Curvature Enthusiasm: Correspondence-Free Interpolation and Matching of Articulated 3D Shapes using Compressed Normal Cycles

ADAM HARTSHORNE, ALLEN PAUL, and TONY SHARDLOW, University of Bath, UK
NEILL D F CAMPBELL, University College London and University of Bath, UK

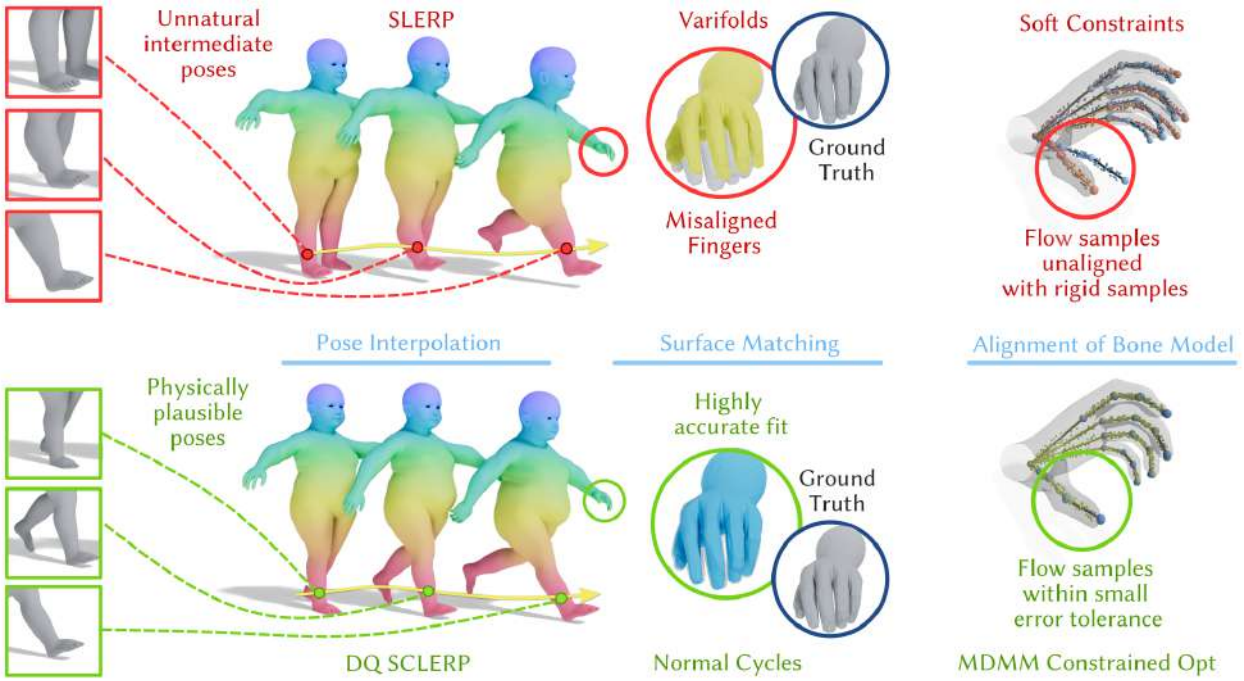

Fig. 1. *We propose 3 key innovations (bottom row) over the current state-of-the-art approach, ARC-Flow [Hartshorne et al. 2025] (top row), that lead to more physically plausible interpolations and higher quality recovery of dense correspondence.* ***Left: 3D Shape Interpolation.*** *The kinematics of the articulated skeleton are modelled using Dual Quaternions leading to pose interpolation using SCLERP (vs SLERP)* ***Centre: Improved Surface Matching.*** *Normal Cycles are more effective in accurately modelling high curvature areas compared to Varifolds* ***Right: Exact Skeleton-Driven Transformation.*** *The benefit of constrained optimisation (MDMM [Platt and Barr 1987] vs soft constraints) is demonstrated on a sequence from the MANO dataset; it ensures the flow driven deformation exactly matches the rigid skeletal deformation.*

We present an unsupervised framework for physically plausible shape interpolation and dense correspondence estimation between 3D articulated shapes. Our approach intentionally focuses upon pose variation within the same identity, which we believe is a meaningful and challenging problem in its own right. Our method uses Neural Ordinary Differential Equations (NODEs) to generate smooth flow fields that define diffeomorphic transformations, ensuring topological consistency and preventing self-intersections while accommodating hard constraints, such as volume preservation.

By incorporating a lightweight skeletal structure, we impose kinematic constraints that resolve symmetries without requiring manual skinning or predefined poses. We enhance physical realism by interpolating skeletal motion with dual quaternions and applying constrained optimisation to align the flow field with the skeleton, preserving local rigidity. Additionally, we employ an efficient formulation of Normal Cycles, a metric from geometric measure theory, to capture higher-order surface details like curvature, enabling precise alignment between complex articulated structures and recovery of accurate dense correspondence mapping.

Evaluations on multiple benchmarks show notable improvements over state-of-the-art methods in both interpolation quality and correspondence accuracy, with consistent performance across different skeletal configurations, demonstrating broad utility for shape matching and animation tasks.

CCS Concepts: • **Computing methodologies** → **Shape representations**; **Reconstruction**.

Additional Key Words and Phrases: Shape Interpolation, Shape Registration, 3D Articulated Shapes, Diffeomorphic Transformations, Neural Ordinary Differential Equations (NODEs), Geometric Measure Theory

Authors' addresses: Adam Hartshorne, ath35@bath.ac.uk; Allen Paul, ap2746@bath.ac.uk; Tony Shardlow, t.shardlow@bath.ac.uk, University of Bath, UK; Neill D F Campbell, neill.campbell@ucl.ac.uk, University College London and University of Bath, UK.

ACM 0730-0301/2025/12-ART230
https://doi.org/10.1145/3763366

ACM Reference Format:
Adam Hartshorne, Allen Paul, Tony Shardlow, and Neill D F Campbell. 2025. *Curvature Enthusiasm*: Correspondence-Free Interpolation and Matching of Articulated 3D Shapes using Compressed Normal Cycles. *ACM Trans. Graph.* 44, 6, Article 230 (December 2025), 24 pages. https://doi.org/10.1145/3763366

1 INTRODUCTION

Determining dense correspondences between between two articulated poses of the same shape identity is one of the fundamental problems of geometry processing, particularly when dealing with complex postural variations. In addition, modelling realistic non-rigid shape deformations 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.

A core challenge lies in developing efficient regularisation frameworks that ensure physically plausible trajectories while balancing target alignment with geometric feature preservation [Laga 2018; Sorkine and Botsch 2009].

Recent work by Hartshorne et al. [2025] proposed a framework for the unsupervised prediction of physically plausible interpolations between two 3D articulated shapes and the automatic estimation of their dense correspondence. They used smooth diffeomorphic fields to ensure topological consistency and align the deformed source shape to a target using a technique from geometric measure theory that enables comparison independent of shape fidelity.

We build upon their work by introducing several key innovations that mitigate weaknesses in their framework. We show that this leads to improved physical plausibility and numerical stability of unsupervised interpolations between 3D articulated shapes, while simultaneously enhancing the accuracy of automated dense correspondence estimation between them. Code for this paper is available at https://curvature-enthusiasm.github.io/.

Contributions

- Improving the quality of the dense surface matching, in particular in areas of high curvature, by replacing Varifolds with a Normal Cycle metric; the new metric also comes with a suitable compression scheme and, although slightly more expensive to compute, achieves superior results with fewer iterations, thus offsetting the additional computation cost.
- Bone (articulation) model with a superior kinematic formulation and rotation representation network that enforces physical constraints and improves accuracy.
- A constrained optimization procedure eliminates complex schedules and tuning; this adaptive approach is more robust delivering better results.
- Better sampling scheme for the skeleton and soft tissue; which improves physical compliance, efficiency, and further eliminates hyperparameters from the existing approach; making it less sensitive to the initial skeleton.

2 RELATED WORK

Reconstructing dynamic 3D motion between a source and target pose requires jointly addressing two interconnected challenges: interpolating a physically plausible trajectory between shapes and establishing accurate correspondences.

2.1 Non-Rigid 3D Shape Matching

Functional Mapping (FM): The original approach of Ovsjanikov et al. [2012] computed a fuzzy correspondence by aligning the spectral basis (Laplace-Beltrami Eigenfunctions) of isometrically deformed shapes, with dense pointwise correspondences recovered through ICP-based nearest neighbour search in functional space. Deep learning based approaches; both supervised [Donati et al. 2020; Groueix et al. 2018; Litany et al. 2017; Vestner et al. 2017a] and unsupervised [Cao and Bernard 2022; Cao et al. 2023; Li et al. 2022] are effective alternatives to hand-crafted feature descriptors, but require datasets for training.

Mapping shapes in the low-frequency spectral domain often leads to local geometrically inconsistent pointwise correspondences. Recent works address this via post-processing techniques [Eisenberger et al. 2020; Melzi et al. 2019; Vestner et al. 2017b] or also registering shapes in the spatial domain [Attaiki and Ovsjanikov 2024; Cao et al. 2024; Jiang et al. 2023] to try to ensure spatial smoothness of the pointwise correspondences. Traditional FM approaches are limited by their reliance on learning a point-to-point matrix, which are both memory-intensive for high-resolution meshes and ill-suited for shapes of very different resolutions. Previous mitigation strategies have included, subsampling strategies to reduce computational cost by selecting sparse sets of points on high-resolution meshes [Magnet and Ovsjanikov 2023], while the widely used ZoomOut algorithm [Melzi et al. 2019] has been adapted for GPU architectures, with memory-efficient modifications introduced to further enhance its scalability [Magnet and Ovsjanikov 2024]. Despite these advances, enforcing geometric consistency remains challenging, as it often requires solving large integer linear programs that are only tractable for low-resolution meshes with just a few thousand triangles [Roetzer and Bernard 2024]. Although coarse-to-fine strategies [Roetzer et al. 2024] extend scalability they typically introduce discretisation artefacts.

Geometric Measure Theory: Embedding shapes into measure spaces provides a mathematical framework for handling surfaces without explicit correspondences; this avoids issues with FM methods (e.g. specifying/learning features, permutation matrices and correspondence consistency). Currents [Vaillant and Glaunes 2005] and Varifolds [Charon and Trouvé 2013] have been used for computational anatomy and human shape analysis [Bauer et al. 2021; Hartman et al. 2023a; Pierson et al. 2022], but only capture first-order geometric information which restricts performance on complicated surfaces [Charon et al. 2020].

Normal Cycles: The concept of Normal Cycles (NCs) originates in Federer's curvature measures [Federer 1959], extended to integral currents by Zähle [1986], and later generalized to singular and polyhedral sets by Fu [1994] and Wintgen [1982]. Building on this foundation, Cohen-Steiner and Morvan introduced NCs as practical tools for curvature approximation on meshes [Cohen-Steiner and

Morvan 2003]. Beyond curvature estimation, NCs have more recently been applied to 3D surface matching [Roussillon and Glaunès 2017, 2019], where they were shown to outperform Current/Varifold based methods on highly curved or topologically complex shapes, albeit at tenfold increase in computational cost. Despite GPU-based acceleration strategies [Charlier et al. 2020], NCs have so far struggled to scale to real-world datasets. To address this limitation, Paul et al. [2025] adapt recent Varifold compression techniques [Paul et al. 2024] to the NC setting, providing the theoretical framework that motivates our approach.

2.2 Shape Interpolation

Geometric Methods: Traditional geometric approaches deform shapes by minimizing energy in well-defined shape spaces [Bauer et al. 2010; Beg et al. 2005]; they often seek geodesics in higher-dimensional spaces [Brandt et al. 2016; Heeren et al. 2014, 2012; Wirth et al. 2011] through local distortion measures such as ARAP [Sorkine and Alexa 2007] or PriMo [Botsch et al. 2006]. Although recent work simplifies initialisation [Laga et al. 2017] and reduces search spaces [Hartman et al. 2023a] to improve efficiency, the non-convex optimizations remain sensitive to initial conditions. Alternatives based on geometric measure theory, such as SRNF pseudo-distance [Bauer et al. 2021] and Varifold-based interpolation [Hartman et al. 2023b], avoid registration issues but scale poorly and rely on complex PDE-defined geodesics.

Vector Flow Fields: The representation of shape transformations through continuous flows is a well-established approach that predates the deep learning era. Central to our work is the integration of ODE-based fields to prevent self-intersections, building upon extensive prior research in shape transfer applications. Von Funck et al. [2006] exemplify this approach by constructing divergence-free fields as the cross-product of two gradient fields, which preserve volume by construction. To preserve diffeomorphisms, Gupta and Chandraker [2020] chain multiple ODE integrations to flow spherical meshes toward target shapes in a topology-preserving manner. Eisenberger et al. [2019] and Eisenberger and Cremers [2020] proposed a time-dependent gradient flow combining divergence-free fields with ARAP constraints, though this requires expensive per-step constraint computation. More recently, ARC-Flow [Hartshorne et al. 2025] leverages skeletal structures and Varifolds for estimating correspondence-free deformation, but struggles with fine detail recovery and the alignment between the skeleton and flow field.

Implicit Surfaces: While mesh-based approaches remain dominant due to their straightforward shape manipulation capabilities, implicit neural fields are a powerful alternative, representing surfaces as level-sets of scalar functions in $\mathbb{R}^3$ [Ma et al. 2020; Sitzmann et al. 2020; Yang et al. 2021]. These representations allow continuous surface sampling and computation of differential properties without the overhead of explicit representations.

Interpolating implicit surfaces has been addressed through: latent space interpolation [Liu et al. 2022], manipulation of user-defined handle points [Li et al. 2021], and level-set equations for deformation fields [Mehta et al. 2022; Novello et al. 2023]. While promising, these velocity based methods are restricted to pre-defined fields and struggle with non-linear translations. Sang et al. [2025] proposed diffeomorphic flow fields but require initial sparse correspondence (typically 10%) learned through explicit shape methods [Cao et al. 2024], while still facing difficulties with large deformations and introducing surface artefacts.

Deep Learning: Autoencoder architectures for shape deformation have been proposed, e.g. 3DCODED [Groueix et al. 2018], but methods struggle with large non-linear deformations. Neural Jacobian Fields [Aigerman et al. 2022] predict piecewise linear mappings between meshes with varying triangulations, but require training data, focus primarily on lower-resolution mappings, and can produce physically invalid results unlike our approach that prioritizes physically plausible interpolants alongside accurate correspondence.

Closely related works include NeuroMorph [Eisenberger et al. 2021], generating continuous interpolations with dense correspondences using dual networks for displacement fields, and SMS [Cao et al. 2024], enhancing with spectral-spatial map harmonisation. These methods rely on large datasets, need permutation matrices which limit scalability, and do not guarantee non-intersecting surface trajectories, volume preservation or smooth point-wise correspondence maps.

3 NORMAL CYCLES FOR SHAPE MATCHING

We seek a *matching metric* $d(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}})$ for two shapes, $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$, that is (i) independent of the particular choice of shape parameterisation, and (ii) does not rely on any known correspondence between locations on the two shapes.

This work presents the case of a 2D surface embedded in 3D space; whilst the majority of the surfaces are closed, our approach can also handle open surfaces (e.g. hands) and can be readily extended to arbitrary dimensions. The difference between shapes can be thought of as the variation that remains after the removal of translation, rotation and scale.

The formulation we present here comes from "Geometric Measure Theory" and includes approaches such as Currents, Varifolds and Normal Cycles (NC) as measures that are increasingly sensitive to subtle shape changes. Previous works, e.g. Charon and Trouvé [2013]; Hartshorne et al. [2025]; Kaltenmark et al. [2017], have considered the Varifold metric and we extend this to the more involved Normal Cycles. We provide a high-level overview for NCs with the detailed mathematical derivations left to the literature, e.g. Roussillon and Glaunès [2019].

