastic surface matching, comparison, and interpolation. *International Journal of Computer Vision*, 129(8):2425–2444, 2021. 2, 3

[4] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J Black. Dynamic faust: Registering human bodies in motion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6233–6242, 2017. 7

[5] Nathanael Bosch, Philipp Hennig, and Filip Tronarp. Calibrated adaptive probabilistic ode solvers. In *International Conference on Artificial Intelligence and Statistics*, pages 3466–3474. PMLR, 2021. 5

[6] Mario Botsch, Mark Pauly, Markus H Gross, and Leif Kobbelt. Primo: coupled prisms for intuitive surface modeling. In *Symposium on Geometry Processing*, pages 11–20, 2006. 3

[7] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. 6

[8] Christopher Brandt, Christoph von Tycowicz, and Klaus Hildebrandt. Geometric flows of curves in shape space for processing motion of deformable objects. In *Computer Graphics Forum*, pages 295–305. Wiley Online Library, 2016. 3

[9] Blanche Buet, Gian Paolo Leonardi, and Simon Masnou. A varifold approach to surface approximation. *Archive for Rational Mechanics and Analysis*, 226:639–694, 2017. 2

[10] Blanche Buet, Gian Paolo Leonardi, and Simon Masnou. Weak and approximate curvatures of a measure: a varifold perspective. *Nonlinear Analysis*, 222:112983, 2022. 2

[11] Dongliang Cao, Paul Roetzer, and Florian Bernard. Unsupervised learning of robust spectral shape matching. *ACM Transactions on Graphics (TOG)*, 2023. 2

[12] Dongliang Cao, Marvin Eisenberger, Nafie El Amrani, Daniel Cremers, and Florian Bernard. Spectral meets spatial: Harmonising 3d shape matching and interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3658–3668, 2024. 1, 2, 3, 6, 7, 15, 16, 17, 18, 19

[13] Benjamin Charlier, Jean Feydy, Joan Alexis Glaunes, François-David Collin, and Ghislain Durif. Kernel operations on the gpu, with autodiff, without memory overflows. *Journal of Machine Learning Research*, 22(74):1–6, 2021. 2

[14] Nicolas Charon and Alain Trouvé. The varifold representation of nonoriented shapes for diffeomorphic registration. *SIAM journal on Imaging Sciences*, 6(4):2547–2580, 2013. 2, 3

[15] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2018. 4

[16] Aharon Cohen and Mirela Ben-Chen. Robust shape collection matching and correspondence from shape differences. In *Computer Graphics Forum*, pages 555–568. Wiley Online Library, 2020. 2

[17] Stanley Durrleman, Xavier Pennec, Alain Trouvé, and Nicholas Ayache. Statistical models of sets of curves and surfaces based on currents. *Medical image analysis*, 13(5): 793–808, 2009. 3

[18] Marvin Eisenberger and Daniel Cremers. Hamiltonian dynamics for real-world shape interpolation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 179–196. Springer, 2020. 2, 3, 7

[19] M. Eisenberger, Z. Lähner, and D. Cremers. Divergence-free shape correspondence by deformation. In *Computer Graphics Forum*, pages 1–12, 2019. 2, 3, 5, 7

[20] Marvin Eisenberger, Zorah Lahner, and Daniel Cremers. Smooth shells: Multi-scale shape registration with functional maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12265–12274, 2020. 2

[21] Marvin Eisenberger, David Novotny, Gael Kerchenbaum, Patrick Labatut, Natalia Neverova, Daniel Cremers, and Andrea Vedaldi. Neuromorph: Unsupervised shape interpolation and correspondence in one go. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7473–7483, 2021. 2, 3

[22] Davide Eynard, Emanuele Rodola, Klaus Glashoff, and Michael M Bronstein. Coupled functional maps. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 399–407. IEEE, 2016. 2

[23] Jean Feydy, Joan Glaunès, Benjamin Charlier, and Michael Bronstein. Fast geometric learning with symbolic matrices. *Advances in Neural Information Processing Systems*, 33, 2020. 2

[24] Maolin Gao, Zorah Lahner, Johan Thunberg, Daniel Cremers, and Florian Bernard. Isometric multi-shape matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14183–14193, 2021. 2

[25] Pietro Gori, Olivier Colliot, Linda Marrakchi-Kacem, Yulia Worbe, Fabrizio De Vico Fallani, Mario Chavez, Cyril Poupon, Andreas Hartmann, Nicholas Ayache, and Stanley Durrleman. Parsimonious approximation of streamline trajectories in white matter fiber bundles. *IEEE transactions on medical imaging*, 35(12):2609–2619, 2016. 3

[26] Oshri Halimi, Or Litany, Emanuele Rodola, Alex M Bronstein, and Ron Kimmel. Unsupervised learning of dense shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4370–4379, 2019. 2

[27] Emmanuel Hartman, Emery Pierson, Martin Bauer, Nicolas Charon, and Mohamed Daoudi. Bare-esa: A riemannian framework for unregistered human body shapes. In

Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 14181–14191, 2023. 2

[28] 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. *International Journal of Computer Vision*, 131(5):1183–1209, 2023. 2, 3, 6, 7, 15, 16, 17, 18, 19

[29] Behrend Heeren, Martin Rumpf, Max Wardetzky, and Benedikt Wirth. Time-discrete geodesics in the space of shells. In *Computer Graphics Forum*, pages 1755–1764. Wiley Online Library, 2012. 3

[30] Behrend Heeren, Martin Rumpf, Peter Schröder, Max Wardetzky, and Benedikt Wirth. Exploring the geometry of the space of shells. In *Computer Graphics Forum*, pages 247–256. Wiley Online Library, 2014. 3

[31] Hsi-Wei Hsieh and Nicolas Charon. Metrics, quantization and registration in varifold spaces. *Foundations of Computational Mathematics*, 21(5):1317–1361, 2021. 3

[32] Qixing Huang, Xiangru Huang, Bo Sun, Zaiwei Zhang, Junfeng Jiang, and Chandrajit Bajaj. Arapreg: An as-rigid-as possible regularization loss for learning deformable shape generators. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5815–5825, 2021. 7

[33] Ruqi Huang, Jing Ren, Peter Wonka, and Maks Ovsjanikov. Consistent zoomout: Efficient spectral map synchronization. In *Computer Graphics Forum*, pages 265–278. Wiley Online Library, 2020. 2

[34] Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. Harmonic coordinates for character articulation. *ACM Transactions on Graphics (TOG)*, 26(3):71–es, 2007. 3

[35] Irene 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. 2

[36] Patrick Kidger and Cristian Garcia. Equinox: neural networks in JAX via callable PyTrees and filtered transformations. *Differentiable Programming workshop at Neural Information Processing Systems 2021*, 2021. 6

[37] Nicholas Krämer and Philipp Hennig. Stable implementation of probabilistic ode solvers. *Journal of Machine Learning Research*, 25(111):1–29, 2024. 5

[38] Nicholas Krämer, Nathanael Bosch, Jonathan Schmidt, and Philipp Hennig. Probabilistic ode solutions in millions of dimensions. In *International Conference on Machine Learning*, pages 11634–11649. PMLR, 2022. 5

[39] Peter Nicholas Krämer. *Implementing probabilistic numerical solvers for differential equations*. PhD thesis, Universität Tübingen, 2024. 6

[40] Hamid Laga. A survey on nonrigid 3d shape analysis. *Academic Press Library in Signal Processing*, 6:261–304, 2018. 3

[41] Selena Zihan Ling, Nicholas Sharp, and Alec Jacobson. Vectoradam for rotation equivariant geometry optimization. *Advances in Neural Information Processing Systems*, 35:4111–4122, 2022. 6, 13

[42] Or Litany, Tal Remez, Emanuele Rodola, Alex Bronstein, and Michael Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE international conference on computer vision*, pages 5659–5667, 2017. 2

[43] Or Litany, Emanuele Rodolà, Alexander M Bronstein, and Michael M Bronstein. Fully spectral partial shape matching. In *Computer Graphics Forum*, pages 247–258. Wiley Online Library, 2017. 2

[44] Zhen Liu, Hao Zhu, Qi Zhang, Jingde Fu, Weibing Deng, Zhan Ma, Yanwen Guo, and Xun Cao. Finer: Flexible spectral-bias tuning in implicit neural representation by variable-periodic activation functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2713–2722, 2024. 5, 12

[45] Jing Ma and Dongliang Zhang. Tarig: Adaptive template-aware neural rigging for humanoid characters. *Computers & Graphics*, 114:158–167, 2023. 8

[46] Robin Magnet, Jing Ren, Olga Sorkine-Hornung, and Maks Ovsjanikov. Smooth non-rigid shape matching via effective dirichlet energy optimization. In *2022 International Conference on 3D Vision (3DV)*, pages 495–504. IEEE, 2022. 2

[47] Simone Melzi, Jing Ren, Emanuele Rodola, Abhishek Sharma, Peter Wonka, and Maks Ovsjanikov. Zoomout: Spectral upsampling for efficient shape correspondence. *ACM Transactions on Graphics*, 2019. 2

[48] Pietro Musoni, Riccardo Marin, Simone Melzi, and Umberto Castellani. A functional skeleton transfer. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 4(3):1–15, 2021. 2

[49] Stefan Novaković and Vladimir Risojević. Learning localization of body and finger animation skeleton joints on three-dimensional models of human bodies. In *2024 Zooming Innovation in Consumer Technologies Conference (ZINC)*, pages 19–24. IEEE, 2024. 8

[50] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (ToG)*, 31(4):1–11, 2012. 2

[51] Hae-Sang Park and Chi-Hyuck Jun. A simple and fast algorithm for k-medoids clustering. *Expert systems with applications*, 36(2):3336–3341, 2009. 7

[52] Allen Paul, Neill Campbell, and Tony Shardlow. Sparse nystrom approximation of currents and varifolds. *arXiv preprint arXiv:2406.09932*, 2024. 3, 5, 14, 20

[53] Valentin Peretroukhin, Matthew Giamou, David M. Rosen, W. Nicholas Greene, Nicholas Roy, and Jonathan Kelly. A Smooth Representation of SO(3) for Deep Rotation Learning with Uncertainty. In *Proceedings of Robotics: Science and Systems (RSS'20)*, 2020. 6, 12

[54] Emery Pierson, Mohamed Daoudi, and Sylvain Arguillere. 3d shape sequence of human comparison and classification using current and varifolds. In *European Conference on Computer Vision*, pages 523–539. Springer, 2022. 2

[55] Jing Ren, Adrien Poulenard, Peter Wonka, and Maks Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. *ACM Transactions on Graphics (ToG)*, 37(6):1–16, 2018. 2

[56] Jing Ren, Mikhail Panine, Peter Wonka, and Maks Ovsjanikov. Structured regularization of functional map computations. In *Computer Graphics Forum*, pages 39–53. Wiley Online Library, 2019. 2

[57] Jing Ren, Simone Melzi, Peter Wonka, and Maks Ovsjanikov. Discrete optimization for shape matching. In *Computer Graphics Forum*, pages 81–96. Wiley Online Library, 2021. 2

[58] Emanuele Rodolà, Luca Cosmo, Michael M Bronstein, Andrea Torsello, and Daniel Cremers. Partial functional correspondence. In *Computer graphics forum*, pages 222–236. Wiley Online Library, 2017. 2

[59] Javier Romero, Dimitris Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics*, 36(6), 2017. 7

[60] Jean-Michel Roufosse, Abhishek Sharma, and Maks Ovsjanikov. Unsupervised deep learning for structured shape matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1617–1627, 2019. 2

[61] Pierre Roussillon and Joan Alexis Glaunès. Representation of surfaces with normal cycles and application to surface registration. *Journal of Mathematical Imaging and Vision*, 61: 1069–1095, 2019. 2

[62] Michael Schober, Simo Särkkä, and Philipp Hennig. A probabilistic model for the numerical solution of initial value problems. *Statistics and Computing*, 29(1):99–122, 2019. 5

[63] Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, 1985. 5

[64] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020. 5, 12

[65] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, pages 109–116. Citeseer, 2007. 3, 6

[66] Olga Sorkine and Mario Botsch. Interactive shape modeling and deformation. In *Eurographics (Tutorials)*, pages 11–37, 2009. 3

[67] Gary KL Tam, Zhi-Quan Cheng, Yu-Kun Lai, Frank C Langbein, Yonghuai Liu, David Marshall, Ralph R Martin, Xian-Fang Sun, and Paul L Rosin. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *IEEE transactions on visualization and computer graphics*, 19(7):1199–1217, 2012. 2

[68] Filip Tronarp, Hans Kersting, Simo Särkkä, and Philipp Hennig. Probabilistic solutions to ordinary differential equations as nonlinear bayesian filtering: a new perspective. *Statistics and Computing*, 29:1297–1315, 2019. 5

[69] Filip Tronarp, Simo Särkkä, and Philipp Hennig. Bayesian ode solvers: the maximum a posteriori estimate. *Statistics and Computing*, 31(3):23, 2021. 5

[70] Marc Vaillant and Joan Glaunes. Surface matching via currents. In *Biennial international conference on information processing in medical imaging*, pages 381–392. Springer, 2005. 2

[71] Sébastien Valette, Jean-Marc Chassery, and Rémy Prost. Generic remeshing of 3d triangular meshes with metric-dependent discrete voronoi diagrams. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):369–381, 2008. 7

[72] Oliver Van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A survey on shape correspondence. In *Computer graphics forum*, pages 1681–1707. Wiley Online Library, 2011. 2

[73] Amir Vaxman, Christian Müller, and Ofir Weber. Conformal mesh deformations with möbius transformations. *ACM Transactions on Graphics (TOG)*, 34(4):1–11, 2015. 6

[74] Matthias Vestner, Roee Litman, Emanuele Rodola, Alex Bronstein, and Daniel Cremers. 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*, pages 3327–3336, 2017. 2

[75] Benedikt Wirth, Leah Bar, Martin Rumpf, and Guillermo Sapiro. A continuum mechanical approach to geodesics in shape space. *International Journal of Computer Vision*, 93: 293–318, 2011. 3

[76] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. Rignet: neural rigging for articulated characters. *ACM Transactions on Graphics (TOG)*, 39 (4):58–1, 2020. 8

[77] Yusuke Yoshiyasu, Wan-Chun Ma, Eiichi Yoshida, and Fumio Kanehiro. As-conformal-as-possible surface registration. In *Computer Graphics Forum*, pages 257–267. Wiley Online Library, 2014. 6

[78] Laurent Younes. *Shapes and diffeomorphisms*. Springer, 2010. 3

[79] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6365–6373, 2017. 7

ARC-Flow : Articulated, Resolution-Agnostic, Correspondence-Free Matching and Interpolation of 3D Shapes Under Flow Fields

Supplementary Material

S1. Implementation Details

In this section, we provide implementation details that were omitted from the main text due to space constraints.

S1.1. ARC-Net Architecture

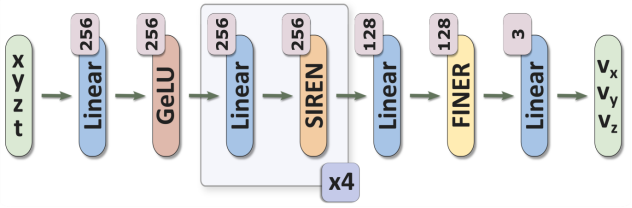

Figure 8. Overview of the ARC-Net Architecture.

The ARC-Net introduced in Section 4.1 comprises an MLP with 4 SIREN layers and one FINER layer, with widths of 256 and 128 respectively, as shown in Fig. 8. SIREN [64] uses a sine as a periodic activation function such that $\mathbf{z}_i : \mathbb{R}^{M_i} \rightarrow \mathbb{R}^{N_i}$ is the i-th layer of the network is defined as,

$$\mathbf{z}_i = \sin(\omega_0(\mathbf{W}_i\mathbf{z}_{i-1} + \mathbf{b}_i)) \,, \tag{18}$$

where $\mathbf{z}_{i-1}$ denotes the output of layer $i-1$ and ω_0 is a user defined parameter for controlling the frequency. For ARC-NET, we use $\omega_0 = 4.0$.

However, the standard formulation exhibits a well-documented spectral bias, wherein the network preferentially learns low-frequency components of the signal. While this bias can be advantageous for learning smooth flow fields, it potentially limits the network's ability to handle fine grain deviations required to handle intricate surface details on target surfaces. Thus, to address this limitation, we append a FINER layer [44] as the final layer, which replaces SIREN's conventional sine activation with a variable-periodic activation function,

$$\begin{aligned} \mathbf{z}_i &= \sin(\omega_0\alpha_i(\mathbf{W}_i\mathbf{z}_{i-1} + \mathbf{b}_i)) \,, \\ \text{where } \alpha_i &= |\mathbf{W}_i\mathbf{z}_{i-1} + \mathbf{b}_i| + 1 \end{aligned} \tag{3}$$

S1.2. Q-Net Architecture

Rotations in three-dimensional space can be represented through various mathematical formulations, including Euler angles, rotation matrices, axis-angle vectors, and unit quaternions. Our model employs unit quaternions due to their compact representation and straightforward geometric and algebraic properties. However, despite these advantages, it is known that naively attempting to learn large rotations via a neural network is problematic due to a critical limitation in the smoothness characteristics of unit quaternions, which can impede the network's ability to accurately learn the correct rotation.

Thus, the Q-Net discussed in Section 4.4 leverages the work of Peretroukhin *et al*. [53]. The network architecture comprises three fully connected layers, each with a width of 128 neurons using Tanh activation functions. Provided with a 4d input, (x, y, z, t), the output of the MLP produces a 10-dimension output which is used to construct the following 4×4 symmetric matrix,

$$\mathbf{A}(\theta) = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \theta_4 \\ \theta_2 & \theta_5 & \theta_6 & \theta_7 \\ \theta_3 & \theta_6 & \theta_8 & \theta_9 \\ \theta_4 & \theta_7 & \theta_9 & \theta_{10} \end{bmatrix} , \tag{19}$$

This represents the set of real symmetric 4×4 matrices with a simple (i.e., non-repeated) minimum eigenvalue:

$$\mathbf{A} \in \mathbb{S}^4 : \lambda_1(\mathbf{A}) \neq \lambda_2(\mathbf{A}), \tag{1}$$

where λ_i are the eigenvalues of $\mathbf{A}$ arranged such that $\lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \lambda_4$, and $\mathbb{S}^n \triangleq \{\mathbf{A} \in \mathbb{R}^{n\times n} : \mathbf{A} = \mathbf{A}^\top\}$.

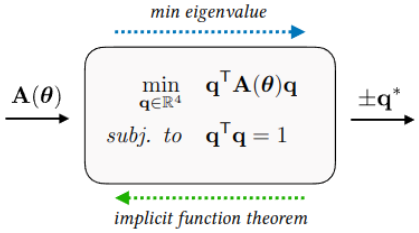

***Figure 9. QCQP layer** - Image Credit [53]*

$\mathbf{A}(\theta)$ is mapped to a unique rotation through a differentiable QCQP layer, illustrated in Figure 9. This layer predicts a quaternion as a solution derived from the minimum-eigenspace of $\mathbf{A}$ and the implicit function theorem allows for an analytic gradient to be computed for use as part of back-propagation,

$$\frac{\partial \mathbf{q}^*}{\partial \text{vec}(\mathbf{A})} = \mathbf{q}^* \otimes (\lambda_1\mathbf{I} - \mathbf{A})^\dagger \,, \tag{9}$$

where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudo-inverse, $\otimes$ is the Kronecker product, and $\mathbf{I}$ refers to the identity matrix.