3.1 Currents and Varifolds

Revisiting the standard current formulation, we start by looking at a vector field $\vec{\mathbf{v}}(\mathbf{x}) : \mathbb{R}^3 \to \mathbb{R}^3$ and taking the (oriented) surface integral over some surface $\boldsymbol{\mathcal{X}} \subset \mathbb{R}^3$ which is defined as

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

For ease of comparison, we attempt to keep notation consistent with Hartshorne et al. [2025] rather than the NC literature, e.g. Roussillon and Glaunès [2019]; thus $\mathbf{x} \in \boldsymbol{\mathcal{X}}$ runs over the surface with unit normal vector $\hat{\mathbf{n}}(\mathbf{x})$ and elemental area $dS_{\boldsymbol{\mathcal{X}}}(\mathbf{x})$. We represent the vector field with a Reproducing Kernel Hilbert Space (RKHS) $\mathbb{V}$, using some spatial kernel $\kappa(\mathbf{x}, \mathbf{x}')$, which allows us to define the

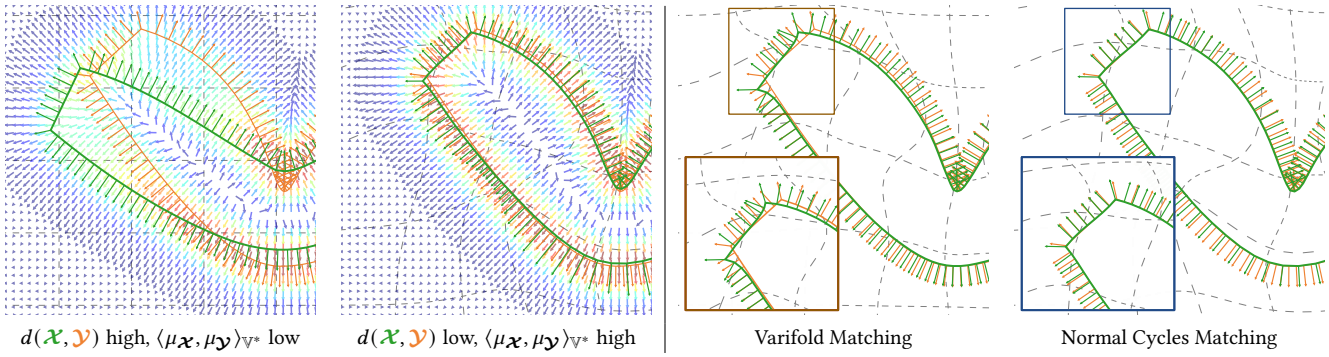

Fig. 2. Matching with Normal Cycles. Left: Currents Field. *The currents field, $\vec{\mathbf{v}}(\mathbf{x})$, is illustrated for the cross terms $\langle \mu_{\mathcal{X}}, \mu_{\mathcal{Y}} \rangle_{\mathbb{V}^*}$ for two different shapes to illustrate the mechanism of the dual norm as a matching metric (this field does not need to be evaluated when using the metric).* ***Right: Varifold vs Normal Cycles.*** *The source in green is matched to the target in orange using a varifold (left) and normal cycles (right) metric; the curvature sensitivity of the Normal Cycles matches the corners correctly unlike the Varifold.*

dual norm

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

implicitly, this employs the optimal vector field that aligns best with the normals over the whole surface. Using the norm, we can define a metric between two shapes as

$$\begin{aligned} d(\mathcal{X},\mathcal{Y}) &:= \|\mu_{\mathcal{X}} - \mu_{\mathcal{Y}}\|^2_{\mathbb{V}^*} = \langle \mu_{\mathcal{X}} - \mu_{\mathcal{Y}}, \mu_{\mathcal{X}} - \mu_{\mathcal{Y}} \rangle_{\mathbb{V}^*} \\ &= \langle \mu_{\mathcal{X}}, \mu_{\mathcal{X}} \rangle_{\mathbb{V}^*} - 2\langle \mu_{\mathcal{X}}, \mu_{\mathcal{Y}} \rangle_{\mathbb{V}^*} + \langle \mu_{\mathcal{Y}}, \mu_{\mathcal{Y}} \rangle_{\mathbb{V}^*} . \end{aligned} \tag{3}$$

Importantly, the RKHS structure allows us to evaluate this norm in closed form without needing to evaluate the field; each of the inner products $\langle \mu_{\mathcal{X}}, \mu_{\mathcal{Y}} \rangle_{\mathbb{V}^*}$ are found as

$$\int_{\mathcal{X}} \int_{\mathcal{Y}} \kappa(\mathbf{x},\mathbf{y}) \langle \hat{\mathbf{n}}_{\mathcal{X}}(\mathbf{x}), \hat{\mathbf{n}}_{\mathcal{Y}}(\mathbf{y}) \rangle \, dS_{\mathcal{X}}(\mathbf{x}) \, dS_{\mathcal{Y}}(\mathbf{y}) . \tag{4}$$

Figure 2 (left) illustrates the (implicit) vector field for the cross term $\langle \mu_{\mathcal{X}}, \mu_{\mathcal{Y}} \rangle_{\mathbb{V}^*}$ when matching two shapes; we can see that the matching metric is high when the shapes do not match as it is not possible to create a *smooth* field that is optimal for both shapes unless they are the same. Varifold matching uses this construction but uses a more general kernel $\kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\mathcal{X}}, \hat{\mathbf{n}}_{\mathcal{Y}})$, defined over the normal vectors, instead of the (Euclidean) inner product $\langle \hat{\mathbf{n}}_{\mathcal{X}}(\mathbf{x}), \hat{\mathbf{n}}_{\mathcal{Y}}(\mathbf{y}) \rangle$, in combination with the spatial kernel $\kappa_{\mathbf{x}}(\mathbf{x},\mathbf{x}')$. The inner product in Eq. (4) is replaced with

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

3.2 Normal Cycles

Figure 2 demonstrates a limitation of the Varifolds measure when dealing with areas of changing curvature; this is a limitation arising from employing a fixed kernel $\kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\mathcal{X}}, \hat{\mathbf{n}}_{\mathcal{Y}})$ for the normals. To overcome this limitation, Normal Cycles extend the currents approach of Eq. (1) to consider the *current over both the spatial and normal locations* and integrate this over a higher order object. The Normal Cycles inner product $\langle N_{\mathcal{X}}, N_{\mathcal{Y}} \rangle_{\mathbb{W}^*}$ is

$$\int_{N_{\mathcal{X}}} \int_{N_{\mathcal{Y}}} \kappa_{\mathbf{w}}(\mathbf{w}_{\mathbf{x}}, \mathbf{w}_{\mathbf{y}}) \langle \boldsymbol{\tau}_{\mathbf{x}}, \boldsymbol{\tau}_{\mathbf{y}} \rangle \, d\mathcal{H}^2(\mathbf{w}_{\mathbf{x}}) \, d\mathcal{H}^2(\mathbf{w}_{\mathbf{y}}) . \tag{6}$$

Comparing to the standard currents formulation, we have $\mathbf{x} \rightarrow \mathbf{w}_{\mathbf{x}}$, where $\mathbf{w}_{\mathbf{x}} := (\mathbf{x}, \hat{\mathbf{n}}_{\mathcal{X}}(\mathbf{x})) \in \mathbb{R}^3 \times \mathbb{S}^2$ combines both the spatial location and the associated normal ($\mathbb{S}^{d-1} := \{\hat{\mathbf{n}} \in \mathbb{R}^d \mid \|\hat{\mathbf{n}}\| = 1\}$ is the unit sphere). The second term in Eq. (6) is the inner product between differential 2-forms $\boldsymbol{\tau}_{\mathbf{x}}$ and $\boldsymbol{\tau}_{\mathbf{y}}$, [Roussillon and Glaunès 2019]; these combine the derivatives of the two components of $\mathbf{w}$. The derivative of the spatial component is the normal (as with standard currents) and the second term is the derivative of the normal component (this term introduces the curvature sensitivity). Finally, the whole integration occurs over the "normal bundle" $\mathcal{N}_{\mathcal{X}}$; this is the set

$$\mathcal{N}_{\mathcal{X}} := \left\{ (\mathbf{x}, \hat{\mathbf{n}}_{\mathcal{X}}) \,\middle|\, \mathbf{x} \in \mathcal{X},\ \hat{\mathbf{n}}_{\mathcal{X}} \in \mathbb{S}^2,\ \mathbf{t} \in T_{\mathbf{x}}[\mathcal{X}],\ \mathbf{t} \cdot \hat{\mathbf{n}}_{\mathcal{X}} = 0 \right\} , \tag{7}$$

where the tuple comprises the points and all unit vectors that are orthogonal to the tangent space (plane) of the surface at that point. Integration is with respect to the appropriate Hausdorff measure, $d\mathcal{H}^2(\mathbf{w}_{\mathbf{x}})$; this is a generalisation of the surface area to the "area" of the normal bundle defined in the higher dimensional space of $\mathbf{w}$.

3.3 Illustration with Planar 2D Curve

Figure 3 provides an illustration of the Normal Cycle $\mathcal{N}_{\mathcal{X}}$ for a planar 2D curve $\mathcal{X}$ where the space considered will be $\mathbf{w}_{\mathbf{x}} := (\mathbf{x}, \hat{\mathbf{n}}_{\mathcal{X}}(\mathbf{x})) \in \mathbb{R}^2 \times \mathbb{S}^1$. We visualise it, in green, as the accumulation of all the pointwise normal bundles along the surface. For an open curve, the end points comprise a half sphere (semi-circle) as the end cap. The normal bundle, and hence the normal cycle, for the union of two surfaces is given by the union rule

$$\mathcal{N}[A \cup B] = \mathcal{N}[A] + \mathcal{N}[B] - \mathcal{N}[A \cap B] . \tag{8}$$

When we discretise the continuous curve into a series of segments, $\{[\mathbf{x}_i, \mathbf{x}_j]\}$, we can use the union rule to build up an approximation to the complete surface by summation. We partition the normal bundle into a cylindrical component along the segment and spherical components at the vertices. It is easiest to use a complete sphere at each vertex and subtract off the half-sphere component facing into the segment from the cylindrical component (see bottom of Fig. 3).

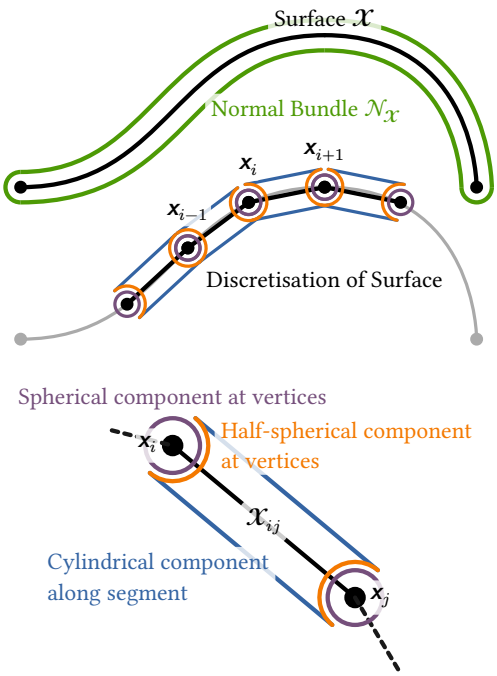

Fig. 3. Illustration of the discretisation of the normal bundle for a planar 2D curve. *The normal bundle for the full curve can be visualised as the normals (inwards and outwards) along the curve plus the half-sphere of normals at the end points. This is decomposed into a set of segments based on the discretisation of the curve. The union rule for combining the segments splits into contributions from a cylindrical component along the segment and sphere and half-sphere components at the vertices.*

Thus we can break down the normal cycle computation into

$$\mathcal{N}[\boldsymbol{\mathcal{X}}] = \sum_{i \in V_{\boldsymbol{\mathcal{X}}}} \mathcal{N}^{\text{vert}}[\boldsymbol{\mathcal{X}}_i] + \sum_{(i,j) \in E_{\boldsymbol{\mathcal{X}}}} \mathcal{N}^{\text{seg}}[\boldsymbol{\mathcal{X}}_{ij}]\,, \tag{9}$$

where the $\{\mathcal{N}^{\text{vert}}[\boldsymbol{\mathcal{X}}_i]\}$ vertex components are the spherical components at the vertices $V_{\boldsymbol{\mathcal{X}}}$, and the $\mathcal{N}^{\text{seg}}[\boldsymbol{\mathcal{X}}_{ij}]$ segment components are the cylindrical components minus the half-spheres over the edges (faces) $E_{\boldsymbol{\mathcal{X}}}$.

We can simplify the resulting computations by using a separable kernel

$$\kappa_{\mathbf{w}}(\mathbf{w}_{\mathbf{x}}, \mathbf{w}_{\mathbf{y}}) := \kappa_{\mathbf{x}}(\mathbf{x}, \mathbf{y})\, \kappa_{\mathbf{n}}(\hat{\mathbf{n}}_{\boldsymbol{\mathcal{X}}}(\mathbf{x}), \hat{\mathbf{n}}_{\boldsymbol{\mathcal{Y}}}(\mathbf{y}))\,, \tag{10}$$

with a constant kernel for the normal component, $\kappa_{\mathbf{n}}(\cdot,\cdot) = 1$. Then the dual inner product calculation, $\langle N_{\boldsymbol{\mathcal{X}}}, N_{\boldsymbol{\mathcal{X}}'} \rangle_{\mathbb{W}^*}$, for the planar curve reduces to

$$\frac{\pi^2}{4} \sum_{i \in V_{\boldsymbol{\mathcal{X}}}} \sum_{i' \in V_{\boldsymbol{\mathcal{X}}'}} \kappa_{\mathbf{x}}(\mathbf{x}_i, \mathbf{x}'_{i'}) \left\langle \sum_{(i,j) \in E_{\boldsymbol{\mathcal{X}}}} \frac{\mathbf{f}_{ij}}{\|\mathbf{f}_{ij}\|}, \sum_{(i',j') \in E_{\boldsymbol{\mathcal{X}}'}} \frac{\mathbf{f}'_{i'j'}}{\|\mathbf{f}'_{i'j'}\|} \right\rangle\,, \tag{11}$$

where $\mathbf{f}_{ij} := \mathbf{x}_j - \mathbf{x}_i$; please see Roussillon and Glaunès [2019] for the full derivation.

Curvature Sensitivity Intuition: The result for the planar curves gives us intuition as to the curvature sensitivity. If two points $\mathbf{x}_i$ and $\mathbf{x}'_{i'}$ are nearby, then $\kappa_{\mathbf{x}}(\mathbf{x}_i, \mathbf{x}'_{i'})$ will be high and we want the averaged, normalised $\mathbf{f}_{ij}$ to match (i.e. the inner product to be high). These vectors are the change in the surface geometry either side of the vertex, so will encode whether the surface has a high curvature or is flat. Thus the metric ensures that spatially close regions have similar curvatures.

3.4 Extension to 3D Surfaces

For a surface in 3D, the Normal Cycle is constructed similarly; segments are the triangulation and the normal bundle for each triangle comprises an additional planar component (over the faces) alongside a cylindrical component (along the edges) and a spherical component (at the vertices). The full calculation of Eq. (6) is quite involved and is derived by Roussillon and Glaunès [2019]. Due to the linear nature of the calculations, they can be reduced to weighted sums of inner products over the mesh vertices (as for the 2D curve case) as well as the edges for open surfaces.

Efficient Computation via Compression: In practice, we make use of a discrete approximation of the continuous integral in Eq. (6). Due to the RKHS structure, the approximations use kernel evaluations with quadratic complexity. A key feature of our method is using recent Normal Cycle compression techniques to produce sparse surface evaluations that closely approximate true surfaces, yielding substantial computational savings [Paul et al. 2025]. This allows us to match to high resolution target meshes without incurring the quadratic computational expense. The algorithmic details of the computation and compression of the NC metric for 3D surfaces are provided in Section 4.2 and we provide illustrative code in Appendix A.

4 METHOD

4.1 Overview and Background

Notation: We match a source shape $\boldsymbol{\mathcal{X}} = \{V_{\boldsymbol{\mathcal{X}}}, N_{\boldsymbol{\mathcal{X}}}, E_{\boldsymbol{\mathcal{X}}}, dS_{\boldsymbol{\mathcal{X}}}\}$ (comprising $i = 1, \ldots, I_{\boldsymbol{\mathcal{X}}}$ discrete vertices, normals, triangulation edges and elemental surface area respectively) to a target $\boldsymbol{\mathcal{Y}}$; for non-watertight surfaces, we define $\partial\boldsymbol{\mathcal{X}}$ as the boundary (vertices and edges). For *only the source shape*, we are also given a simple internal skeleton $\mathbb{S}_{\boldsymbol{\mathcal{X}}} = \{\mathbf{b}_j, e_k\}$ of joints and edges (bones) connected in a directed acyclic graph. We solve for the target skeleton jointly with the deformation and correspondence.

Diffeomorphic Deformation: 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 according to an Ordinary Differential Equation (ODE). Since continuous differentiable vector fields yield unique ODE solutions, streamlines cannot cross, which guarantees that the resulting transformation is smooth, invertible, and topology-preserving. The surface therefore evolves continuously under the flow, independent of mesh resolution or parameterisation.

The deformation maps each surface point $\mathbf{x}$ from its initial position $\mathbf{x}(0) \in V_{\boldsymbol{\mathcal{X}}}$ as a function of time, $\mathbf{x}(t)$. Our aim is that at a fixed time, $t = T$, $\mathbf{x}(T)$ lies on the target surface, $\boldsymbol{\mathcal{Y}}$, where this evolution governed by the initial value ODE:

$$\frac{d}{dt}\mathbf{x}(t) = \vec{\mathbf{f}}(\mathbf{x}(t), t), \quad \text{s.t.} \quad \mathbf{x}(t=0) \in V_{\boldsymbol{\mathcal{X}}}\,. \tag{12}$$

The time varying vector flow field $\vec{\mathbf{f}}_\theta(\mathbf{x},t) : \mathbb{R}^{3\times 1} \to \mathbb{R}^3$ is modeled via a neural network, also termed a "NeuralODE" [Chen et al. 2018], with parameters θ. This field is numerically integrated over "time" from the source at $t=0$ to match the target at $t=T$:

$$\mathbf{x}^{(T)} = \mathbf{x}^{(0)} + \int_0^T \vec{\mathbf{f}}_\theta(\mathbf{x}(t),t)\,dt =: \text{ODESolve}\left(\mathbf{f}_\theta, \mathbf{x}_0, T\right), \quad (13)$$

where the ODESolve function as the result of a numerical solver that approximates the integral to a given tolerance.

For shapes where we wish to enforce the (hard) constraint of *volume preservation*, we parameterise the flow field via the curl operator, $\vec{\mathbf{f}}_\theta(\mathbf{x},t) := \nabla \times \vec{\mathbf{a}}_\theta(\mathbf{x},t)$, to ensure a divergence-free flow. We guarantee continuity (smoothness) of the field to ensure that differential properties are easy to propagate; we can evaluate, for example, the surface normals or change in area at any point inside or on the surface of the shape (e.g. Fig. 4).

4.2 Normal Cycle Matching

While ARC-Flow uses a Varifold formulation for matching (outlined in Section 3.1), we enhance matching quality by adopting the Normal Cycles metric, as discussed in Section 3.2. Calculating the NC metric requires evaluation of the NC inner product, Eq. (6); this is more involved than the Varifold metric due to the increased dimensionality of the operations as we must also include discrete approximations of the derivatives of normals to incorporate curvature information. We wish to evaluate the metric $\mathcal{L}_{\text{nc}}(\theta) := d_{\text{nc}}(\boldsymbol{\mathcal{X}}^{(T)}, \boldsymbol{\mathcal{Y}})$ between the deformed source shape $\boldsymbol{\mathcal{X}}^{(T)}$ and the fixed target $\boldsymbol{\mathcal{Y}}$.

Discrete Sum: The discrete approximation is formed in two stages. Firstly, "representer" vectors $\{\boldsymbol{\alpha}_i, \boldsymbol{\gamma}_e\}$ are calculated for each i^{th} surface location and e^{th} boundary edge. Secondly, the inner product may be evaluated as

$$\begin{aligned}\langle \mathcal{N}_{\boldsymbol{\mathcal{X}}}, \mathcal{N}_{\boldsymbol{\mathcal{Y}}} \rangle_{\mathbb{W}^*} \approx \frac{\pi^2}{4} \sum_{e=1}^{|E_{\boldsymbol{\mathcal{X}}}|} \sum_{e'=1}^{|E_{\boldsymbol{\mathcal{Y}}}|} \kappa_{\mathbf{x}}(\mathbf{c}_e^{(\boldsymbol{\mathcal{X}})}, \mathbf{c}_{e'}^{(\boldsymbol{\mathcal{Y}})}) \langle \boldsymbol{\gamma}_e^{(\boldsymbol{\mathcal{X}})}, \boldsymbol{\gamma}_{e'}^{(\boldsymbol{\mathcal{Y}})} \rangle \\ + \frac{\pi^2}{4} \sum_{i=1}^{I_{\boldsymbol{\mathcal{X}}}} \sum_{i'=1}^{I_{\boldsymbol{\mathcal{Y}}}} \kappa_{\mathbf{x}}(\mathbf{x}_i, \mathbf{y}_{i'}) \langle \boldsymbol{\alpha}_i^{(\boldsymbol{\mathcal{X}})}, \boldsymbol{\alpha}_{i'}^{(\boldsymbol{\mathcal{Y}})} \rangle ,\end{aligned} \quad (14)$$

where $\mathbf{c}_e := \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_{i'})$ are centres for edges $(i,i')_e \in E_{\boldsymbol{\mathcal{X}}}$. Terms in $\{\boldsymbol{\alpha}\}$ disappear for watertight surfaces with no boundary. The kernel $\kappa_{\mathbf{x}}(\cdot,\cdot)$ is *spatial* with the normal/curvature information embedded in $\{\boldsymbol{\alpha}, \boldsymbol{\gamma}\}$. Algorithm 1 describes the calculation of $\{\boldsymbol{\alpha}, \boldsymbol{\gamma}\}$; the mathematical foundation is found in Paul et al. [2025].