S1.3. Skeleton Parameterisation

We utilise the skeleton provided with each dataset with a minor adjustment. To enhance realism, we extend the existing skeleton by adding bones at the extremities; specifically

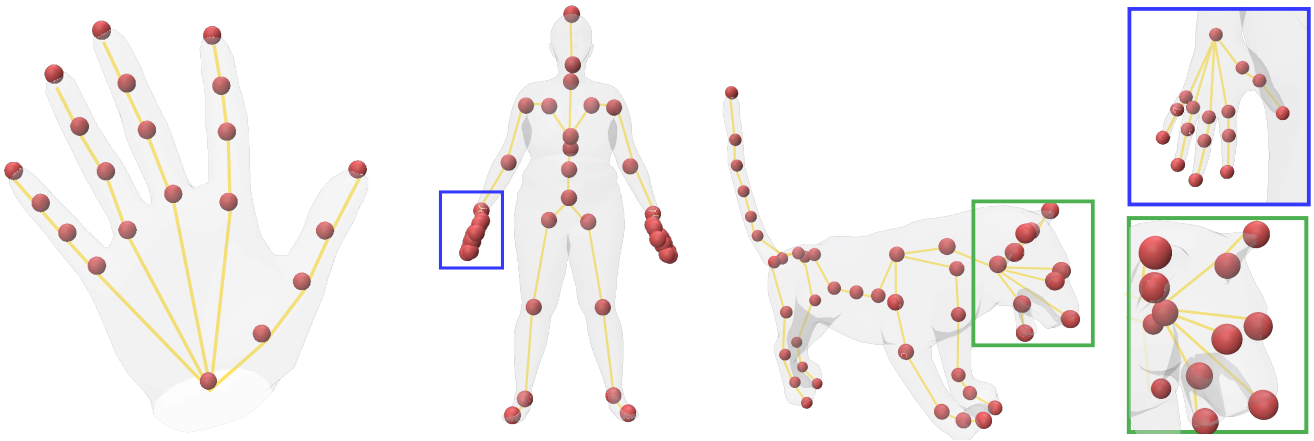

Figure 10. The simple skeletons, $\mathbb{S}_{\mathcal{X}}$, based upon skeleton provided with each dataset, used to augment the source shapes in our method. Left: MANO. *consisting of 21 joints and 21 bones.* ***Centre: DFAUST.*** *consisting of 53 joints and 52 bones.* ***Right: SMAL.*** *consisting of 44 joints and 43 bones.*

in the fingers, feet, and paws. This augmentation results in a more anatomically accurate representation, more closely mimicking real-life skeletal structures as shown in Fig. 10.

The exact regions defined as rigid / soft tissue and the number of samples used are summarised in Table 3. In general, for each bone, we use 50 samples for the bone, $\{\alpha_m\}$, and the soft tissue, $\{c_m\}$, components, while 500 samples are used for the surface points, $\{x_i\}$. All of which are randomly resampled in every epoch. We use a radius of 10% of the length of each bone and between 10% and 25% depending on the dataset for the soft tissue region. Human fingers are relative to the rest of a human body long and narrow, thus for DFAUST dataset they require significantly smaller regions to stay within the surface.

S1.4. Training details

In this section, we provide further details of the training procedure and parameters used to fit our model to the various datasets tested, a summary which is provided in Table 4.

As discussed in the main text, the loss function, Equation (17), is optimised using the VectorAdam [41] algorithm and the ODE modelling of the flow field is solved using a probabilistic ODE solver, specifically the Kronecker EK0 formulation with a single derivative operating with a smoother strategy. We use a variable learning rate, which is controlled via a warmup cosine decay schedule, in which 50 epochs are assigned to the warm up stage and the initial and final rates for different datasets are shown in the aforementioned table.

The model is optimised in two stages; main and fine-tuning (FT). During the main phase, after 1k, 2k and 3k epochs, the length scales of the Varifold kernels are adjusted in a coarse to fine manner, after which it is held constant for the remainder of the optimisation. Although many parameters vary slightly depending on the dataset, the weightings of $\mathcal{L}_{\text{bone}}$, $\mathcal{L}_{\text{soft}}$, and $\mathcal{L}_{\text{surf}}$, represented by $\boldsymbol{\lambda_1}$, $\boldsymbol{\lambda_2}$, and $\boldsymbol{\lambda_3}$ respectively, are consistent across all datasets.

Following the completion of the main stage, the values of $\boldsymbol{\lambda_1}$ and $\boldsymbol{\lambda_2}$ are increased (all other parameters are held constant), and the network is trained for an additional 2k epochs with these increased values. This additional fine-tuning step was found to improve the quality of the interpolation, both qualitatively and quantitatively (via the conformal metric). Starting initially with these higher values was problematic, as they place a high cost on any initial deformation of the source surface towards the target.

Dataset	Rigid $\{\alpha_m\}$		Soft Tissue $\{c_m\}$	
	Radius	# Samples	Radius	# Samples
MANO	0.1	50	0.25	50
DFAUST (Body)	0.1	50	0.25	25
DFAUST (Hands)	0.01	50	0.02	25
SMAL	0.1	50	0.15	50

Table 3. Parameters for Rigid & Soft Tissue Sampling: *The radii as defined in terms of the percentage of the bone length and used for both the bone and soft tissue regions.*

S2. Quaternion Interpolation Derivation

Section 4.3 discusses how the locations of samples modelling bones are interpolated under quaternion rotations. In this section, we provide additional mathematical background on the derivation of these formulae.

Given a point $\mathbf{p}_0$ and a quaternion representing rotation $\mathbf{q}$, the position of the point after the rotation has been applied is:

$$p_1 = \mathbf{q} \cdot \mathbf{p}_0 \cdot \mathbf{q}^* \tag{20}$$

Dataset	Epochs		LR		ODE	Initial		Epoch 1k		Epoch 2k		Epoch 3k	
	Main	FT	Init	Final	Steps	ℓ_{κ_x}	ℓ_{κ_n}	ℓ_{κ_x}	ℓ_{κ_n}	ℓ_{κ_x}	ℓ_{κ_n}	ℓ_{κ_x}	ℓ_{κ_n}
MANO	4k	2k	1e-2	1e-3	10	0.5	0.5	0.1	0.5	0.1	0.4	0.1	0.3
DFAUST	5k	2k	5e-3	1e-4	15	0.5	0.5	0.25	0.5	0.1	0.4	0.1	0.3
SMAL	4k	2k	5e-3	1e-4	10	0.5	0.5	0.25	0.5	0.1	0.4	0.1	0.3

Opt	λ_1	λ_2	λ_3
Main	2e2	1e1	5e3
FT	1e3	1e2	5e3

Table 4. Left: *Hyper-parameters & Varifold settings used for training on each dataset.* ***Right:*** *Loss function weightings for the two stages of the optimisation; main & fine tuning, across all datasets.*

If we assume that over $t \in [0,1]$ we apply a rotation of $\mathbf{q}$ and translation of $\mathbf{T}$, then the path traced out will be

$$\mathbf{p}(t) = t\mathbf{T} + \mathbf{q}(t) \cdot \mathbf{p}_0 \cdot \mathbf{q}^*(t) \tag{21}$$

Now we want to find the velocity field $\mathbf{f}_{\mathbf{q}}(\mathbf{p}, t)$ that will trace out the same trajectory for any initial $\mathbf{p}_0$. Thus we have

$$\mathbf{p}_T = \mathbf{p}_0 + \int_0^t \mathbf{f}(\mathbf{p}, t)\, dt \tag{22}$$

Therefore,

$$\int_0^t \mathbf{f}(\mathbf{p}, t)\, dt = \mathbf{p}(t) - \mathbf{p}_0 = t\mathbf{T} + \mathbf{q}(t) \cdot \mathbf{p}_0 \cdot \mathbf{q}^*(t) \tag{23}$$

And hence,

$$\frac{\partial}{\partial t} \Rightarrow \mathbf{f}(\mathbf{p}, t) = \mathbf{T} + \frac{\partial}{\partial t}\Big[\mathbf{q}(t) \cdot \mathbf{p}_0 \cdot \mathbf{q}^*(t)\Big] \tag{24}$$

By product rule,

$$\frac{\partial}{\partial t}\Big[\mathbf{q}(t)\cdot\mathbf{p}_0\cdot\mathbf{q}^*(t)\Big] = \frac{\partial \mathbf{q}\,(t)}{\partial t}\cdot\mathbf{p}_0\cdot\mathbf{q}^*(t) + \mathbf{q}(t)\cdot\mathbf{p}_0\cdot\frac{\partial \mathbf{q}^*(t)}{\partial t} \tag{25}$$

Now, interpolating between between a unit identity quaternion and some new quaternion $\mathbf{q}$,

$$\mathbf{q}(t) = \text{SLERP}(\mathbf{1}, \mathbf{q}_0, t) = \left(\cos\frac{\alpha t}{2}, \vec{\mathbf{n}}\sin\frac{\alpha t}{2}\right) = \mathbf{q}_0^t\,, \tag{26}$$

where $0 \leq t \leq T$.