Compression: As with the Varifold metric, the computation of the inner product, Eq. (14), is quadratic in nature as the product of the number of points in each shape, $O(I_{\boldsymbol{\mathcal{X}}} I_{\boldsymbol{\mathcal{Y}}})$. A key advantage of our approach is to lower the cost by reducing the number of points that need to be evaluated whilst retaining the quality of the output; this is termed compression and proceeds via a sparse Nyström approximation that offers mathematical guarantees to its performance [Paul et al. 2025]. The algorithm employs a Ridge Leverage Score (RLS) approach to assess a surface point's importance, which is used to create a Nyström approximation for the 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.

ALGORITHM 1: Forming Normal Cycle Vectors

Input: Shape $\mathcal{X} = \{V_{\boldsymbol{\mathcal{X}}}, N_{\boldsymbol{\mathcal{X}}}, E_{\boldsymbol{\mathcal{X}}}, dS_{\boldsymbol{\mathcal{X}}}\}$
Vertices $\mathbf{x}_i \in V_{\boldsymbol{\mathcal{X}}}$, Normals $\mathbf{n}_i \in N_{\boldsymbol{\mathcal{X}}}$
Edges $(i,i')_e \in E_{\boldsymbol{\mathcal{X}}}$
Spatial Kernel $\kappa_{\mathbf{x}}(\cdot,\cdot)$
Oriented edges $\mathbf{f}_e := \mathbf{x}_{i'} - \mathbf{x}_i \;\forall\; (i,i')_e \in E_{\boldsymbol{\mathcal{X}}}$
Edge centres $\mathbf{c}_e^{(\boldsymbol{\mathcal{X}})} := \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_{i'}) \;\forall\; (i,i')_e \in E_{\boldsymbol{\mathcal{X}}}$

Output: Representation vectors $\{\boldsymbol{\alpha}_i^{(\boldsymbol{\mathcal{X}})}, \boldsymbol{\gamma}_e^{(\boldsymbol{\mathcal{X}})}\}$

1 ▸ **Define Mapping** $B(\cdot,\cdot) : \mathbb{R}^6 \times \mathbb{R}^6 \longrightarrow \mathbb{R}^{15}$ **as:**

$$B(a,b) = (a_i b_j - a_j b_i)_{1\le i<i',\, i'=2,\ldots,6}$$

and

$$\tilde{b}_1 := B\left([\mathbf{0}^\top, \mathbf{e}_2^\top]^\top, [\mathbf{0}^\top, \mathbf{e}_3^\top]^\top\right)$$
$$\tilde{b}_2 := B\left([\mathbf{0}^\top, \mathbf{e}_1^\top]^\top, [\mathbf{0}^\top, \mathbf{e}_3^\top]^\top\right)$$
$$\tilde{b}_3 := B\left([\mathbf{0}^\top, \mathbf{e}_1^\top]^\top, [\mathbf{0}^\top, \mathbf{e}_2^\top]^\top\right)$$

where $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ are standard ordered basis of $\mathbb{R}^3$.

2 ▸ **Find** $\{\boldsymbol{\gamma}_e^{(\boldsymbol{\mathcal{X}})}\}$ **for each edge:**
for $e = 1$ **to** $|E_{\boldsymbol{\mathcal{X}}}|$ **do**

$$\boldsymbol{\gamma}_e^{(\boldsymbol{\mathcal{X}})} := -\sum_{i\in(i,i')_e} B\left(\left[\frac{\mathbf{f}_j^\top}{\|\mathbf{f}_j\|}, \mathbf{0}^\top\right]^\top, \left[\mathbf{0}^\top, \mathbf{n}_i\right]^\top\right)$$

where $\mathbf{n}_i \times \mathbf{f}_j$ is oriented inward for the triangulation.

3 ▸ **Find** $\{\boldsymbol{\alpha}_i^{(\boldsymbol{\mathcal{X}})}\}$ **for each vertex:**
for $i = 1$ **to** $I_{\boldsymbol{\mathcal{X}}}$ **do**

$$\boldsymbol{\alpha}_i^{(\boldsymbol{\mathcal{X}})} := \begin{cases} -\sum_{\mathbf{f}_e : i\in(i,i')_e} \sum_{d=1}^{3} \frac{[\mathbf{f}_e]_d}{\|\mathbf{f}_e\|} \tilde{b}_d \\ \mathbf{0} \quad \text{if } i \text{ not on the boundary} \end{cases}$$

where $[\mathbf{f}_e]_d$ is the component of boundary edge $\mathbf{f}_e \in \partial\boldsymbol{\mathcal{X}}$ attached to $\mathbf{x}_i$ reoriented to point outwards.

4 **return** $\{\boldsymbol{\alpha}_i^{(\boldsymbol{\mathcal{X}})}, \boldsymbol{\gamma}_e^{(\boldsymbol{\mathcal{X}})}\}$

Algorithm 2 describes how to perform the compression; application to the target shape, results in a compressed approximation $\boldsymbol{\mathcal{Y}}^c = \{V_{\boldsymbol{\mathcal{Y}}^c}, \boldsymbol{\beta}\}$ containing a subset of $I_{\boldsymbol{\mathcal{Y}}^c} := |V_{\boldsymbol{\mathcal{Y}}^c}|$ of the original vertices and a corresponding set of weights $\boldsymbol{\beta}$. The NC inner product, Eq. (14), for a compressed target becomes

$$\langle \mathcal{N}_{\boldsymbol{\mathcal{X}}}, \mathcal{N}_{\boldsymbol{\mathcal{Y}}^c} \rangle_{\mathbb{W}^*} \approx \sum_{i=1}^{I_{\boldsymbol{\mathcal{X}}}} \sum_{i'=1}^{I_{\boldsymbol{\mathcal{Y}}^c}} \kappa_{\mathbf{x}}(\mathbf{x}_i, \mathbf{y}_{i'}^c) \langle \boldsymbol{\alpha}_i^{(\boldsymbol{\mathcal{X}})}, \boldsymbol{\beta}_{i'} \rangle , \quad (15)$$

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

$$\mathcal{L}_{\text{nc}}(\theta) := d_{\text{nc}}(\boldsymbol{\mathcal{X}}^{(T)}, \boldsymbol{\mathcal{Y}}^c) \quad (16)$$
$$= \langle \mathcal{N}_{\boldsymbol{\mathcal{X}}^{(T)}}, \mathcal{N}_{\boldsymbol{\mathcal{X}}^{(T)}} \rangle_{\mathbb{W}^*} + 2\langle \mathcal{N}_{\boldsymbol{\mathcal{X}}^{(T)}}, \mathcal{N}_{\boldsymbol{\mathcal{Y}}^c} \rangle_{\mathbb{W}^*} + C , \quad (17)$$

ALGORITHM 2: Sparse Nyström Approx of Normal Cycles

Input: Shape $\mathcal{Y} = \{V_{\mathcal{Y}}, N_{\mathcal{Y}}, E_{\mathcal{Y}}, dS_{\mathcal{Y}}\}$
Vertices $\mathbf{x}_i \in V_{\mathcal{Y}}$, Normals $\mathbf{n}_i \in N_{\mathcal{Y}}$
Edges $(i, i')_e \in E_{\mathcal{Y}}$
Spatial Kernel $\kappa_{\mathbf{x}}(\cdot,\cdot)$
n: Number of Samples in $\mathcal{Y}$ $(= I_{\mathcal{Y}})$
m: Desired Compressed Size $(= I_{\mathcal{Y}^c})$
λ: Regularisation parameter
$\kappa_{\mathbf{x}}(\cdot,\cdot)$: Positional Kernel

Output: $\mathcal{Y}^c$: Compressed Representors & Weights : $\{V_{\mathcal{Y}^c}, \boldsymbol{\beta}\}$

1
► **Compute leverage scores:**

$b_s \leftarrow \lfloor \sqrt{n} \rfloor$
$b \leftarrow \lfloor n/s \rfloor$
Split $\mathcal{Y}$ into b random batches $\{\mathcal{Y}_1, \ldots, \mathcal{Y}_b\}$ of size b_s
for $j = 1$ **to** b **do**
  for $i = 1$ **to** b_s **do**
    $\Lambda_i \leftarrow \mathbf{K}_{i,\mathcal{Y}_j}(\mathbf{K}_{i,\mathcal{Y}_j} + \lambda \mathbf{I})^{-1}$
    ► $\mathbf{K}_{i,\mathcal{Y}_j} := \sum_{i' \in \mathcal{Y}_j} K_p(y_j, y_i')\, \boldsymbol{\alpha}_i^{(\mathcal{Y})} \in \mathbb{R}^{15}$

2
► **Draw weighted samples:**

Define sampling distribution:
Let $X_i \sim p(W)$ where $p(X_i = s_j) = \frac{\Lambda_j}{\sum_{k=1}^{n} \Lambda_k}$
$\mathcal{C} \leftarrow \{\ \}$
for $i = 1$ **to** m **do**
  $\{\mathbf{x}_s\} \leftarrow$ Draw a sample from $\mathcal{Y}$ according to $p(W)$;
  Add $\{\mathbf{x}_s\}$ to $\mathcal{C}$
$V_{\mathcal{Y}^c} \leftarrow \mathcal{C}$

3
► **Calculate sample weights:**

► $\mathbf{K}_{\mathcal{C},\mathcal{C}} = \sum_{i=1}^{I_{\mathcal{C}}} \sum_{i'=1}^{I_{\mathcal{C}}} \kappa_{\mathbf{x}}(\mathbf{c}_i, \mathbf{c}_{i'})$
► $\mathrm{Y} = \sum_{i=1}^{I_{\mathcal{C}}} \sum_{i'=1}^{I_{\mathcal{Y}}} \kappa_{\mathbf{x}}(\mathbf{c}_i, \mathbf{x}_{i'})$
$\boldsymbol{\beta} \leftarrow \mathbf{K}_{\mathcal{C},\mathcal{C}}^{-1} \mathrm{Y} \in \mathbb{R}^{m \times k}$

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

where $C := \langle \mathcal{N}_{\mathcal{Y}^c}, \mathcal{N}_{\mathcal{Y}^c} \rangle_{\mathbb{W}^*}$ is constant wrt θ and need not be evaluated for the purpose of optimisation.

Efficiency: The evaluation of the NC metric is more expensive (although with the same complexity) than for Varifolds, but the advantages are in the improved fine detail of surface quality in the matching. We begin by using the Varifold metric to obtain a coarse solution for the neural flow field, then transition to the Normal Cycle metric for fine detail refinement. Overall, we found, empirically, that this reduces the total matching time as fewer iterations of NC matching are required to achieve superior quality correspondences.

4.3 Skeleton Interpolation

We assume an articulated body has an internal skeletal structure and are provided a simple skeleton $\mathbb{S}_{\mathcal{X}} = \{\mathbf{b}_j, e_k\}$, of vertices and edges, for the source shape $\mathcal{X}$. This should deform in a locally rigid manner whilst the surrounding soft tissue and surface deform non-rigidly. The local rigidity of the field can be enforced by taking samples along the original source skeleton and penalising any locally non-rigid motion (i.e. we estimate where the bones should move to and ensure that the deformation field is consistent. This is shown in Fig. 4 where we illustrate failure to constrain the skeleton. We introduce three key improvements to the underlying model leading to more efficient optimisation and interpolated poses that improve both numerical properties and physical plausibility.

Improved Skeleton Model: We solve for the skeleton of the final pose via estimating a global translation $\tilde{\mathbf{s}} \in \mathbb{R}^3$ and a set of angles for the joints between each bone. ARC-Flow makes use of a standard forward kinematics skeleton and then performs per-bone interpolation. In contrast, we adopt a unifying representation, via Dual Quaternions, and perform more natural interpolation with consistent forward dynamics and improved numerical stability.

At the same time, ARC-Flow encodes the local rigidity of the skeleton as a soft constraint with a single hyperparameter; this parameter is particularly difficult to tune as it trades off over the whole skeleton; even with complex tuning and optimisation schedules, there are often misalignments in the final skeleton. We use a constrained optimisation approach that automatically tunes per-bone hyperparameters to ensure that the skeleton recovered is consistent to within a small tolerance. This improves accuracy as well as removing complex scheduling and the need to tune the skeleton hyperparameter.

Dual-Quaternion Representation: Even with fixed bone lengths, the translation of a joint is not independent of its rotation. The motion of a rigid body is naturally a screw motion; a rotation about an axis combined with a translation along that axis. Dual-quaternions (DQ) Kavan et al. [2007] provide a unified representation for rigid-body motion. Given a quaternion rotation $\mathbf{q} \in \mathbb{Q}$ and a translation vector $\mathbf{s} \in \mathbb{R}^3$, the corresponding dual-quaternion $\mathbf{Q}(\mathbf{q}, \mathbf{s})$ is

$$\mathbf{Q}(\mathbf{q}, \mathbf{s}) := \mathbf{q}_r + \mathbf{q}_d \in \mathbb{DQ}\,, \tag{18}$$
$$\mathbf{q}_r := \mathbf{q}\,,\ \mathbf{q}_d := \frac{1}{2}(\mathbf{q}_r \mathbf{q}_s)\,,\ \mathbf{q}_s := (0, s_x, s_y, s_z)\,.$$

We encode the forward kinematics of the skeleton using DQs for each bone; $\{\mathbf{Q}_k^{(0)}\}$ encode the initial source skeleton (provided) at $t = 0$ and $\{\tilde{\mathbf{Q}}_k^{(T)}\}$ the final skeleton, at $t = T$ to be solved for. This represents motion as a combined rotation and translation along an axis (screw motion) in a single compact object. Consistent interpolation over SE(3) ensures smooth and mathematically valid rigid body motion by maintaining proper coupling between rotation and translation, unlike linear interpolation which can create unnatural paths despite reaching correct endpoints. For animated skeletons, maintaining the natural screw motion during interpolation often produces smoother and more physically plausible movements. In addition, DQs tend to be more numerically stable and compact.

Dual-Quaternion Interpolation: We use "SCLERP" Kenwright [2023], a dual quaternion interpolation method that smoothly blends rigid body transformations while preserving both rotation and translation consistency. The motion between $\mathbf{Q}_A$, at $t = 0$, and $\mathbf{Q}_B$, at

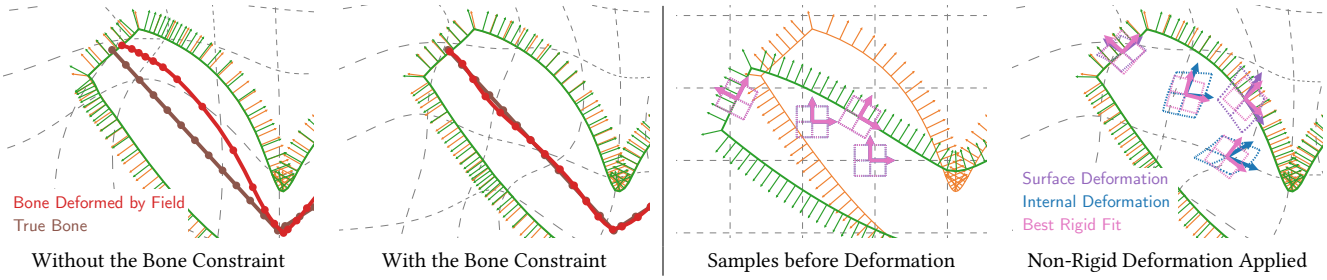

Fig. 4. Skeleton and Soft Tissue. Left: Bone Constraint*. Without the bone constraint, the deformation field (matching the estimate in green to the target in orange) may warp the internal skeleton (shown in red) resulting in distorted matching.* ***Right: Soft Tissue Priors.*** *The differential formulation makes it easy to evaluate non-rigid distortion (e.g. change of area) of surface samples (shown in purple) or internal tissue (shown in blue); we can penalise this distortion against the best local rigid fit (shown in pink).*

$t = T$ is given as

$$\text{SCLERP}(t; \mathbf{Q}_A, \mathbf{Q}_B) := \mathbf{Q}_A (\mathbf{Q}_B^* \mathbf{Q}_B)^{t/T} \,. \tag{19}$$

Unlike linear blending, which can cause shearing, SCLERP preserves unit norms for smooth, natural transitions. It promotes physically plausible motion by interpolating along a geodesic in dual quaternion space; see Appendix C for details.

Alignment of Bone Model: Local rigidity of the internal skeleton should ideally be enforced as a *hard constraint*. We therefore adopt a constrained optimization framework (detailed in Section 4.5) that guarantees bone samples remain a specified tolerance, ϵ_{bone}, of their locally rigid positions when deformed through the flow field.

We take samples $\{\mathbf{p}_k^{(0)}\}$ on the bones of the source skeleton (details in Section 4.6) and push them through the deformation field to find their location $\{\mathbf{x}_{\mathbf{p}_k}(t)\}$ sampled at time $t \in [0, T]$. We compare this to locations resulting from the locally rigid interpolation for the bone at time t

$$\begin{aligned} \mathbf{p}_k(t) &:= \mathbf{Q}_k^*(t)\, \mathbf{p}_k^{(0)}\, \mathbf{Q}_k(t) \,, \\ \mathbf{Q}_k(t) &:= \text{SCLERP}\left(t; \mathbf{Q}_k^{(0)}, \tilde{\mathbf{Q}}_k^{(T)}\right) \,. \end{aligned} \tag{20}$$

We use this to define an error metric for each bone, $g_k(\cdot)$, that is zero when the samples deformed by the field match the locally rigid interpolation

$$g_k\left(\theta, \tilde{\mathbf{s}}, \{\tilde{\mathbf{Q}}_k^{(T)}\}\right) := \mathbb{E}_{t, \mathbf{p}_k^{(0)}}\left[\, \|\mathbf{x}_{\mathbf{p}_k}(t) - \mathbf{p}_k(t)\|^2 \,\right], \tag{21}$$

where the expectation denotes samples distributed across time and spatially over the bone.