For a unit quaternion,

$$\mathbf{q}^t = \exp(\ln(\mathbf{q}) * t) \tag{27}$$

The derivative of the function $\mathbf{q}$, where $\mathbf{q}$ is a constant unit quaternion is,

$$\frac{\partial}{\partial t}\mathbf{q}^t = \ln(\mathbf{q}) \cdot \mathbf{q}^t = \ln(\mathbf{q}) \cdot \exp(\ln(\mathbf{q}) * t) \tag{28}$$

, where the quaternion forms of exp, log and to a power are,

$$\exp(\mathbf{q}) = \exp(s)\begin{pmatrix} \cos(|\mathbf{v}|) \\ \frac{\mathbf{v}}{|\mathbf{v}|}\sin(|\mathbf{v}|) \end{pmatrix}. \tag{29}$$

$$\ln(\mathbf{q}) = \begin{pmatrix} \ln(|\mathbf{q}|) \\ \frac{\mathbf{v}}{|\mathbf{v}|}\arccos\left(\frac{s}{|\mathbf{q}|}\right) \end{pmatrix}. \tag{30}$$

$$\mathbf{q}^{\mathbf{p}} = \exp(\ln(\mathbf{q}) \cdot \mathbf{p})\,. \tag{31}$$

Note that if p is in fact scalar, then the power is

$$\mathbf{q}^p = \exp(\ln(\mathbf{q}) * p)\,. \tag{32}$$

S3. Sparse Nyström Varifold Approximation

This section details the Sparse Nyström Approximation algorithm introduced by Paul *et al*. [52]. We recall Sec. 4.2 where we describe the varifold matching loss

$$\begin{aligned} \mathcal{L}_{\text{var}}(\theta) &:= d(\boldsymbol{\mathcal{X}}^{(T)}, \boldsymbol{\mathcal{Y}}) \\ &= \langle \mu_{\boldsymbol{\mathcal{X}}^{(T)} - \mu_{\boldsymbol{\mathcal{Y}}}}, \mu_{\boldsymbol{\mathcal{X}}^{(T)}} - \mu_{\boldsymbol{\mathcal{Y}}} \rangle_{\mathbb{V}^*} \\ &= \langle \mu_{\boldsymbol{\mathcal{X}}^{(T)}}, \mu_{\boldsymbol{\mathcal{X}}^{(T)}} \rangle_{\mathbb{V}^*} - 2\langle \mu_{\boldsymbol{\mathcal{X}}^{(T)}}, \mu_{\boldsymbol{\mathcal{Y}}} \rangle_{\mathbb{V}^*} \\ &\quad + \langle \mu_{\boldsymbol{\mathcal{Y}}}, \mu_{\boldsymbol{\mathcal{Y}}} \rangle_{\mathbb{V}^*}\,. \end{aligned} \tag{33}$$

Again, $\boldsymbol{\mathcal{X}}^{(T)}$ denotes the set of vertices, normals and differential surface areas that are obtained at time $t = T$ from pushing $\boldsymbol{\mathcal{X}}$ through the $\mathsf{ODE}_{\mathsf{Solve}}$ in Sec. 4.1. In practice, we do not need to calculate $\langle \mu_{\boldsymbol{\mathcal{Y}}}, \mu_{\boldsymbol{\mathcal{Y}}} \rangle_{\mathbb{V}^*}$ as it is a constant due to the target $\boldsymbol{\mathcal{Y}}$ remaining unchanged.

For the first two terms in Eq. (33), we use a discrete approximation of the inner product integrals in Eq. (5)

$$\langle \mu_{\boldsymbol{\mathcal{X}}}, \mu_{\boldsymbol{\mathcal{Y}}} \rangle \approx \sum_{i=1}^{I_{\boldsymbol{\mathcal{X}}}} \sum_{i'=1}^{I_{\boldsymbol{\mathcal{Y}}}} \kappa_{\mathbf{x}}(\mathbf{x}_i, \mathbf{y}_{i'})\, \kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\boldsymbol{\mathcal{X}}_i}, \hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}_{i'}}) s_{\boldsymbol{\mathcal{X}}_i}\, s_{\boldsymbol{\mathcal{Y}}_{i'}}\,, \tag{34}$$

where the summation has the computational complexity $\mathcal{O}(I_{\boldsymbol{\mathcal{X}}}\, I_{\boldsymbol{\mathcal{Y}}})$; this cost becomes very expensive when fitting to a very high resolution target (e.g. fitting a template to a raw point cloud scan) where the number of vertices on the target is far larger than the template, $I_{\boldsymbol{\mathcal{Y}}} \gg I_{\boldsymbol{\mathcal{X}}}$ and we seek to reduce this cost.

Sparse Approximation: The algorithm of Paul *et al*. [52] creates sparse approximations of Varifold representations significantly smaller than the input data while maintaining high accuracy. It employs a Ridge Leverage Score (RLS) approach to assess a data point's importance, which is used to create a Nyström approximation for the in the Reproducing Kernel Hilbert Space (RKHS), offering a computationally efficient and accurate approach to shape compression. For a comprehensive understanding of the theoretical foundations, including detailed mathematical proofs, readers are referred to the original paper.

Application to the target, results in a compressed approximation $\boldsymbol{\mathcal{Y}}^c = \{V_{\boldsymbol{\mathcal{S}}_{\boldsymbol{\mathcal{Y}}}\rfloor}, N_{\boldsymbol{\mathcal{S}}_{\boldsymbol{\mathcal{Y}}}\rfloor}, \boldsymbol{\beta}\}$ containing a subset of $I^c_{\boldsymbol{\mathcal{Y}}}$

of the original vertices and normals with a corresponding set of weights $\boldsymbol{\beta}$. The Varifold matching representation from Eq. (34) to a compressed target becomes

$$\langle \mu_{\boldsymbol{\mathcal{X}}}, \mu_{\boldsymbol{\mathcal{Y}}^c} \rangle \approx \sum_{i=1}^{I_{\boldsymbol{\mathcal{X}}}} \sum_{i'=1}^{I_{\boldsymbol{\mathcal{Y}}^c}} \kappa_{\mathbf{x}}(\mathbf{x}_i, \mathbf{y}_{i'}^c) \kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\boldsymbol{\mathcal{X}}_i}, \hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}_{i'}^c}) s_{\boldsymbol{\mathcal{X}}_i} \beta_{\boldsymbol{\mathcal{Y}}_j^c} , \tag{35}$$

where $I_{\boldsymbol{\mathcal{Y}}}^c \ll I_{\boldsymbol{\mathcal{Y}}}$ dramatically reducing the computational cost of calculating the Varifold loss required.

ALGORITHM 1: Varifold Compression Algorithm

Input:
$\boldsymbol{\mathcal{Y}}$: Uncompressed Data - $\{V_{\boldsymbol{\mathcal{Y}}}, N_{\boldsymbol{\mathcal{Y}}}, dS_{\boldsymbol{\mathcal{Y}}}\}$
n: Number of Samples in $\boldsymbol{\mathcal{Y}}$ $(= I_{\boldsymbol{\mathcal{Y}}})$
m: Desired Compressed Size $(= I_{\boldsymbol{\mathcal{Y}}^c})$
λ: Regularisation parameter
$\kappa_{\mathbf{x}}(\cdot,\cdot)$: Positional Kernel
$\kappa_{\mathbf{n}}(\cdot,\cdot)$: Normal Kernel
Output:
$\boldsymbol{\mathcal{Y}}^c$: Compressed Representation - $\{V_{\boldsymbol{\mathcal{Y}}^c}, N_{\boldsymbol{\mathcal{Y}}^c}, \boldsymbol{\beta}\}$

► **Compute leverage scores:**
$b_s \leftarrow \lfloor \sqrt{n} \rfloor$;
$b \leftarrow \lfloor n/s \rfloor$;
Split $\boldsymbol{\mathcal{Y}}$ into b random batches $\{\boldsymbol{\mathcal{Y}}_1, \ldots, \boldsymbol{\mathcal{Y}}_b\}$ of size b_s;
for $j = 1$ **to** b **do**
  for $i = 1$ **to** b_s **do**
    $\boldsymbol{\Lambda}_i \leftarrow \mathbf{K}_{i,\boldsymbol{\mathcal{Y}}_j} (\mathbf{K}_{i,\boldsymbol{\mathcal{Y}}_j} + \lambda \mathbf{I})^{-1}$;
    ► $\mathbf{K}_{i,\boldsymbol{\mathcal{Y}}_j} = \sum_{i' \in \boldsymbol{\mathcal{Y}}_j} \kappa_{\mathbf{x}}(\mathbf{y}_i, \mathbf{y}_{i'}) \kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}_i}, \hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}_{i'}})$
  end
end

► **Draw weighted samples:**
Define sampling distribution:
Let $X_i \sim p(W)$ where $p(X_i = s_j) = \frac{\boldsymbol{\Lambda}_j}{\sum_{k=1}^n \boldsymbol{\Lambda}_k}$;
$\boldsymbol{\mathcal{C}} \leftarrow \{\,\}$;
for $i = 1$ **to** m **do**
  $\{\mathbf{x}_s, \hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}_s}\} \leftarrow$ Draw a sample from $\boldsymbol{\mathcal{Y}}$ acc. to $p(W)$;
  Add $\{\mathbf{x}_s, \hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}_s}\}$ to $\boldsymbol{\mathcal{C}}$;
end

► **Calculate sample weights:**
$\boldsymbol{\beta} \leftarrow \mathbf{K}_{\boldsymbol{\mathcal{C}},\boldsymbol{\mathcal{C}}}^{-1} \mathbf{Y}$;
► $\mathbf{K}_{\boldsymbol{\mathcal{C}},\boldsymbol{\mathcal{C}}} = \sum_{i=1}^{I_{\boldsymbol{\mathcal{C}}}} \sum_{i'=1}^{I_{\boldsymbol{\mathcal{C}}}} \kappa_{\mathbf{x}}(\mathbf{c}_i, \mathbf{c}_{i'}) \kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\boldsymbol{\mathcal{C}}_i}, \hat{\mathbf{n}}_{\boldsymbol{\mathcal{C}}_{i'}})$
► $\mathbf{Y} = \sum_{i=1}^{I_{\boldsymbol{\mathcal{C}}}} \sum_{i'=1}^{I_{\boldsymbol{\mathcal{Y}}}} \kappa_{\mathbf{x}}(\mathbf{c}_i, \mathbf{x}_{i'}) \kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\boldsymbol{\mathcal{C}}_i}, \hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}_{i'}}) s_{\boldsymbol{\mathcal{Y}}_i} s_{\boldsymbol{\mathcal{Y}}_{i'}}$

return $\{V_{\boldsymbol{\mathcal{Y}}^c}, N_{\boldsymbol{\mathcal{Y}}^c}, \boldsymbol{\beta}\}$

Compression Process: The compression process, as outlined in Algorithm 1, consists of three main steps:

1. The RLS values for each input element are generated efficiently using a sampling method that avoids calculating the full all-pairwise matrices.
2. Control points are then selected using a weighted sampling approach, with the RLS score determining the probability of selection.
3. Updated weights are calculated for each selected control point.

Several parameters are required as input for the compression process. Firstly, the desired compression size $m < I_{\boldsymbol{\mathcal{Y}}}$ determines the final number of control points. Additionally, the length scales of the kernels, $\kappa_{\mathbf{x}}$ and $\kappa_{\mathbf{n}}$, need to be set (they are the same as in the matching algorithm $\ell_{\mathbf{x}}$ and $\ell_{\mathbf{n}}$). Finally, an approximation parameter λ is required; we used a default value of 1 for all our experiments.

S4. Additional results

This section presents additional results that were omitted from the main results section due to space limitations.

S4.1. More Quantitative Results

In Figures 11 to 13 the interpolation results for our method against the state-of-the-art SMS [12] and ESA [28] methods for the three datasets; MANO, DFAUST and SMAL, showing the mean and confidence intervals for all metrics.

Our method demonstrates improvement across all metrics for each dataset, showing both higher mean values and reduced variance in results.

The reduction in variance of our method can be attributed primarily to our method's superior performance on more challenging problems. This is illustrated in Fig. 14, where we plot individual curves for interpolations where we colour each line with a "difficulty rating" calculated based on the average vertex displacement (normalised to one). When dealing with shapes that undergo a larger degree of deformation, competing methods show a significant drop in the quality of their results; in contrast, our approach maintains its effectiveness, leading to more consistent performance across varying levels of problem difficulty. All approaches show a roughly monotone increase in performance as the difficulty rating decreases.

S4.2. Further Qualitative Results

In this section we provide additional qualitative results, highlighting some of the advantages of our method which may not be accounted for by performance metrics alone.

S4.3. Non-Intersection of Surfaces

In Fig. 15, we demonstrate how our method handles a leg lift scenario where the leg comes into contact with the stomach of the individual. Since we model deformation as a diffeomorphism, represented by a time-varying vector flow field, our approach guarantees non-intersection by construction. In contrast, the interpolation produced by the ESA method fails to maintain surface integrity, resulting in unrealistic overlapping and severe mesh distortions.

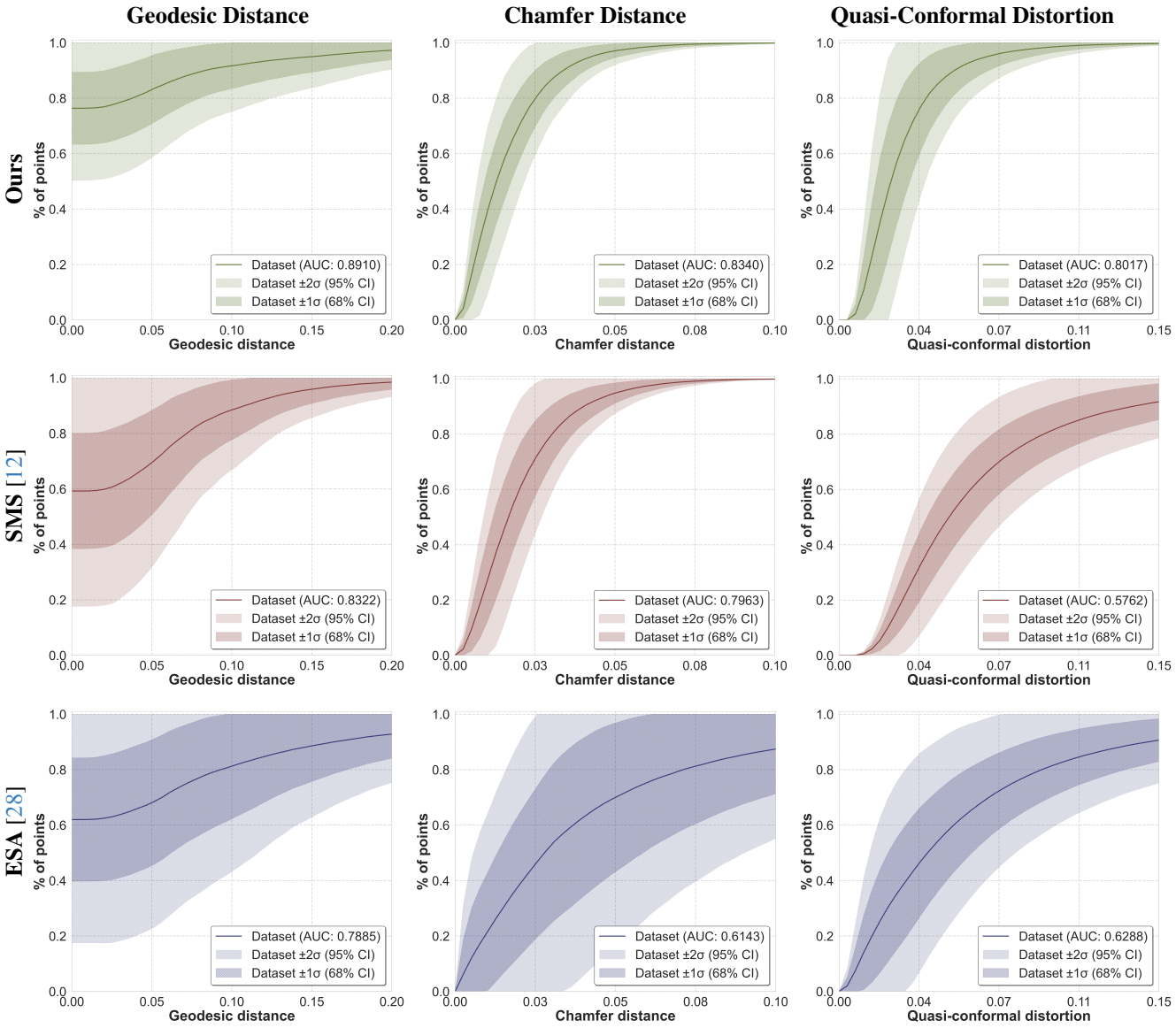

Figure 11. Interpolation results for MANO: Ours vs SMS [12] & ESA [28]. *Mean and confidence intervals for the three metrics are shown; top row has our results, middle SMS & bottom row ESA.*

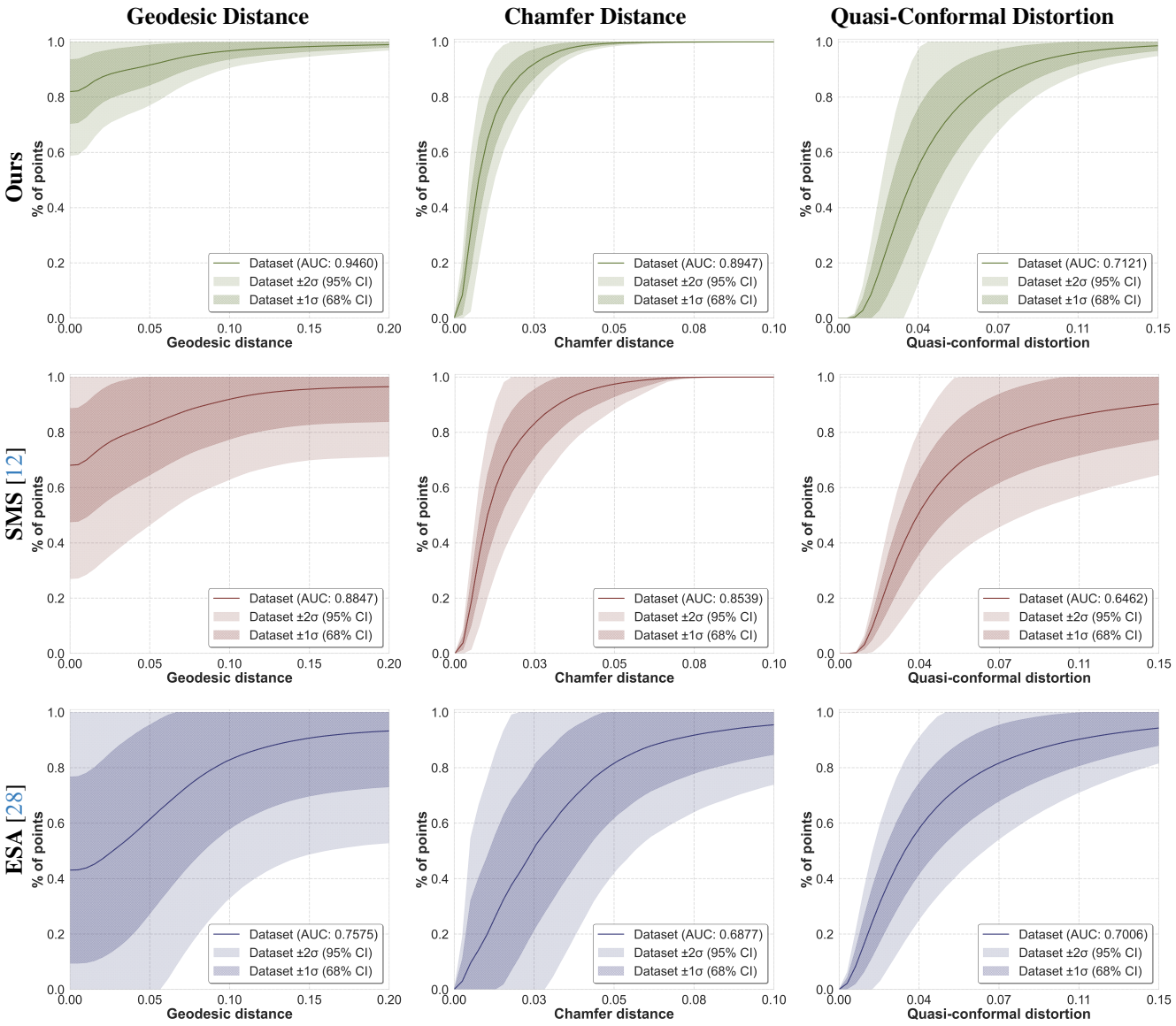

Figure 12. Interpolation results for DFAUST: Ours vs SMS [12] & ESA [28]. *Mean and confidence intervals for the three metrics are shown; top row has our results, middle SMS & bottom row ESA.*