4.4 Soft Tissue Sampling

ARC-Flow estimated regions of soft tissue from cylindrical shells surrounding the pseudo bone cylinders. However, this results in uneven sampling density, with a bias towards groups of small bones e.g. fingers of a hand, and does not extend to the full volume due to its relation to the bone lengths.

Uniform Volumetric Sampling: We address this limitation with a preprocessing step that generates a uniformly distributed set of interior sample points. We employ the fast tetrahedral meshing algorithm of Hu et al. [2020] to fill the volume enclosed by the source surface. Any tetrahedra whose volume exceeds a predefined threshold are subdivided so that all elements have approximately equal volume; their centroids are then used as our sampling locations. This robust method requires only a few seconds and is run on the source shape once to generate our set of soft tissue sample locations. Figure 5 illustrates these steps. During training, we sample uniformly distributed soft tissue locations by randomly selecting tetrahedral centroids, and we penalise surface ($\mathcal{L}_{\text{surf}}$) and soft tissue ($\mathcal{L}_{\text{soft}}$) distortion using the same soft constraints as ARC-Flow.

4.5 Constrained Optimisation

ARC-Flow trades off four loss terms (shape matching, skeleton, soft tissue, surface distortion) in a weighted sum; this involves hyperparameters that are difficult to balance and may vary according to schedules or different datasets.

We replace this with a constrained optimisation where we guarantee that the hard skeletal constraints are met to a given tolerance

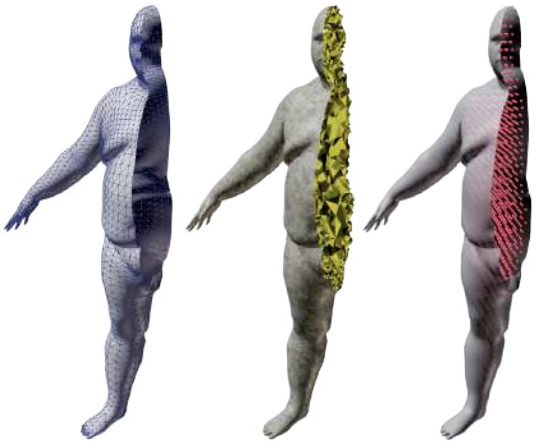

Fig. 5. Soft Tissue Sampling: *First, we use a fast tetrahedral meshing technique to fill the volume of our source mesh, then generate uniform random tissue samples by randomly sampling from the centroids of the tetrahedra.*

with the remaining losses reduced as much as possible. We place a tolerance $\varepsilon_{\text{skel}}$ on the each bone of the skeleton constraint (where $g_k(\cdot) = 0$ if the k^{th} bone deforms rigidly). So we optimise a reconstruction loss, $\mathcal{L}_{\text{match}}$ such that the constraints on the bone rigidity are satisfied:

$$\mathcal{L}_{\text{match}}(\Omega) \coloneqq \mathcal{L}_{\text{nc}}(\theta) + \mu_{\text{surf}}\,\mathcal{L}_{\text{surf}}(\Omega) + \mu_{\text{soft}}\,\mathcal{L}_{\text{soft}}(\Omega) \qquad (22)$$
$$\text{s.t.} \quad g_k(\Omega) \le \varepsilon_{\text{skel}},\ \forall k\,,$$

where $\mathcal{L}_{\text{nc}}(\theta)$ is the Normal Cycles metric between the deformed shape and the target from Eq. (16), and μ_{surf} and μ_{soft} are positive weights to trade off between surface and internal soft tissue distortion losses from Section 4.4. We jointly optimise with respect to $\Omega \coloneqq (\theta, \phi, \tilde{\mathbf{s}}, \{\tilde{\mathbf{Q}}_k^{(T)}\})$, comprising: the parameters of the Flow-Net, θ, defining the time-varying vector flow field; the GSO-Net, ϕ, predicting rotation of conformal samples; and the translation, $\tilde{\mathbf{s}}$, and dual quaternions, $\{\tilde{\mathbf{Q}}_k^{(T)}\}$, of the forward kinematic skeleton. The constrained optimisation is transformed to an unconstrained optimisation via the Modified Differential Method of Multipliers (MDMM) [Platt and Barr 1987]. The modified Lagrangian is

$$\mathcal{L}_{\text{mdmm}} \coloneqq \mathcal{L}_{\text{match}}(\Omega) + \boldsymbol{\lambda}^\top(\mathbf{G}(\Omega) - \boldsymbol{\varepsilon}) + \delta\,\|\mathbf{G}(\Omega) - \boldsymbol{\varepsilon}\|^2\,, \qquad (23)$$

$$\mathbf{G}(\Omega) \coloneqq [\dots, g_k(\Omega), \dots]^\top \ge \mathbf{0}\,, \qquad (24)$$

where $\boldsymbol{\lambda} \ge \mathbf{0}$ is a vector of Lagrange multipliers (one for each bone) for the constraint $\mathbf{G}(\Omega) \le \boldsymbol{\varepsilon}$, and $\delta > 0$ is some damping hyperparameter. For improved convergence properties, we optimise a modified version of the objective

$$\mathcal{L}_{\text{obj}}(\Omega, \boldsymbol{\lambda}) \coloneqq \mathcal{L}_{\text{dist}}(\Omega) + \boldsymbol{\lambda}^\top \boldsymbol{\Gamma}(\Omega) + \delta\,\boldsymbol{\Gamma}(\Omega)\,, \qquad (25)$$

$$\boldsymbol{\Gamma}(\Omega) \coloneqq \max\left(\boldsymbol{\varepsilon}, \min(\mathbf{0}, \mathbf{G}(\Omega))\right) - \mathbf{G}(\Omega)\,. \qquad (26)$$

We seek a saddle point of Eq. (25) via gradient descent wrt the model parameters Ω and gradient *ascent* wrt the dual variables $\boldsymbol{\lambda}$:

$$\hat{\Omega} \coloneqq \arg\min_{\Omega} \max_{\boldsymbol{\lambda}}\ \mathcal{L}_{\text{obj}}(\Omega, \boldsymbol{\lambda})\,, \qquad (27)$$

subject to the constraint that $\boldsymbol{\lambda} \ge 0$; we guarantee this by thresholding during the optimisation. The automatic balancing provided by the dual parameters (i.e. dynamically changing the per bone weight to ensure the constraints are met) means that we can avoid the complex scheduling present in ARC-Flow; we achieve superior results whilst improving the robustness of the optimisation.

4.6 Implementation Details

In this section, we provide implementation details. Our implementation uses the JAX framework [Bradbury et al. 2018] and Equinox neural network library [Kidger and Garcia 2021]. We follow Hartshorne et al. [2025] in leveraging an efficient kernel ops library, probabilistic ODE solver and vector optimiser, namely Keops [Charlier et al. 2021], ProbDiffEq [Krämer 2024] and VectorAdam [Ling et al. 2022] respectively.

Flow-Net Architecture: The MLP used for the Flow-Net NODE comprises a simple 5 layer MLP with layer widths of 256 and use a GeLU activation function. The constrained optimisation permits superior results with a more simple architecture for the NeuralODE than ARC-Flow which relies on SIREN and FINER layers.

Skeleton Parameterization: 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 in the fingers, feet, and paws. This augmentation results in a more anatomically accurate representation, more closely mimicking real-life skeletal structures.

For the optimisation, we use 50 samples for each bone, $\{\alpha_m\}$, 2000 samples for the soft tissue, while 500 samples are used for the surface points, $\{x_i\}$. All of which are randomly resampled in every epoch. In general, we use a radius of 10% of the length of each bone to define the rigid region. Human fingers are relative to the rest of a human body long and narrow, thus for DFAUST dataset they require significantly smaller regions (1% of the length of the bone) to stay within the surface.

Training details: The objective of Eq. (25), is optimised using the VectorAdam [Ling et al. 2022] algorithm. We found further improvement by incorporating Cautious Optimization [Liang et al. 2024], a simple performance booster for any momentum-based optimizer, that accelerates the reduction of the loss function while maintaining Adam's Hamiltonian function and preserving the convergence guarantees established through Lyapunov analysis.

The ODE 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.

We initialize the system using the Varifold data term of Hartshorne et al. [2025] for 500 epochs, with a lengthscale of 0.5 for both the spatial and normal kernels. The result of which provides a rough alignment between the template and target shape. We use the same weighting terms for all experiments, whereby μ_{surf} is set to 5e3 and μ_{soft} to 1e1. The lengthscale of the Normal Cycle kernel is reduced in a coarse to fine manner every 2000 epochs; moving through 0.5, 0.25, 0.1.

5 RESULTS

Our problem lies at the intersection of physically plausible interpolation and accurate correspondence, requiring a careful balance of both. We evaluate quantitatively and qualitatively against a range of leading approaches including; current state-of-the-art ARC-Flow [Hartshorne et al. 2025]; the FM, training data based SMS [Cao et al. 2024] that also provides dense correspondence and interpolation; another Varifold approach, without correspondence, ESA [Hartman et al. 2023b]; a Hamiltonian approach for high quality interpolants, but requiring correspondence, HAM [Eisenberger and Cremers 2020]; and a divergence-free field approach, for volume preservation, DIV FREE [Eisenberger et al. 2019]. **Datasets**: We evaluate quantitatively on three standard datasets: DFAUST [Bogo et al. 2017], MANO [Romero et al. 2017], and SMAL [Zuffi et al. 2017]; SMAL samples are from a modified version of the dataset [Huang et al. 2021]. To standardise across methods, all examples are normalised to fit a length-one cube and we use K-medoids [Park and Jun 2009] to select a diverse range of poses from each dataset. We use an 80/20 split; as SMS requires training data, leaving 20 examples for testing. For DFAUST, we select random pairs of poses across multiple subjects, but ensure a pairing is of the same individual. We qualitatively evaluate the performance of our method

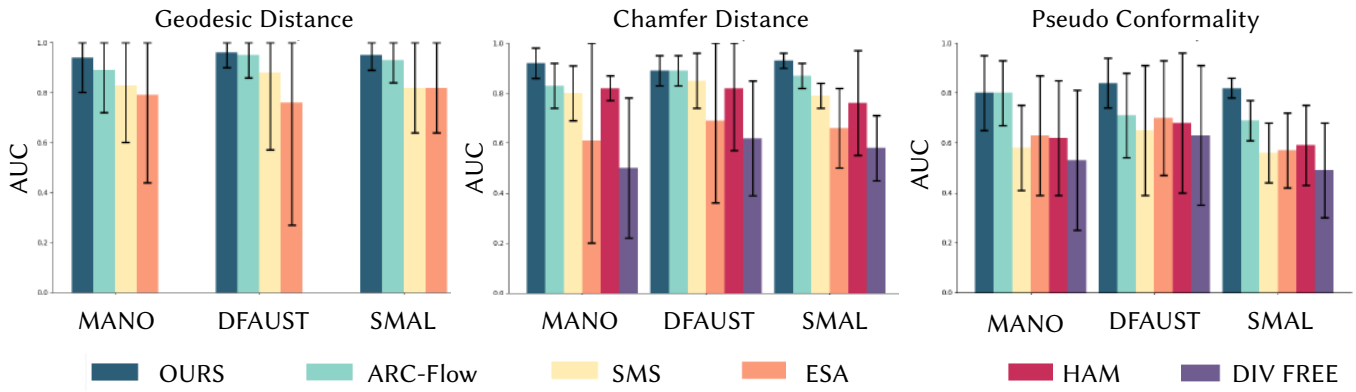

Fig. 6. Summary of Quantitive Analysis. *Our method improves across all metrics and also has narrower error bars indicating more consistent performance. Area Under the Curve (AUC) metrics were calculated with a threshold of 0.20, 0.10 and 0.15 for Geodesic, Chamfer and Conformal metrics respectively. The full data tables and detailed charts are provided in Appendix D.*

on topologically noisy data via the TOPKIDS dataset [Lähner et al. 2016] that contains synthetic shapes of children with topological merging. Such topological noise is historically challenging for shape matching methods. We also apply our method to the TOSCA dataset [Bronstein et al. 2008], demonstrating success when using using user-defined skeletons created in blender in a few minutes.

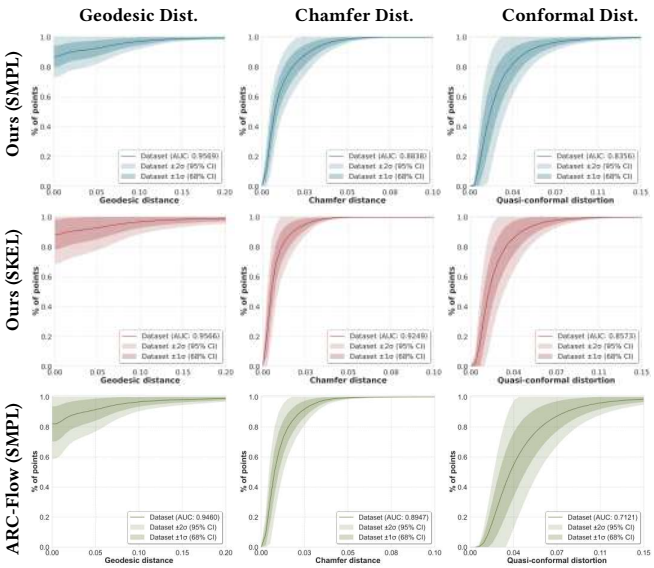

Fig. 7. Interpolation results on DFAUST with our method using SMPL [Romero et al. 2017] & SKEL [Keller et al. 2023] skeletons; vs ARC-Flow [Hartshorne et al. 2025] using SMPL. *Mean and confidence intervals for the three metrics are shown; our method improves across all metrics irrespective of the skeleton and also has narrower error bars indicating more consistent performance than ARC-Flow. Please zoom for details.*

5.1 Quantitative Results

We use three standard metrics to evaluate the performance (both as full curves with mean and confidence intervals, and via Area Under the Curve (AUC) values): we asses the point-to-point correspondence accuracy to ground truth via the *Geodesic Distance*; the accuracy of the overall shape reconstruction (predicted vs target) via the *Chamfer Distance*; and a measure of the quality of the local geometric properties of the interpolation on the recovered surface via the *Quasi-Conformal Distortion* Hormann and Greiner [2000].

Our approach demonstrates superior performance across all three metrics, as shown in Fig. 6. Further in-depth results may be found in Appendix D including numerical results in Table 1 and full performance curves for the three datasets in Figs. 21 to 23.

SMS has previously exhibited competitive performance against established pure matching methods such as ZoomOut [Melzi et al. 2019], while the provided results show that our method achieves superior correspondence accuracy versus SMS on comparable datasets.

Choice of Skeleton: Incorporating a simple source skeleton is a key aspect of our method. To assess its impact, we repeat the quantitative experiments comparing our approach using both SMPL [Loper et al. 2015] and SKEL [Keller et al. 2023] skeletons on the DFAUST versus ARC-Flow datasets. As shown in Fig. 7, the results demonstrate that our method consistently achieves high-quality outcomes irrespective of the chosen skeleton. Figure 14 shows a visual example of the two different skeletons with no discernible difference between the results.

5.2 Ablation Study

To assess the individual contributions of our proposed improvements, we conducted an ablation study systematically comparing the effects of adding each component: Normal Cycles (NC), Dual Quaternions (DQ), and MDMM-based constrained optimisation (CO) against the baseline ARC-FLOW method [Hartshorne et al. 2025]. Fig. 8 presents these comparisons, where VAR denotes the original

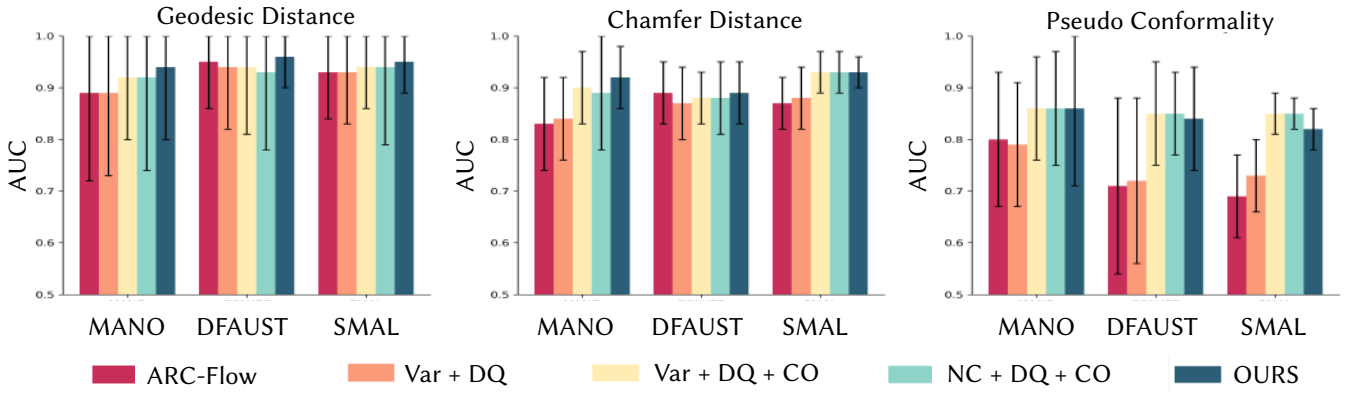

Fig. 8. Summary of Ablation Study Results. *Evaluation is performed using Area Under the Curve (AUC) metrics across all datasets, with thresholds of 0.20, 0.10, and 0.15 for Geodesic, Chamfer, and Conformal metrics respectively. The results highlight the effect of incrementally adding the different component of our method. The full data tables are provided in Table 1.*

Varifold metric from ARC-FLOW, while NC, DQ, and CO represent our proposed enhancements as described above.

Replacing single quaternions from ARC-Flow with DQs alone has little effect, but combining them with CO improves both fit quality and interpolation smoothness. Similarly, initialising the optimisation with Varifolds before refining with NCs (Full Method) consistently outperforms just using NCs. For large deformations, Varifolds' lower sensitivity to orientation allows coarse matching to guide the early optimisation before the vector field is well-formed.

5.3 Qualitative Results

In addition to our quantitative analysis, we present a number of qualitative examples that both that both validate the quantitative metrics, but also reveal nuances that the summary statistics alone don't always convey. Figure 9 shows results from the DFAUST, MANO and SMAL datasets of example pairs with the largest deformations (most challenging examples) and confirms that the visual quality matches the quantitative improvement.

Improved accuracy of Surface Fitting: The improved accuracy in dense correspondence and surface fitting reported in Fig. 6 can

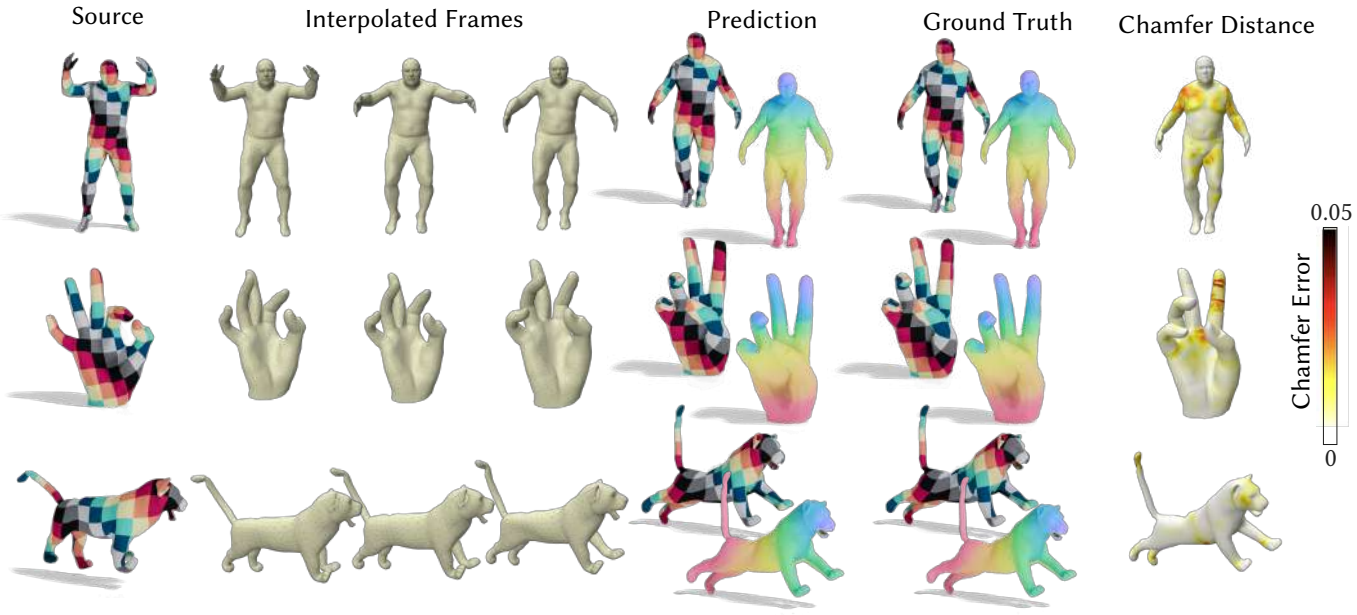

Fig. 9. Qualitative Results. *We showcase example pairs from the datasets exhibiting large deformations. Even in these challenging scenarios, our method produces plausible interpolations and recovers accurate correspondences with low surface matching errors.*

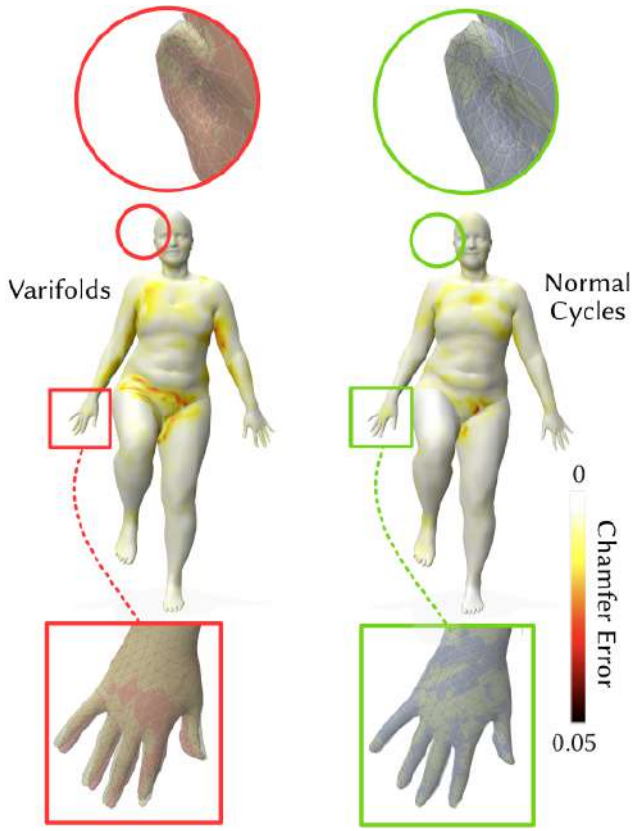

Fig. 10. Accuracy of fitting Varifolds vs Normal Cycles. *Example from DFAUST dataset. Left hand side is the result using Varifolds (red mesh) and right hand side using Normal Cycles (blue mesh) fitted to the target (yellow mesh). Importantly, unlike the numerical results, we match between source and target meshes of the same parameterisation to aid clarity of visualisation just for this figure. We note that high curvature areas such as the ears and fingers may not incur a high chamfer error but are misaligned by Varifolds, whereas the Normal Cycles provides a very high quality fit as it is sensitive to the changing local curvature.*

be attributed to the ability of Normal Cycles to better capture high-curvature regions, such as fingers and ears; we provide close-up illustrations of this in Fig. 10. For this figure, we do not randomise the mesh structure of the visualisation to compare the quality of the fit more easily.

Quality of Interpolation - SCLERP vs SLERP: The inclusion of the skeleton in ARC-Flow improved the recovery of plausible intermediate poses; however, SLERP interpolation along the shortest path leads to interpolation paths which do not always follow what one would perceive as natural motion. In Figs. 1, 11 and 12, we illustrate the improved quality of the interpolations generated by our SCLERP based method in comparison.

Constrained Bone Optimisation: ARC-Flow employs soft constraints to encourage locally rigid deformation of the internal skeleton. This necessitates additional hyperparameters and scheduling mechanisms to balance loss terms, which may require tuning for different target geometries. While ARC-Flow exhibits a clear trade-off between flow accuracy for skeletal rigid deformation and final pose—requiring shape-specific parameter tuning, our constrained optimisation approach ensures consistent alignment between sample sets throughout the interpolation process to within a predefined tolerance. We illustrate the advantages of constrained optimisation over soft constraints in Fig. 1, when both approaches are applied to a challenging pose from the MANO dataset.

NC Compression - Surface Fidelity and Timings: In Fig. 13 we match surfaces with vastly different resolutions (i.e. 12k matched to 120k vertices). Compression allows us to perform this in a computationally efficient manner; we can see that the Chamfer error results for as few as 10k compressed mesh samples are visually indistinguishable from the full resolution but obtained at a huge reduction in computational (and memory) cost. An illustration of the resulting interpolations is provided in Fig. 20.

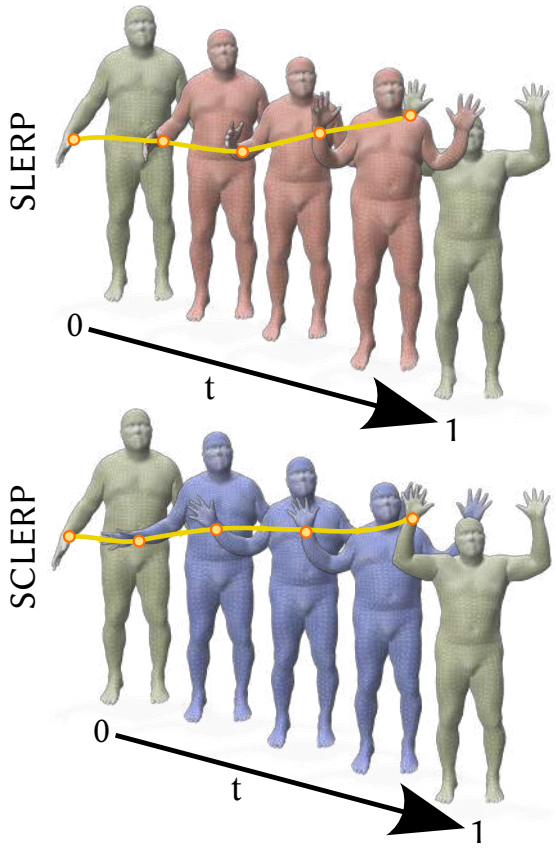

Fig. 11. Comparison of Interpolation Paths. *SLERP (**top**) yields a sequence of individually valid poses between the source and target, but the overall motion trajectory appears unnatural (e.g., raising the arms via an awkward route). In contrast, SCLERP (**bottom**) produces a smooth, more natural interpolation path that aligns with typical human movement.*

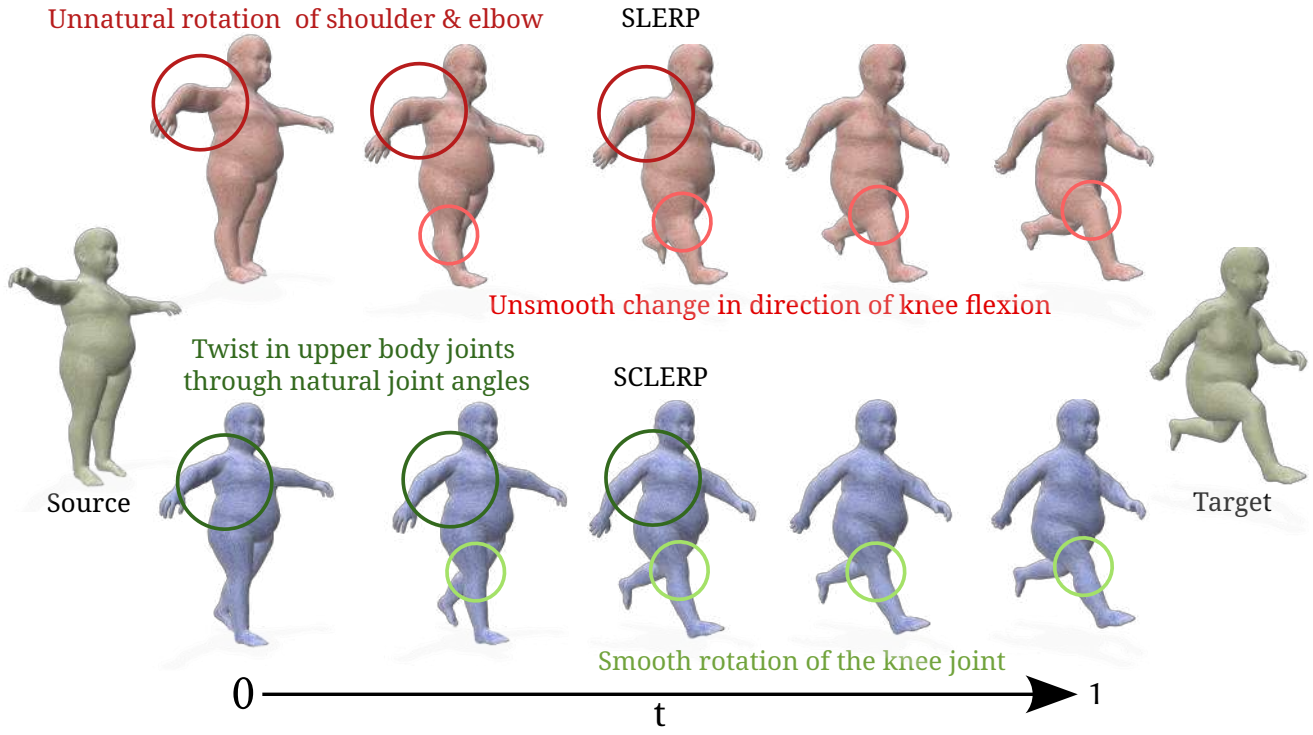

Fig. 12. Comparison of interpolation paths. *ARC-Flow [Hartshorne et al. 2025] using SLERP interpolation (top) versus our proposed method using SCLERP (bottom). While SLERP ensures each individual pose is valid, the resulting trajectory often introduces unnatural motions, such as implausible shoulder and elbow rotations or abrupt reversals in knee flexion. In contrast, our approach produces more physically plausible motion, particularly in the upper body and knee joints and also achieves higher quality recovery of dense correspondence.*

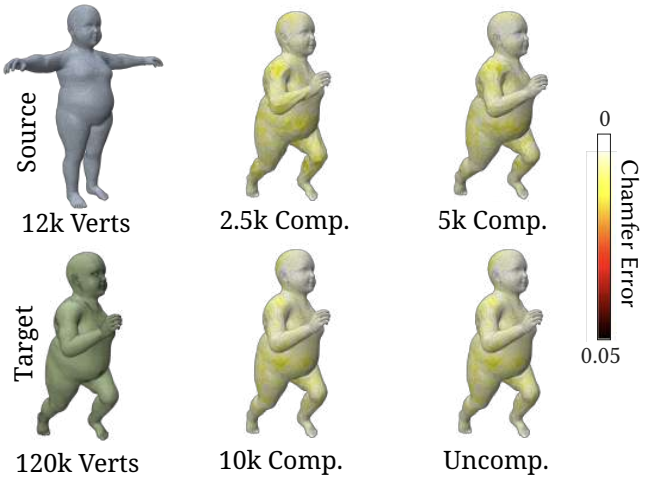

Fig. 13. Normal Cycle Compression: *We match a 12k source mesh from the TOPKIDS dataset to a 120k high resolution target; results are shown with mesh compression down to 2.5k, 5k and 10k as well as the full uncompressed 120k. Compression yields comparable performance but with greatly reduced run times.*

5.4 Additional Experiments

Choice of Skeleton: We provide further evidence of the flexibility of our method in relation to the choice of skeleton. In Figure 14 we illustrate that the same high quality result is achieved with two different well known skeletons. In Figure 18, we also demonstrate the effectiveness of our method on examples from the TOSCA dataset, where we have manually defined our own skeletons in Blender.

Timings: We evaluate the runtime of our Normal Cycles based approach compared to the Varifold method of ARC-Flow. We randomly select 10 pairs of shapes to solve for the interpolation and correspondence; we match parameters where appropriate, such as the same learning rate and number of ODE steps used by the solver. We record the wall clock time and Chamfer distance every 500 epochs. We repeat this experiment 5 times and report the mean and standard deviations in Fig. 15.

Without using compression it yields runtimes comparable to ARC-Flow, requiring minutes to compute high-quality trajectories and dense correspondences between dataset samples on a NVIDIA RTX A5000. Although computing NC is computationally more expensive, it achieves higher quality alignment to the data in fewer epochs. This is also in sharp contrast to deep learning methods e.g. SMS, that require a minimum of 24 hours pre-computing on high-performance GPU hardware for each new dataset.

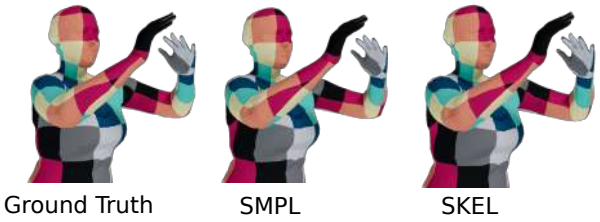

Fig. 14. SMPL vs SKEL. *Qualitative comparison confirming that the results of using different internal skeletons are indistinguishable.*

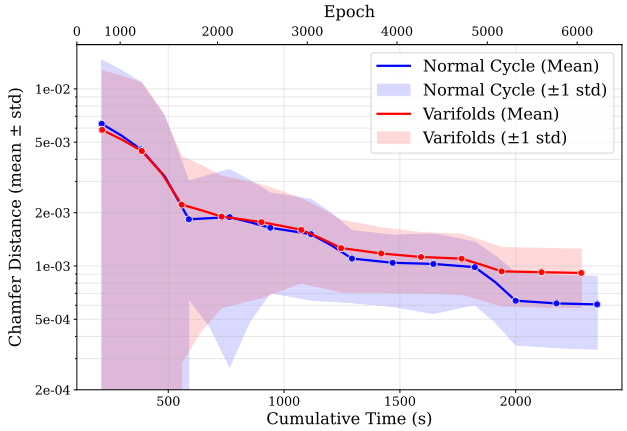

Fig. 15. Runtime Comparison: Varifolds vs Normal Cycles. *We compare per-epoch runtimes and total wall clock times and show, empirically, that the more computationally intensive Normal Cycles formulation outperforms Varifolds in overall time to achieve the same alignment quality.*

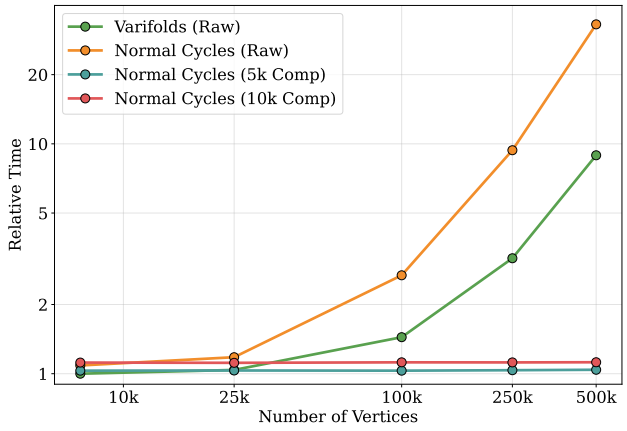

Fig. 16. Scaling to high-resolution meshes. *Wall-clock times for Varifolds, Normal Cycles, and compressed Normal Cycles on DFAUST samples, relative to uncompressed Varifolds at native resolution (~7k vertices). While Varifolds and raw Normal Cycles grow exponentially with resolution, compression (5k/10k) maintains constant computational cost across all mesh resolutions.* ***Note:*** *both axes are log scale.*

We quantitatively analyse the timing of our method when using the compressed forms. Using the DFAUST dataset, we apply "Instant Field-Aligned Meshes" algorithm [Jakob et al. 2015] to increase the resolution, then compress the result to specified levels (Uncompressed/5k/10k), and optimise for 5k epochs. Fig. 16 illustrates that relative training times of different sized meshes depend solely on the compressed size i.e. are independent of original mesh resolution. Using compression obtains dramatic computational savings, while maintaining similar perceptual quality, allowing us to scale to high resolution meshes. In contrast, the functional-mapping based approach SMS requires large permutation matrices, limiting scalability due to memory requirements, while ESA's dense pairwise computations scale poorly. The raw data table is provided in Appendix D.