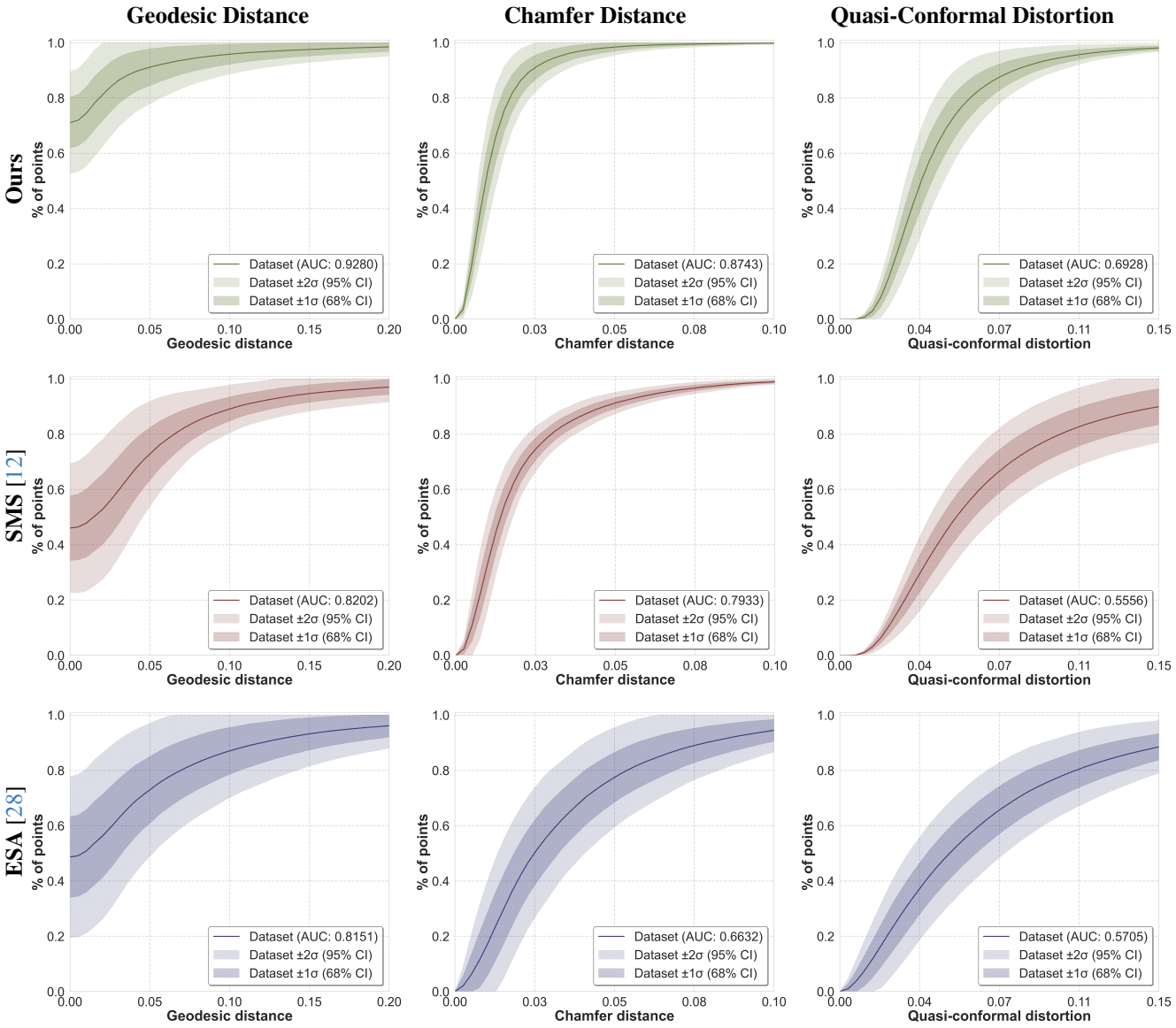

Figure 13. Interpolation results for SMAL: Ours vs SMS [12] & ESA [28]. *Mean and confidence intervals for the three metrics are shown; top row has our results, middle SMS & bottom row ESA.*

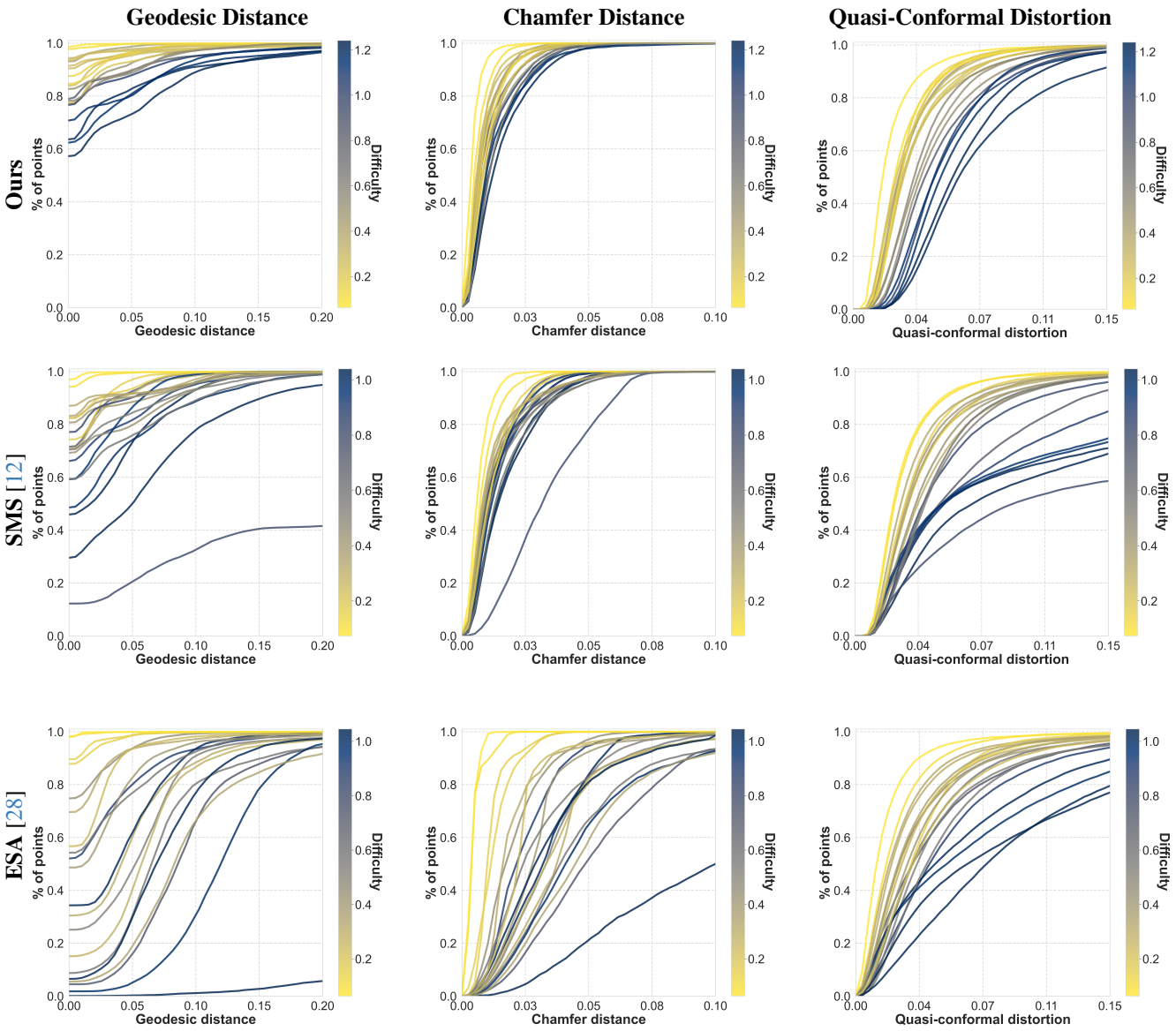

Figure 14. Interpolation results for DFaust: Ours vs SMS [12] & ESA [28]. *Plotting individual interpolations in which the difficulty of the problem is colour-coded; top row has our results, middle SMS & bottom row ESA. The difficulty rating is determined by the average vertex displacement for each interpolation task (from the ground-truth) normalised to one.*

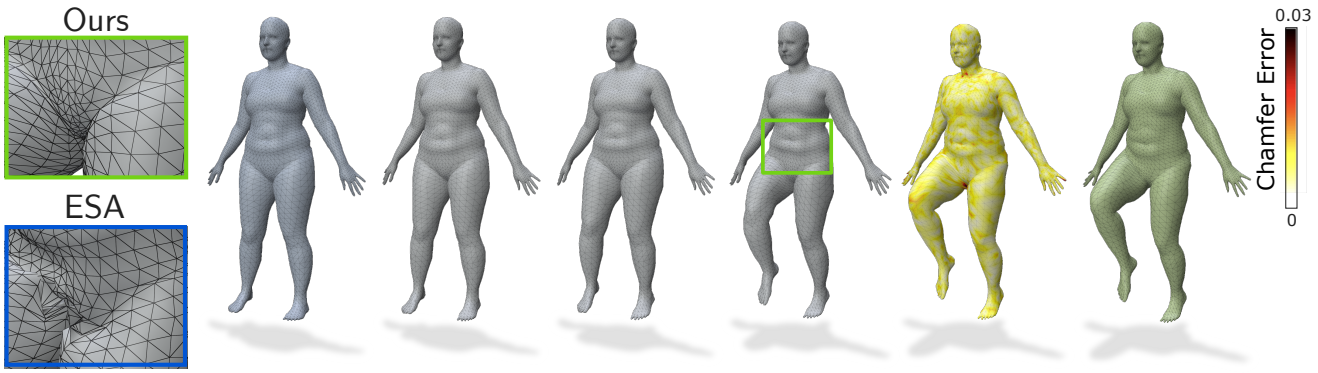

Figure 15. Non-Intersection of Surfaces: *We show interpolants from our method from the source on the left to target on the right (with final chamfer error on the second to right). As the leg is raised the interpolation from ESA fails to maintain surface integrity where the top of the leg meets the stomach resulting in unrealistic mesh distortions and overlapping (see closeup on bottom-left). Our diffeomorphic approach preserves topology by construction and guarantees non-intersection of the mesh (see closeup on top-left).*

S4.4. Fitting a Template to Topologically Noisy Data

A common approach in statistical shape analysis is to first bring all the raw data into correspondence by fitting a template to each sample. This also has the advantage that it removes noise and partial surfaces. However, this is a tricky task that often requires manual input.