Fitting & Transfer of Skeleton to Real World Data: Despite the increasing quality of automated rigging methods, an alternative, is designing a rig for one shape and then transferring it to other. In Fig. 19, we demonstrate the transfer of a known MANO skeleton to real world scans of hands.

6 CONCLUSIONS

We introduced multiple innovative techniques that improve unsupervised learning of natural 3D shape interpolation paths while advancing the accuracy of automatic dense shape correspondences beyond current standards. Through both numerical evaluations and visual examples across diverse experiments, we demonstrated efficacy of this new approach.

Limitations: While our method achieves state-of-the-art results, it has several limitations. First, like ARC-Flow, we assume diffeomorphic deformations and thus cannot correct topology errors in either the source or target mesh. Second, our current skeleton-based formulation is explicitly targeted to handle different poses of the same identity; it is not designed to accommodate changes in shape identity since we restrict bones to rigid transformations. Third, the Normal Cycle framework does not support partial inputs and is sensitive to very noisy target shapes.

Future Work: Partial shape matching within the framework of geometric measure theory has received limited attention, with existing work restricted to Varifolds [Antonsanti et al. 2021]. In contrast, partial matching using Normal Cycles remains largely unexplored, and we regard this as a promising avenue for future research.

ACKNOWLEDGMENTS

We thank Michelle Wu and Marting Parsons for help in rendering figures and video production. This work was supported by 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.

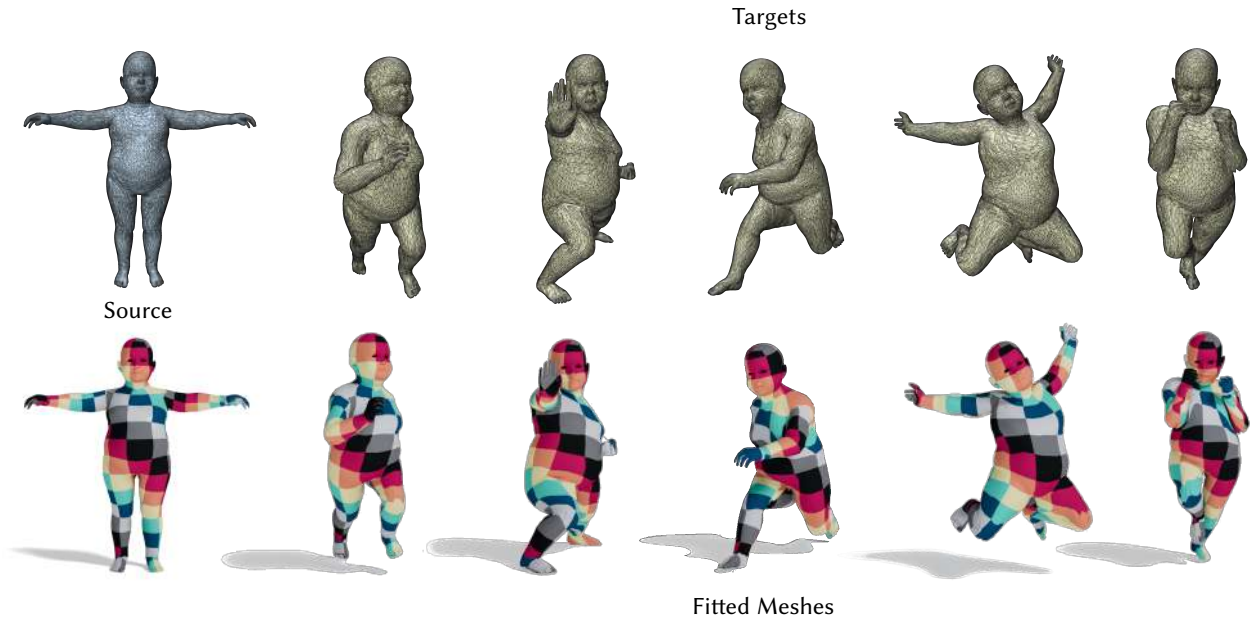

Fig. 17. Qualitative comparison on TOPKIDS dataset *The preservation of conformality under the deformation and consistent correspondences learned by our proposed method are illustrated by transferring a texture under the learned flows from the source to a variety of poses taken from the TOPKIDS dataset. This task is particularly demanding because the dataset's meshes are deliberately irregularly triangulated, with triangles of diverse shapes and sizes and no consistent grid structure..*

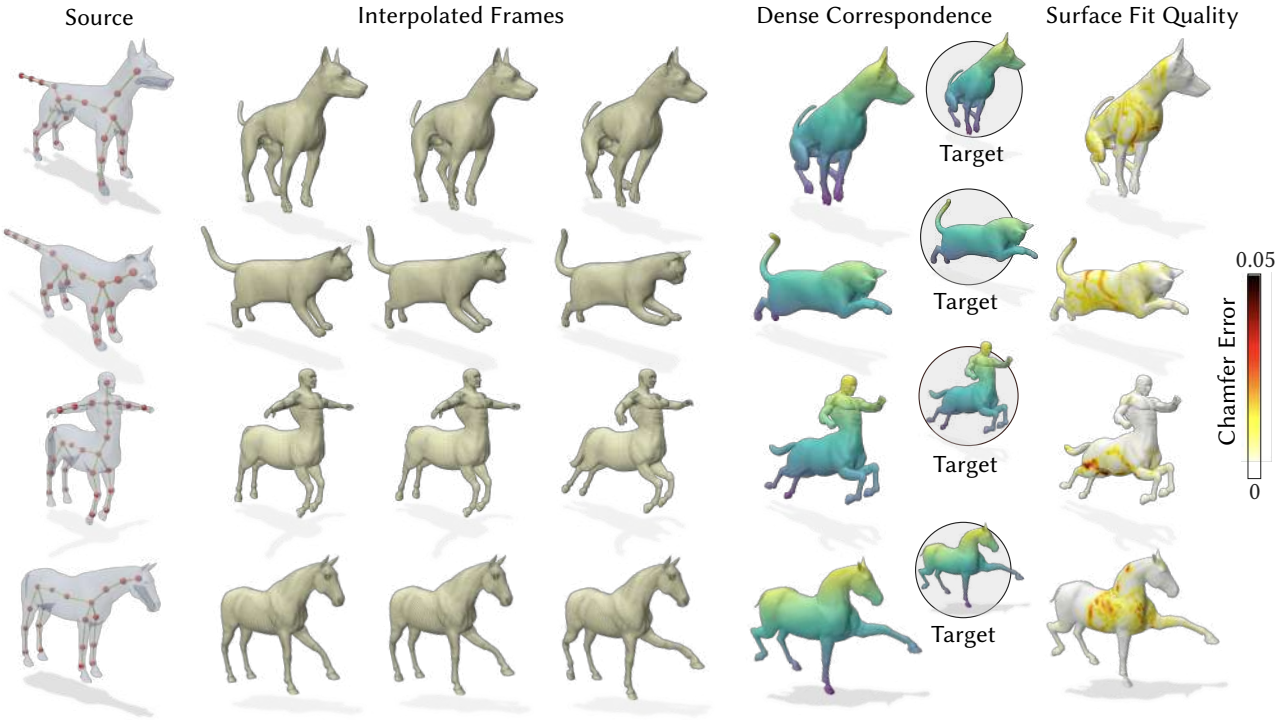

Fig. 18. Manually-Defined Skeletons. *Our method is not restricted to detailed, artist-crafted rigs. Using simple Blender-defined rigs, our method still accurately aligns TOSCA shape pairs and recovers high-fidelity dense correspondences.*

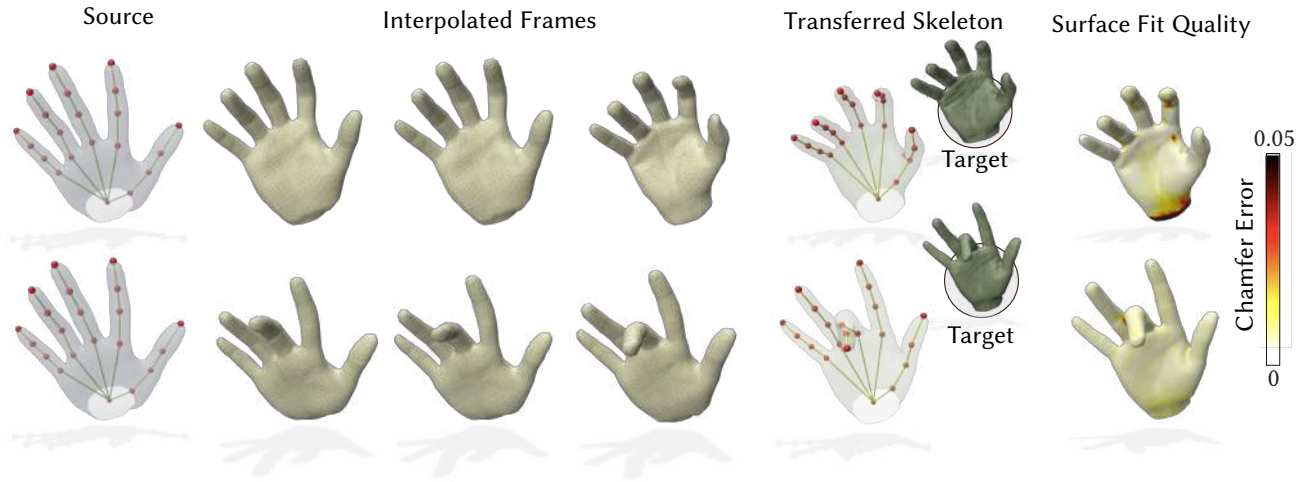

Fig. 19. Skeleton Transfer. *We fit a hand template to two real-world scans from the MANO dataset and show high quality of fit even in the presence of noise & ability to transfer the skeleton into the target domain.*

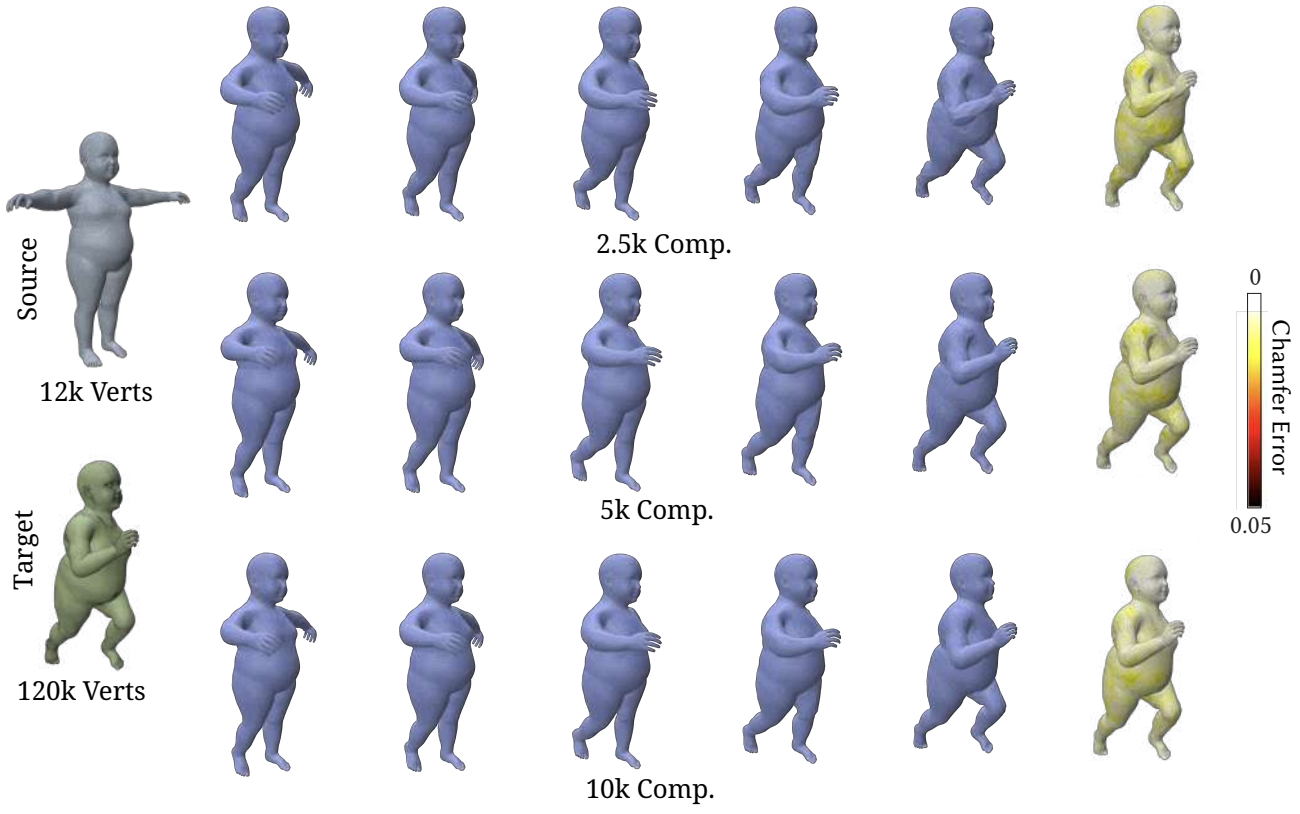

Fig. 20. Qualitative comparison on learned interpolation using different levels of Normal Cycle compression. *We match a 12k source mesh from the TOPKIDS dataset to a 120k high resolution target; results are shown with mesh compression down to 2.5k, 5k and 10k. The recovered interpolation paths are nearly identical but with greatly reduced run times.*

REFERENCES

Noam Aigerman, Kunal Gupta, Vladimir G Kim, Siddhartha Chaudhuri, Jun Saito, and Thibault Groueix. 2022. Neural jacobian fields: Learning intrinsic mappings of arbitrary meshes. *arXiv preprint arXiv:2205.02904* (2022).

Pierre-Louis Antonsanti, Joan Glaunès, Thomas Benseghir, Vincent Jugnon, and Irène Kaltenmark. 2021. Partial matching in the space of varifolds. In *International Conference on Information Processing in Medical Imaging*. Springer, 123–135.

Souhaib Attaiki and Maks Ovsjanikov. 2024. Shape Non-rigid Kinematics (SNK): A Zero-Shot Method for Non-Rigid Shape Matching via Unsupervised Functional Map Regularized Reconstruction. *Advances in Neural Information Processing Systems* 36 (2024).

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

Martin Bauer, Philipp Harms, and Peter W Michor. 2010. Sobolev metrics on shape space of surfaces. *arXiv preprint arXiv:1009.3616* (2010).

M Faisal Beg, Michael I Miller, Alain Trouvé, and Laurent Younes. 2005. Computing large deformation metric mappings via geodesic flows of diffeomorphisms. *International journal of computer vision* 61, 2 (2005), 139–157.

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

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

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. 2018. *JAX: composable transformations of Python+NumPy programs*. http://github.com/jax-ml/jax

Christopher Brandt, Christoph von Tycowicz, and Klaus Hildebrandt. 2016. Geometric flows of curves in shape space for processing motion of deformable objects. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 295–305.

Alexander M Bronstein, Michael M Bronstein, and Ron Kimmel. 2008. *Numerical geometry of non-rigid shapes*. Springer Science & Business Media.

Dongliang Cao and Florian Bernard. 2022. Unsupervised deep multi-shape matching. In *European Conference on Computer Vision*. Springer, 55–71.

Dongliang Cao, Marvin Eisenberger, Nafie El Amrani, Daniel Cremers, and Florian Bernard. 2024. Spectral Meets Spatial: Harmonising 3D Shape Matching and Interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3658–3668.

Dongliang Cao, Paul Roetzer, and Florian Bernard. 2023. Unsupervised Learning of Robust Spectral Shape Matching. *ACM Transactions on Graphics (TOG)* (2023). https://doi.org/10.1145/3592107

Benjamin Charlier, Jean Feydy, Joan Glaunès, François-David Collin, and Ghislain Durif. 2020. Kernel operations on gpu, without memory overflows kernel operations on the gpu, with autodiff, without memory overflows. *arXiv preprint arXiv:2004.11127* (2020).

Benjamin Charlier, Jean Feydy, Joan Alexis Glaunès, François-David Collin, and Ghislain Durif. 2021. Kernel Operations on the GPU, with Autodiff, without Memory Overflows. *Journal of Machine Learning Research* 22, 74 (2021), 1–6. http://jmlr.org/papers/v22/20-275.html

Nicolas Charon, Benjamin Charlier, Joan Glaunès, Pietro Gori, and Pierre Roussillon. 2020. Fidelity metrics between curves and surfaces: currents, varifolds, and normal cycles. In *Riemannian geometric statistics in medical image analysis*. Elsevier, 441–477.

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

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. 2018. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31.

David Cohen-Steiner and Jean-Marie Morvan. 2003. *Approximation of Normal Cycles*. Technical Report RR-4723. INRIA. https://inria.hal.science/inria-00071863

Konstantinos Daniilidis. 1999. Hand-eye calibration using dual quaternions. *The International Journal of Robotics Research* 18, 3 (1999), 286–298.

Nicolas Donati, Abhishek Sharma, and Maks Ovsjanikov. 2020. Deep geometric functional maps: Robust feature learning for shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8592–8601.

Marvin Eisenberger and Daniel Cremers. 2020. 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*. Springer, 179–196.

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

M. Eisenberger, Z. Lähner, and D. Cremers. 2019. Divergence-Free Shape Correspondence by Deformation. In *Computer Graphics Forum*, Vol. 38. 1–12.

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

Herbert Federer. 1959. Curvature measures. *Trans. Amer. Math. Soc.* 93, 3 (1959), 418–491.

Joseph HG Fu. 1994. Curvature measures of subanalytic sets. *American Journal of Mathematics* (1994), 819–880.

Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 2018. 3d-coded: 3d correspondences by deep deformation. In *Proceedings of the european conference on computer vision (ECCV)*. 230–246.

Kunal Gupta and Manmohan Chandraker. 2020. Neural Mesh Flow: 3D Manifold Mesh Generation via Diffeomorphic Flows. *Advances in Neural Information Processing Systems* 33 (2020), 1747–1758.

Emmanuel Hartman, Emery Pierson, Martin Bauer, Nicolas Charon, and Mohamed Daoudi. 2023a. Bare-esa: A riemannian framework for unregistered human body shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14181–14191.

Emmanuel Hartman, Yashil Sukurdeep, Eric Klassen, Nicolas Charon, and Martin Bauer. 2023b. Elastic shape analysis of surfaces with second-order sobolev metrics: a comprehensive numerical framework. *International Journal of Computer Vision* 131, 5 (2023), 1183–1209.

Adam Hartshorne, Allen Paul, Tony Shardlow, and Neill D.F. Campbell. 2025. ARC-Flow : Articulated, Resolution-Agnostic, Correspondence-Free Matching and Interpolation of 3D Shapes Under Flow Fields. In *2025 International Conference on 3D Vision (3DV)*. IEEE.

Behrend Heeren, Martin Rumpf, Peter Schröder, Max Wardetzky, and Benedikt Wirth. 2014. Exploring the geometry of the space of shells. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 247–256.

Behrend Heeren, Martin Rumpf, Max Wardetzky, and Benedikt Wirth. 2012. Time-discrete geodesics in the space of shells. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 1755–1764.

Kai Hormann and Giinther Greiner. 2000. MIPS: An Efficient Global Parametrization Method. *France on 1-7 July 1999. Proceedings, Volume 1. Curve and Surface Design. F61775-99-WF068* (2000), 153.

Yixin Hu, Teseo Schneider, Bolun Wang, Denis Zorin, and Daniele Panozzo. 2020. Fast tetrahedral meshing in the wild. *ACM Transactions on Graphics (ToG)* 39, 4 (2020), 117–1.

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

Wenzel Jakob, Marco Tarini, Daniele Panozzo, and Olga Sorkine-Hornung. 2015. Instant field-aligned meshes. *ACM transactions on graphics (TOG)* 34, 6 (2015), 1–15.

Puhua Jiang, Mingze Sun, and Ruqi Huang. 2023. Non-rigid shape registration via deep functional maps prior. *arXiv preprint arXiv:2311.04494* (2023).

Irene Kaltenmark, Benjamin Charlier, and Nicolas Charon. 2017. 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*. 3346–3355.

Ladislav Kavan, Steven Collins, Carol O'Sullivan, and Jiri Zara. 2006. Dual quaternions for rigid transformation blending. *Trinity College Dublin* 46 (2006).

Ladislav Kavan, Steven Collins, Jiří Žára, and Carol O'Sullivan. 2007. Skinning with dual quaternions. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*. 39–46.

Marilyn Keller, Keenon Werling, Soyong Shin, Scott Delp, Sergi Pujades, C Karen Liu, and Michael J Black. 2023. From skin to skeleton: Towards biomechanically accurate 3d digital humans. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–12.

Benjamin Kenwright. 2023. Dual-quaternion interpolation. *arXiv preprint arXiv:2303.13395* (2023).

Patrick Kidger and Cristian Garcia. 2021. Equinox: neural networks in JAX via callable PyTrees and filtered transformations. *Differentiable Programming workshop at Neural Information Processing Systems 2021* (2021).

Peter Nicholas Krämer. 2024. *Implementing probabilistic numerical solvers for differential equations*. Ph. D. Dissertation. Universit"at T"ubingen. https://github.com/pnkraemer/probdiffeq

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

Hamid Laga, Qian Xie, Ian H Jermyn, and Anuj Srivastava. 2017. Numerical inversion of SRNF maps for elastic shape analysis of genus-zero surfaces. *IEEE transactions on pattern analysis and machine intelligence* 39, 12 (2017), 2451–2464.

Zorah Lähner, Emanuele Rodola, Michael M Bronstein, Daniel Cremers, Oliver Burghard, Luca Cosmo, Alexander Dieckmann, Reinhard Klein, Y Sahillioğlu, et al. 2016. SHREC'16: Matching of deformable shapes with topological noise. In *Eurographics Workshop on 3D Object Retrieval, EG 3DOR*. Eurographics Association, 55–60.

Lei Li, Nicolas Donati, and Maks Ovsjanikov. 2022. Learning multi-resolution functional maps with spectral attention for robust shape matching. *Advances in Neural Information Processing Systems* 35 (2022), 29336–29349.

Yang Li, Hikari Takehara, Takafumi Taketomi, Bo Zheng, and Matthias Nießner. 2021. 4dcomplete: Non-rigid motion estimation beyond the observable surface. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 12706–12716.

Kaizhao Liang, Lizhang Chen, Bo Liu, and Qiang Liu. 2024. Cautious optimizers: Improving training with one line of code. *arXiv preprint arXiv:2411.16085* (2024).

Selena Zihan Ling, Nicholas Sharp, and Alec Jacobson. 2022. Vectoradam for rotation equivariant geometry optimization. *Advances in Neural Information Processing Systems* 35 (2022), 4111–4122.

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

Hsueh-Ti Derek Liu, Francis Williams, Alec Jacobson, Sanja Fidler, and Or Litany. 2022. Learning smooth neural functions via lipschitz regularization. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–13.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)* 34, 6 (Oct. 2015), 248:1–248:16.

Baorui Ma, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. 2020. Neural-pull: Learning signed distance functions from point clouds by learning to pull space onto surfaces. *arXiv preprint arXiv:2011.13495* (2020).

Robin Magnet and Maks Ovsjanikov. 2023. Scalable and efficient functional map computations on dense meshes. In *Computer Graphics Forum*, Vol. 42. Wiley Online Library, 89–101.

Robin Magnet and Maks Ovsjanikov. 2024. Memory-scalable and simplified functional map learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4041–4050.

Ishit Mehta, Manmohan Chandraker, and Ravi Ramamoorthi. 2022. A level set theory for neural implicit evolution under explicit flows. In *European Conference on Computer Vision*. Springer, 711–729.

Simone Melzi, Jing Ren, Emanuele Rodolà, Abhishek Sharma, Peter Wonka, and Maks Ovsjanikov. 2019. ZoomOut: spectral upsampling for efficient shape correspondence. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–14.

Tiago Novello, Vinicius Da Silva, Guilherme Schardong, Luiz Schirmer, Helio Lopes, and Luiz Velho. 2023. Neural Implicit Surface Evolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14279–14289.

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

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

Allen Paul, Neill Campbell, and Tony Shardlow. 2024. Sparse Nystrom Approximation of Currents and Varifolds. *arXiv preprint arXiv:2406.09932* (2024).

Allen Paul, Neill Campbell, and Tony Shardlow. 2025. Sparse Randomized Approximation of Normal Cycles. arXiv:2503.01057 [math.NA] https://arxiv.org/abs/2503.01057

Emery Pierson, Mohamed Daoudi, and Sylvain Arguillere. 2022. 3D Shape Sequence of Human Comparison and Classification Using Current and Varifolds. In *European Conference on Computer Vision*. Springer, 523–539.

John Platt and Alan Barr. 1987. Constrained differential optimization. In *Neural Information Processing Systems*.

Paul Roetzer, Ahmed Abbas, Dongliang Cao, Florian Bernard, and Paul Swoboda. 2024. Discomatch: Fast discrete optimisation for geometrically consistent 3d shape matching. In *European Conference on Computer Vision*. Springer, 443–460.

Paul Roetzer and Florian Bernard. 2024. Spidermatch: 3d shape matching with global optimality and geometric consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14543–14553.

Javier Romero, Dimitris Tzionas, and Michael J Black. 2017. Embodied Hands: Modeling and Capturing Hands and Bodies Together. *ACM Transactions on Graphics* 36, 6 (2017).

Pierre Roussillon and Joan Alexis Glaunès. 2017. Surface matching using normal cycles. In *Geometric Science of Information: Third International Conference, GSI 2017, Paris, France, November 7-9, 2017, Proceedings 3*. Springer, 73–80.

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

Lu Sang, Zehranaz Canfes, Dongliang Cao, Florian Bernard, and Daniel Cremers. 2025. Implicit Neural Surface Deformation with Explicit Velocity Fields. arXiv:2501.14038 [cs.CV] https://arxiv.org/abs/2501.14038

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

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. 2020. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems* 33 (2020), 7462–7473.

Olga Sorkine and Marc Alexa. 2007. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, Vol. 4. Citeseer, 109–116.

Olga Sorkine and Mario Botsch. 2009. Interactive Shape Modeling and Deformation.. In *Eurographics (Tutorials)*. 11–37.

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

Matthias Vestner, Zorah Lähner, Amit Boyarski, Or Litany, Ron Slossberg, Tal Remez, Emanuele Rodola, Alex Bronstein, Michael Bronstein, Ron Kimmel, et al. 2017a. Efficient deformable shape correspondence via kernel matching. In *2017 international conference on 3D vision (3DV)*. IEEE, 517–526.

Matthias Vestner, Roee Litman, Emanuele Rodola, Alex Bronstein, and Daniel Cremers. 2017b. 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*. 3327–3336.

Wolfram Von Funck, Holger Theisel, and Hans-Peter Seidel. 2006. Vector field based shape deformations. *ACM Transactions on Graphics (ToG)* 25, 3 (2006), 1118–1125.

Peter Wintgen. 1982. Normal cycle and integral curvature for polyhedra in Riemannian manifolds. *Differential geometry* 21 (1982).

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

Guandao Yang, Serge Belongie, Bharath Hariharan, and Vladlen Koltun. 2021. Geometry processing with neural fields. *Advances in Neural Information Processing Systems* 34 (2021), 22483–22497.

Martina Zähle. 1986. Integral and current representation of Federer's curvature measures. *Archiv der Mathematik* 46 (1986), 557–567.

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

A NORMAL CYCLES (NC) IMPLEMENTATION

We illustrate the practical computation of the Normal Cycle loss for closed triangular meshes without boundaries using JAX [Bradbury et al. 2018] and Keops [Charlier et al. 2021]; the same Normal Cycles formulation can also be extended to handle meshes with boundaries.

Edge Centres & Weights

For a mesh $\mathcal{X}$ with vertices $V_{\mathcal{X}}$, faces $F_{\mathcal{X}}$, and edges $E_{\mathcal{X}}$, the edge centres and NC weights can be assembled directly from the mesh and its auxiliary adjacency/orientation data. A minimal JAX implementation is provided below.

```
import jax.numpy as jnp

# Pre-Defined Index Pairs (constant)
INDICES_1 = jnp.array([i - 1 for i in range(1, 6)
     for j in range(i + 1, 7)])
INDICES_2 = jnp.array([j - 1 for i in range(1, 6)
     for j in range(i + 1, 7)])

# Compute edge centres and weights for a tri mesh.
def calc_centres_and_weights(v, f, e, stack,
     coords):
    """
    Args:
        v:    Vertices (V,3)
        f:    Face Indices (F,3)
        e:    Unique Edges (E,2)
        e2f:  Edge-to-Face Adjacency (E,4)
        orient_signs: Orientation Coefficients
    Returns:
        edge_centres: Edge Centres (E,3)
        weights:      Edge Weights (E,15)
    """

    # --- Face Normals ---
    v1, v2, v3 = v[f[:,0]], v[f[:,1]], v[f[:,2]]
    g1, g2 = v2 - v1, v3 - v1
    v_normals = 0.5 * jnp.cross(g1, g2)
    v_normals /= jnp.linalg.norm(v_normals,
         axis=1, keepdims=True)

    # --- Reconstructed Normals ---
    v_normal_mod = jnp.vstack([v_normals,
         jnp.zeros((1, 3))])
    v_re = sum(orient_signs[i] *
         v_normal_mod[e2f[:, i]] for i in range(4))

    # --- Edge Geometry ---
    e0, e1 = v[e][:,0,:], v[e][:,1,:]
    fs = e1 - e0                    # Edge Vectors
    edge_centres = 0.5 * (e1 + e0)  # Edge Centres

    # --- NC weights ---
    f_norm = jnp.linalg.norm(fs, axis=1,
         keepdims=True)
    f_hat = fs / f_norm
    zeros = jnp.zeros_like(f_hat)
    a1 = jnp.hstack([f_hat, zeros])
    a2 = jnp.hstack([zeros, v_re])
    alpha = (a[:, INDICES_1] * b[:, INDICES_2] -
             a[:, INDICES_2] * b[:, INDICES_1])
    edge_weights = f_norm * alpha.reshape(-1, 15)
    return edge_centres, edge_weights
```

Edge-to-face adjacency: We define the array

$$\texttt{e2f} \in \{-1, 0, \ldots, |F_{\mathcal{X}}| - 1\}^{E\times 4},$$

where each row corresponds to an edge $e \in E_{\mathcal{X}}$ and stores the indices of up to four faces incident to e. In a closed, manifold triangle mesh, each edge has exactly two incident faces; boundary edges have a single valid entry, and the remaining slots are padded with -1.
Edge orientation coefficients: We define the array

$$\texttt{orient\_signs} \in \{-1, 0, +1\}^{E\times 4},$$

where each entry encodes the orientation of edge e within its i-th incident face listed in `e2f`. A value of $+1$ means the edge appears in the face with consistent orientation (counter-clockwise ordering), while -1 indicates the opposite. Entries corresponding to padded -1 face indices are set to 0.

The `e2f` and `orient_signs` arrays obtained from these preprocessing routines, along with $V_{\mathcal{X}}$, $F_{\mathcal{X}}$, and $E_{\mathcal{X}}$, provide the structural information needed to reconstruct per-edge centres and curvature descriptors.
Edge centres: Each edge $e = (i, j)$ is associated with a centre

$$c_e = \tfrac{1}{2}(x_i + x_j), \quad \forall\, (i, j)_e \in E_{\mathcal{X}}.$$

NC weights: Combining the reconstructed normals with edge geometry yields the per-edge NC weights

$$w_e \in \mathbb{R}^{15},$$

which live in the space of exterior 2-forms and encode curvature information along the edge.

Normal Cycle Loss

The *Normal Cyles* matching metric $d(\mathcal{X}, \mathcal{Y})$ between two shapes, $\mathcal{X}$ and $\mathcal{Y}$, can then be efficiently implemented using Keops [Charlier et al. 2021].

```
from pykeops.torch import LazyTensor

def nc_kernel(p_x, p_y, w_x, w_y, gamma):
    X = LazyTensor(p_x[:, None, :])
    Y = LazyTensor(p_y[None, :, :])
    D = X.sqdist(Y)          # Squared Distances
    K = (-gamma * D).exp()   # Gaussian Kernel
    Wx = LazyTensor(w_x[:, None, :])
    Wy = LazyTensor(w_y[None, :, :])
    dot = (Wx * Wy).sum()    # Weight Dot Products
    return (K * dot).sum()

def normal_cycle_loss(c_x, c_y, w_x, w_y, gamma):
    xx = nc_kernel(c_x, c_x, w_x, w_x, gamma)
    yy = nc_kernel(c_y, c_y, w_y, w_y, gamma)
    xy = nc_kernel(c_x, c_y, w_x, w_y, gamma)
    return xx + yy - 2 * xy
```

The arrays $C_{\mathcal{X}} = \{c_e\}_{e\in E_{\mathcal{X}}}$, $C_{\mathcal{Y}} = \{c_e\}_{e\in E_{\mathcal{Y}}}$ denote the sets of edge centres with $c_e \in \mathbb{R}^3$, $W_{\mathcal{X}} = \{w_e\}_{e\in E_{\mathcal{X}}}$, $W_{\mathcal{Y}} = \{w_e\}_{e\in E_{\mathcal{Y}}}$ denotes the sets of NC weights with $w_e \in \mathbb{R}^15$, and γ is the Gaussian kernel lengthscale.

B POSE INTERPOLATION DERIVATION

Even with fixed bone lengths, the translation of a joint is not independent of its rotation. The motion of a rigid body is naturally a screw motion—a rotation about an axis combined with a translation along that axis.

Dual-quaternions (DQ) are a well-established tool in computer graphics for representing rotational and translation solutions in a unified form. Given a quaternion rotation $\mathbf{q}$ and a three-dimensional translational vector $\vec{t}$, the dual-quaternion $\mathbf{Q}(\mathbf{q}, \vec{t})$ is given below:

$$\begin{aligned} \mathbf{Q}(\mathbf{q}, \vec{t}) &= \mathbf{q}_r + \mathbf{q}_d \\ \mathbf{q}_r &= \mathbf{q} \\ \mathbf{q}_t &= (0, t_x, t_y, t_z) \\ \mathbf{q}_d &= (\mathbf{q}_r \mathbf{q}_t)\,\frac{1}{2} \end{aligned} \tag{28}$$

Unified Representation of Rigid Transforms: We use dual quaternion (DQ) formulation for forward kinematic of the skeleton, as they represent motion as a combined rotation and translation along an axis (screw motion) into a single compact object. Consistent interpolation on SE(3) ensure smooth and mathematically valid rigid body motion by maintaining proper coupling between rotation and translation, unlike linear interpolation which can create unnatural paths despite reaching correct endpoints. For animated skeletons, maintaining the natural screw motion during interpolation often produces smoother and more physically plausible movements.

Furthermore, DQs tend to be more numerically stable and compact than a separate rotation and translation representation, while ensuring that the entire transform remains "rigid", which is especially beneficial in animation. In comparison, when interpolating the rotation and translation separately it is possible to observe undesirable artifacts.