We present an example demonstrating the potential for our method to automate this task. We select a neutral pose from the MANO dataset as our "template" and attempt to register this to a raw hand scan consisting of 38k vertices. Using the compression of Paul *et al*. [52], we form a 5k compressed representation as the target (in increasing computational efficiency).

Figure 16 illustrates the result of applying our approach, resulting in a high-quality registration. Although the majority of the surface fit has a very low Chamfer error, an area of higher error can be observed on the ring finger. This is due to the noise in the raw scan, where an unnatural bulge is clearly visible on one of the fingers. The use of a volume-preserving constraint by construction allows our method to fit the template despite this noise in the raw data.

Overall, as previously demonstrated there is no difference in the quality of the fit between using the full raw data and a Varifold compressed representation.

S4.5. Automatic Skeleton Transfer

The animation community has spent significant effort trying to ease rigging procedures. This is necessitated because the increasing availability of 3d data makes manual rigging infeasible. However, object animations involve understanding elaborate geometry and dynamics, and such knowledge is hard to infuse even with modern data-driven techniques. Automatic rigging methods do not provide adequate control and cannot generalise in the presence of unseen artefacts. An alternative approach is to learn to transfer an existing rig to a target using a dataset of known target poses to train a neural network.

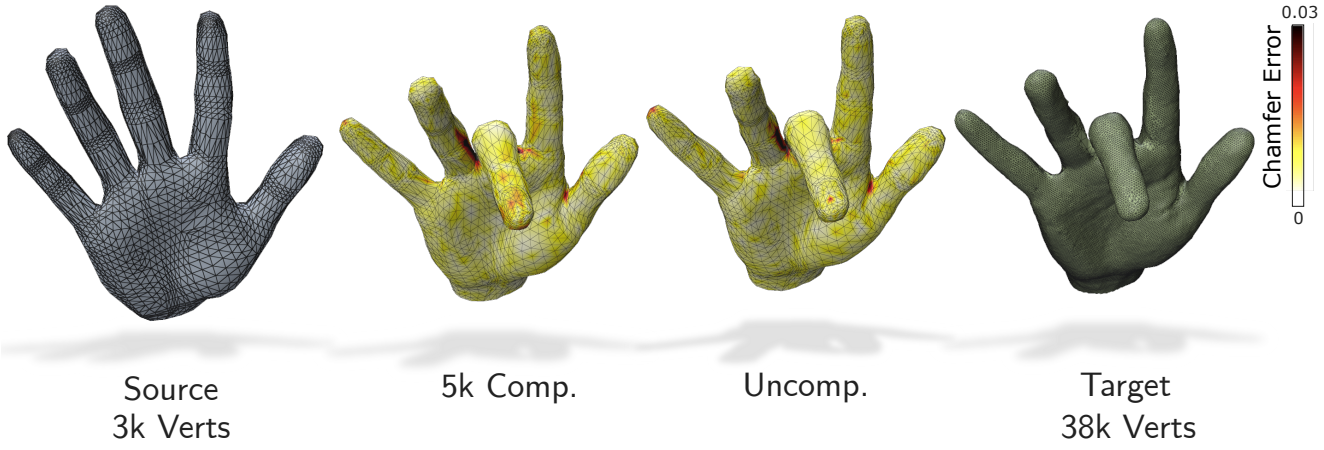

Figure 16. Fitting Template to Raw Scan Data. *We use a high-resolution noisy scan as the target to illustrate use of our method for template fitting. Varifold compression improves the efficiency of our approach with negligible change in final quality to using the full (uncompressed) target. Our method is robust to the noisy and partial data found in the dense target scan.*

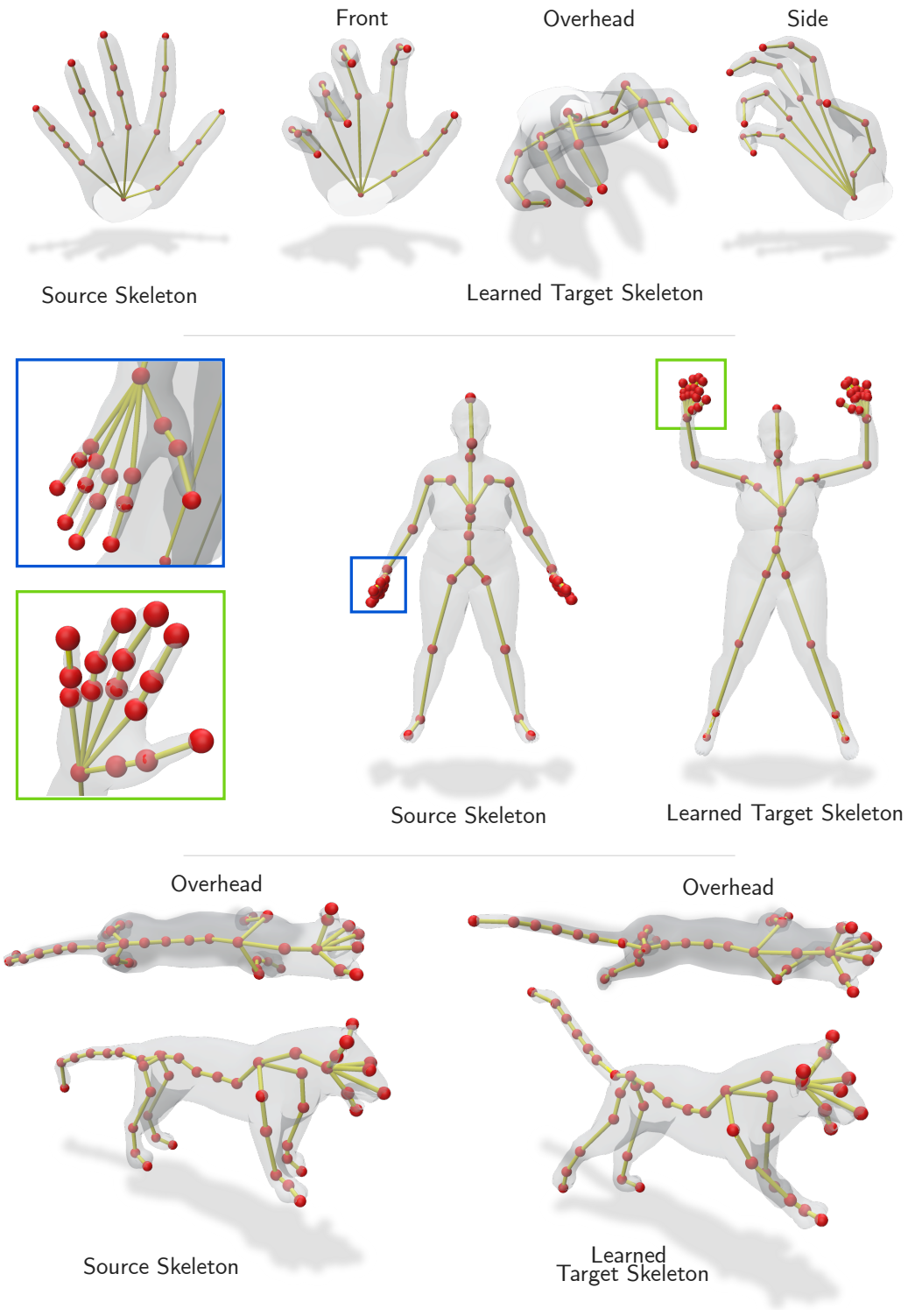

Figure 17. Learned Target Skeleton Examples: *Skeletons resulting from interpolations between poses involving* ***Top:*** *MANO,* ***Middle:*** *DFAUST &* ***Bottom:*** *SMAL datasets. The target skeleton is learned as a by-product of our method without any prior knowledge of the ground truth.*