C DUAL QUATERNION INTERPOLATION

Sep(LERP): extracts the translation and orientation, interpolate separately & reconstruct a new dual-quaternion.

$$\mathrm{LERP}(t; \vec{v_0}, \vec{v_1}) = (1-t)\vec{v_0} + (t)\vec{v_1} \tag{29}$$

$$\mathrm{SLERP}(t; q_0, q_1) = q_0(q_0^* q_1)^t \tag{30}$$

Dual-Quaternion Screw Linear Interpolation (ScLERP) [Kavan et al. 2006]: extends quaternion SLERP [Shoemake 1985] to handle the complete 6 DOF rigid body transformation encoded in dual-quaternions. While SLERP operates solely on rotational motion, ScLERP interpolates both rotation and translation simultaneously using the dual-quaternion's screw parameters. This unified approach provides smooth interpolation across all degrees of freedom through the dual-quaternion power function. Daniilidis [1999] provided a breakdown of the screw parameters in relation to a dual-quaternion

$$\begin{aligned} \mathbf{Q} &= \begin{pmatrix} w_r \\ \vec{v}_r \end{pmatrix} + \epsilon \begin{pmatrix} w_d \\ \vec{v}_d \end{pmatrix} \\ &= \begin{pmatrix} w_r \\ \vec{v}_r \end{pmatrix} + \epsilon \begin{pmatrix} -\frac{1}{2}\vec{v}_r \cdot \vec{t} \\ \frac{1}{2}(w_r \vec{t} + (\vec{t} \times \vec{v}_r)) \end{pmatrix} \\ &= \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right)\vec{l} \end{pmatrix} + \epsilon \begin{pmatrix} -\frac{d}{2}\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right)\vec{m} + \frac{d}{2}\cos\left(\frac{\theta}{2}\right)\vec{l} \end{pmatrix} \end{aligned} \tag{31}$$

Convert screw-parameters to dual-quaternion
Given Equation (31), we can derive the associated dual-quaternion components in relation to specific screw-parameters (and vice versa).

$$\begin{aligned} w_r &= \cos\left(\frac{\theta}{2}\right) \\ \vec{v}_r &= \vec{l}\sin\left(\frac{\theta}{2}\right) \\ w_d &= -\frac{d}{2}\sin\left(\frac{\theta}{2}\right) \\ \vec{v}_d &= \sin\left(\frac{\theta}{2}\right)\vec{m} + \frac{d}{2}\cos\left(\frac{\theta}{2}\right)\vec{l} \end{aligned}$$

Convert dual-quaternion to screw-parameters

$$\theta = 2\cos^{-1}(w_r) \quad \text{(angle from real component)}$$

$$\vec{l} = \frac{\vec{v}_r}{\sin(\theta/2)} \quad (\text{or } ||\vec{v}_r|| \text{ axis from the real component})$$

$$d = \frac{-2w_d}{\sin(\theta/2)} \quad (\text{or } \vec{t} \cdot \vec{l})$$

$$\vec{m} = \frac{\vec{v}_d}{\sin(\theta/2)} - \frac{d}{2}\frac{\cos(\theta/2)}{\sin(\theta/2)}\vec{l} \quad \left(\frac{\cos}{\sin} = \cot\right) \tag{32}$$

Dual-Quaternion power: The dual-quaternion representation takes the form

$$\begin{aligned} \mathbf{Q} &= \cos\left(\frac{\theta + \epsilon d}{2}\right) + (\vec{l} + \epsilon\vec{m})\sin\left(\frac{\theta + \epsilon d}{2}\right) \\ &= \cos\left(\frac{\hat{\theta}}{2}\right) + \hat{v}\sin\left(\frac{\hat{\theta}}{2}\right) \end{aligned} \tag{33}$$

where $\mathbf{Q}$ is a unit dual-quaternion, $\hat{v}$ is a unit dual-vector ($\hat{v} = \vec{l} + \epsilon\vec{m}$), and $\hat{\theta}$ is a dual-angle ($\hat{\theta} = \theta + \epsilon d$).

$$\mathrm{ScLERP}(t; \mathbf{Q}_A, \mathbf{Q}_B) = \mathbf{Q}_A(\mathbf{Q}_B^* \mathbf{Q}_B)^t\,, \tag{34}$$

where $\mathbf{Q}_A$ and $\mathbf{Q}_B$ are the start and end unit dual-quaternion and t is the interpolation amount from 0.0 to 1.0.

A dual-quaternion is able to transform a 3D vector coordinate as,

$$\mathbf{p}' = \mathbf{Q}\mathbf{p}\mathbf{Q}^{-1} \tag{35}$$

where $\mathbf{Q}$ is a unit dual-quaternion representing the transform, $\mathbf{Q}^{-1}$ is the inverse of the unit dual-quaternion transform. $\mathbf{p}$ and $\mathbf{p}'$ are the dual-quaternions holding 3D vector coordinate to before and after the transformation i.e. $p = (1, 0, 0, 0) + \epsilon(0, v_x, v_y, v_z)$.

As seen in Figure 1, it is possible that interpolation under ScLERP may follow a different geodesic path compared to Sep(LERP), which is formed from the independent interpolation of translation (linear) and rotation (via SLERP).

Table 1. 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 best results for each metric are highlighted in **red**, second best in **yellow**.*

Method	MANO			DFAUST			SMAL		
	Geodesic↑	Chamfer↑	Conformal↑	Geodesic↑	Chamfer↑	Conformal↑	Geodesic↑	Chamfer↑	Conformal↑
OURs	**0.94** ± 0.14	**0.92** ± 0.06	**0.86** ± 0.15	**0.96** ± 0.06	**0.89** ± 0.06	**0.84** ± 0.10	**0.95** ± 0.06	**0.93** ± 0.03	**0.82** ± 0.04
ARC-Flow [Hartshorne et al. 2025]	**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 [Cao et al. 2024]	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 [Hartman et al. 2023b]	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. [Eisenberger and Cremers 2020]	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 [Eisenberger et al. 2019]	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
Var + DQ	0.89 ± 0.16	0.84 ± 0.08	0.79 ± 0.12	0.94 ± 0.12	0.87 ± 0.07	0.72 ± 0.16	0.93 ± 0.10	0.88 ± 0.06	0.73 ± 0.07
Var + DQ + CO	0.92 ± 0.12	0.90 ± 0.07	0.86 ± 0.10	0.94 ± 0.13	0.88 ± 0.05	0.85 ± 0.10	0.94 ± 0.08	0.93 ± 0.04	0.85 ± 0.04
NC + DQ + CO	0.92 ± 0.18	0.89 ± 0.11	0.86 ± 0.11	0.93 ± 0.15	0.88 ± 0.07	0.85 ± 0.08	0.94 ± 0.15	0.93 ± 0.04	0.85 ± 0.03

D FURTHER QUANTITATIVE RESULTS

D.1 Detailed Comparative Results

Detailed numerical results comparing all methods and on the three datasets; MANO, DFAUST and SMAL, are provided in Table 1. Figures 21 to 23 visualize the mean values and confidence intervals for all metrics comparing our method against the state-of-the-art approaches SMS [Cao et al. 2024] and ESA [Hartman et al. 2023b]. Our method demonstrates improvement across all measures for each dataset, showing both higher mean values and reduced variance in results.

D.2 Wall-Clock Training Times by Vertex Count

Table 2 illustrates that training times depend solely on compressed size and are independent of original mesh resolution.

Table 2. *Timing Results in mins::secs for 6000 epochs using samples from DFaust dataset. The best method for each mesh resolution is highlighted in **red**, with the second best in **yellow**.*

Method	Comp.	7k	25k	100k	250k	500k
Var	Raw	**17:49**	**18:30**	25:41	56:38	158:52
NC	Raw	19:21	21:01	47:45	167:07	589:37
NC	5k	**18:22**	**18:24**	**18:22**	**18:27**	**18:32**
NC	10k	19:55	19:52	**19:58**	**19:57**	**19:59**

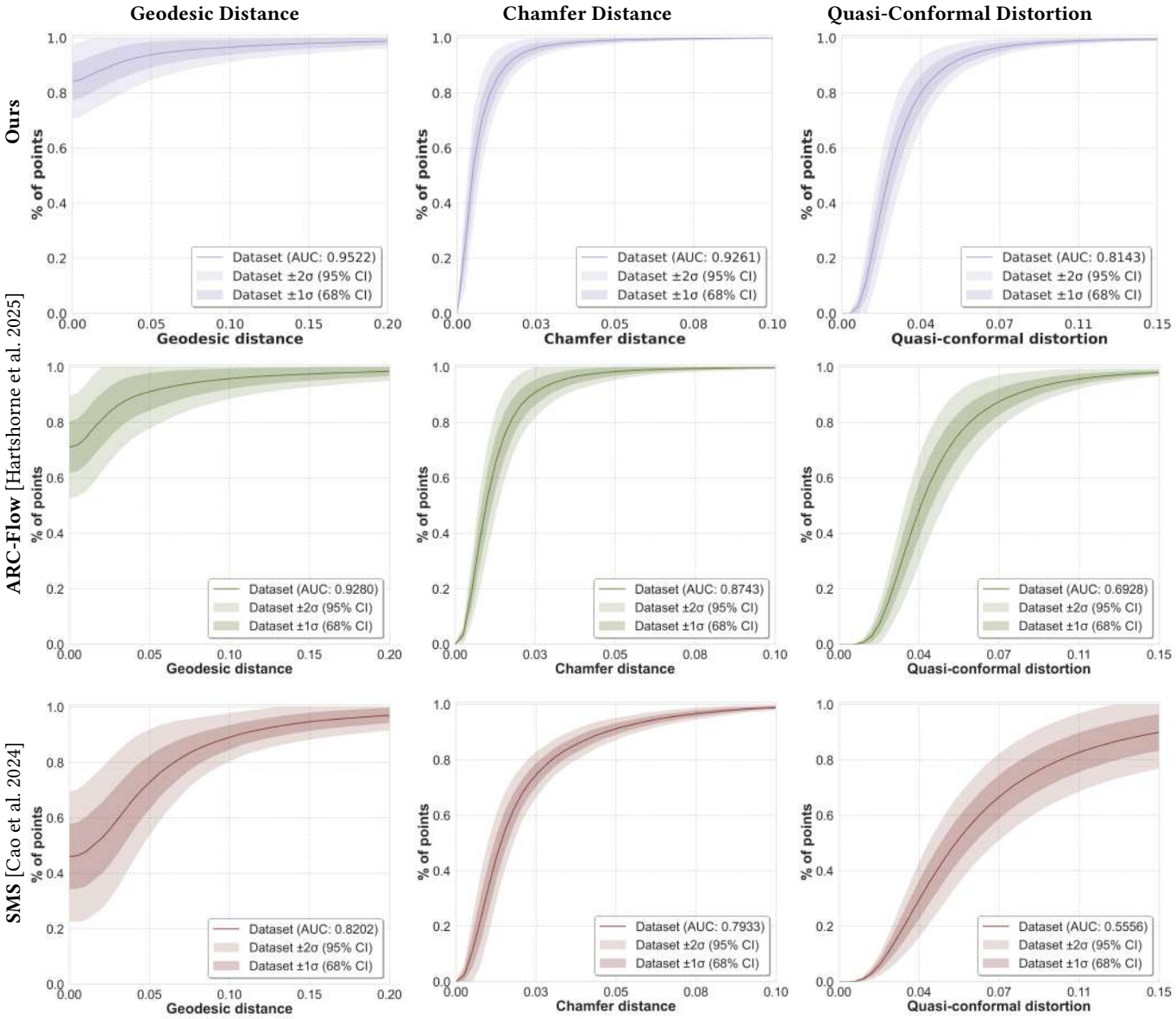

Fig. 21. Interpolation results for MANO: Ours vs ARC-Flow [Hartshorne et al. 2025] & SMS [Cao et al. 2024]. *Mean and confidence intervals for the three metrics are shown; top row has our results, middle SMS & bottom row ESA.*

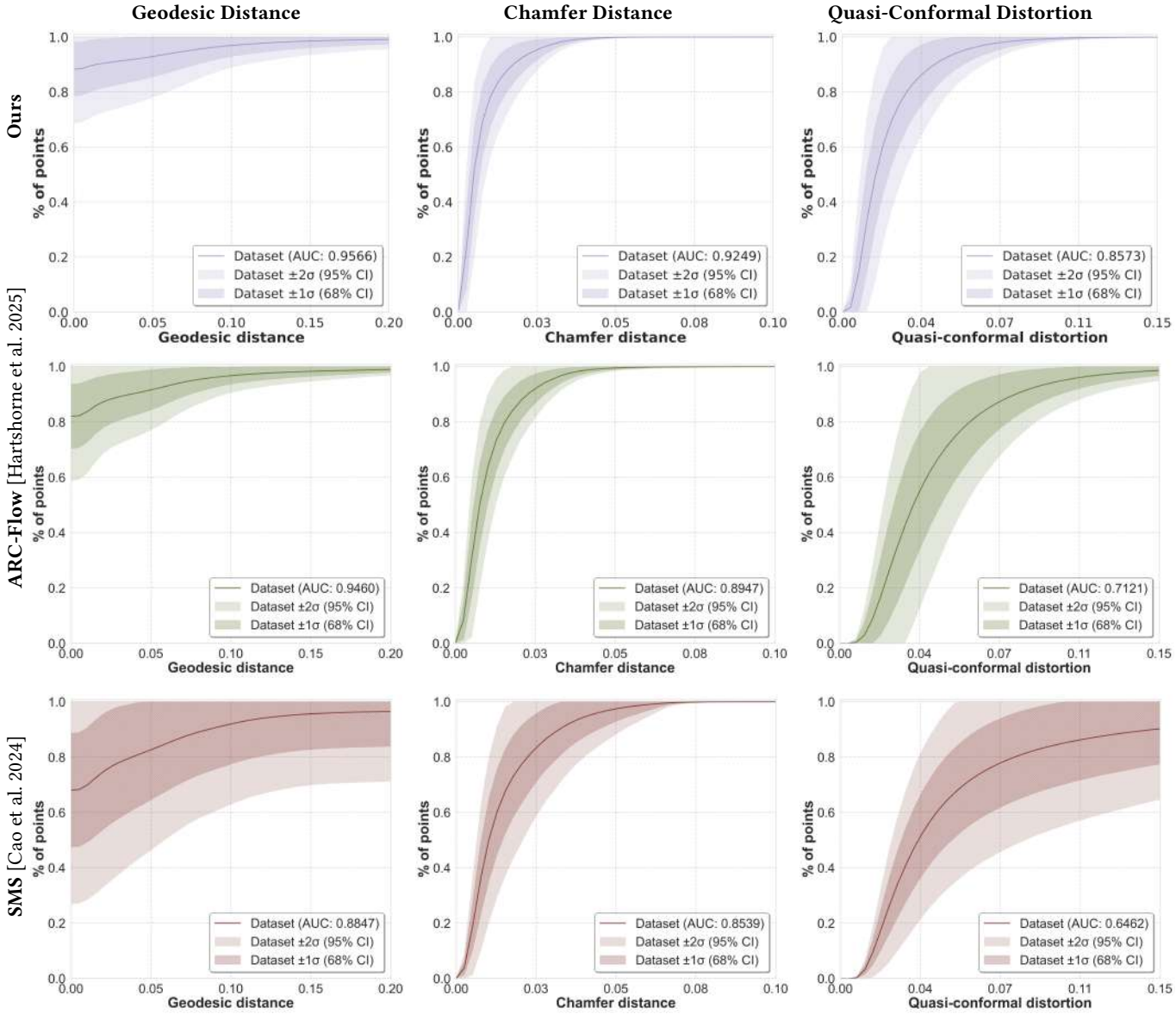

Fig. 22. Interpolation results for DFAUST: Ours vs ARC-Flow [Hartshorne et al. 2025] & SMS [Cao et al. 2024]. *Mean and confidence intervals for the three metrics are shown; top row has our results, middle SMS & bottom row ESA.*

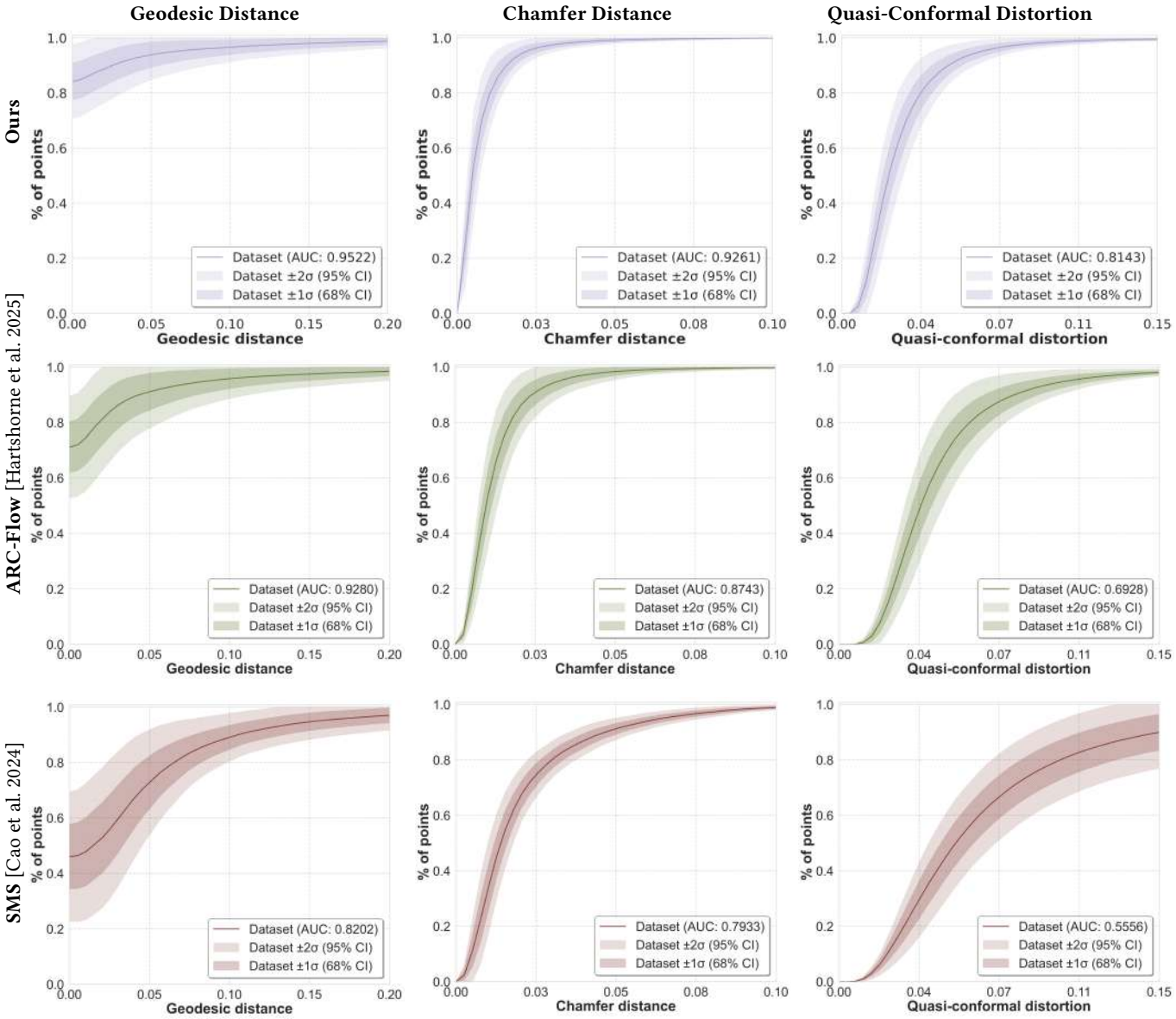

Fig. 23. Interpolation results for SMAL: Ours vs ARC-Flow [Hartshorne et al. 2025] & SMS [Cao et al. 2024]. *Mean and confidence intervals for the three metrics are shown; top row has our results, middle SMS & bottom row ESA.*