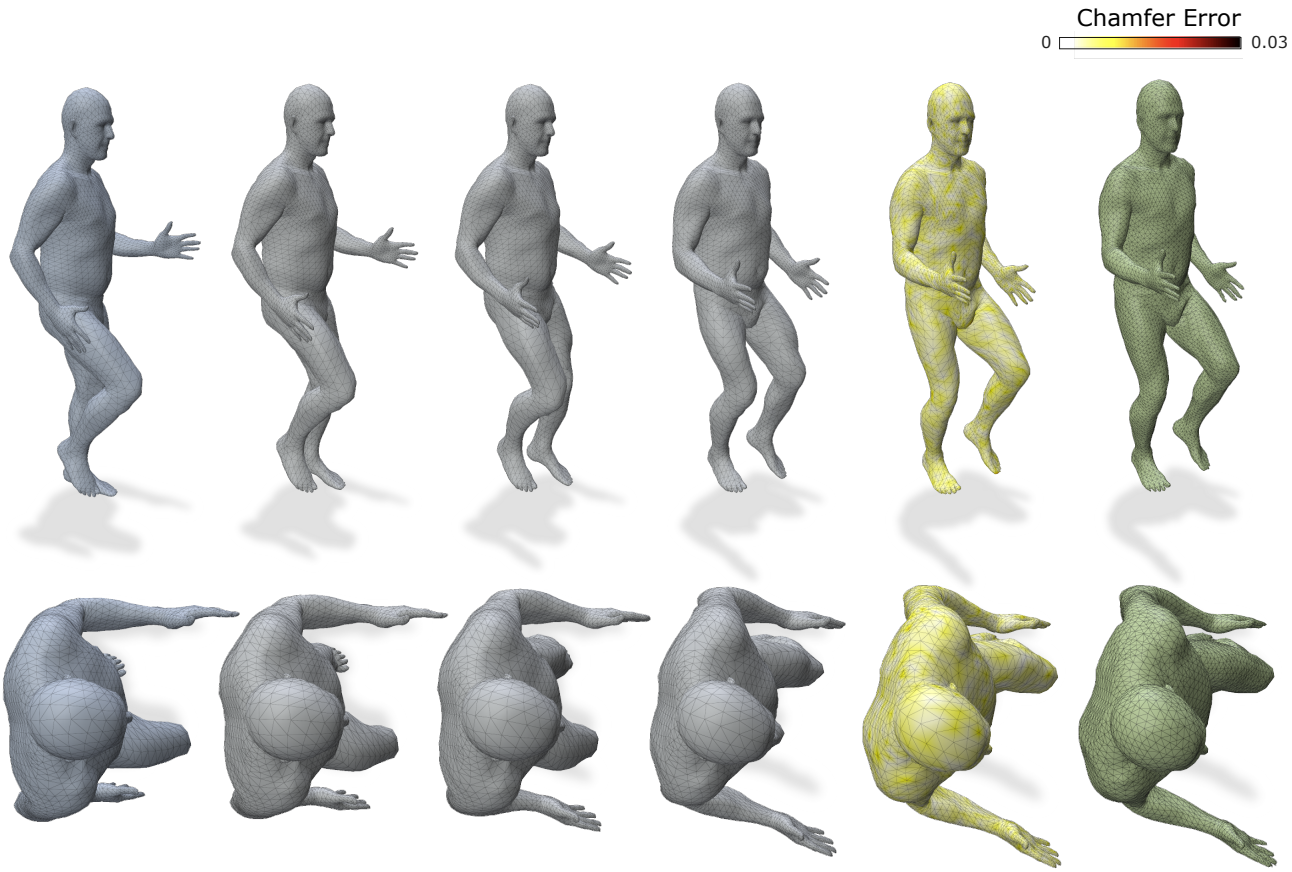

Figure 18. Interpolation Between Frames Capturing A Man Running on the Spot From DFAUST Dataset: *We show interpolants from our method from the source on the left to target on the right (with final chamfer error on the second to right).*

As part of our method the forward kinematics of the final skeletal pose (the global translation $\tilde{\mathbf{s}} \in \mathbb{R}^3$ and quaternion joint angles $\tilde{\mathbf{r}}_k \in \mathbb{Q}$ are optimised. As a result, we learn to transfer the source skeleton to the target as a by-product of our method. In Figure 17 we provide examples of these transferred skeletons, which appear in plausible configurations, and notably were achieved without any prior knowledge of ground truth target configurations.

S4.6. Additional Interpolations Examples

To further illustrate our findings, in Figures 18 to 20 we present additional interpolation examples generated using our method as a comprehensive showcase of our technique's capabilities.

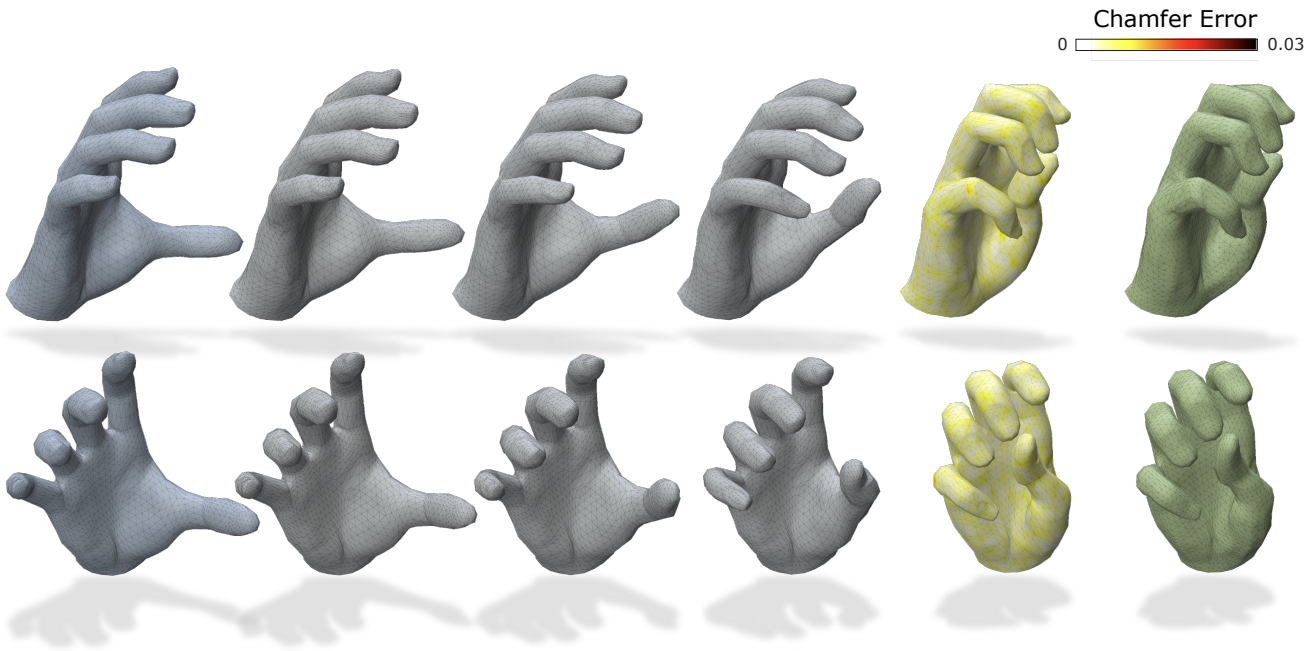

Figure 19. Interpolation Between An Open & Closed Hand Poses from the MANO Dataset: *We show interpolants from our method from the source on the left to target on the right (with final chamfer error on the second to right).*

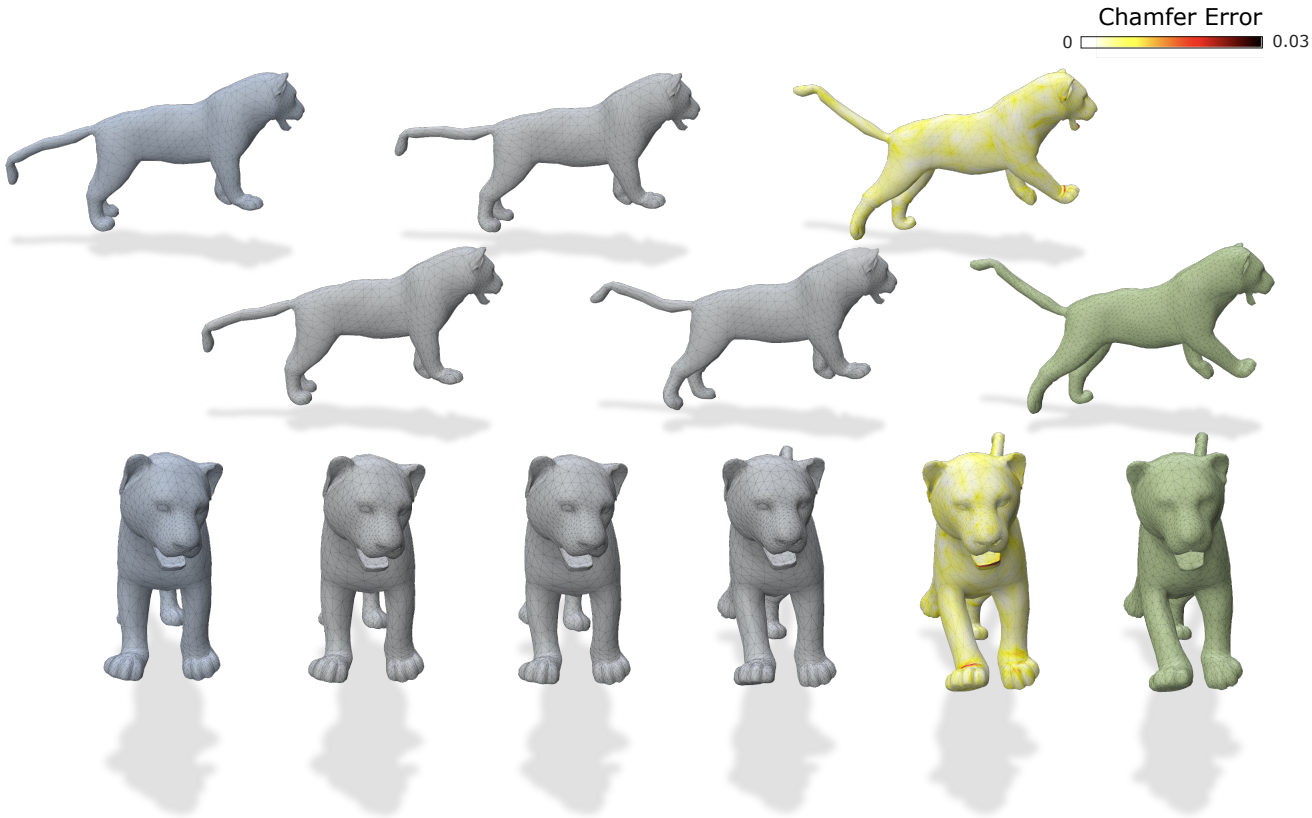

Figure 20. Interpolation Between Two Running Poses from the SMAL Dataset: *We show interpolants from our method from the source on the left to target on the right (with final chamfer error on the second to right).*