NeuralSRNF: Neural Square Root Normal Fields for the Statistical Shape Analysis and Generation of Nonrigid 3D and 4D Objects

Awais Nizamani, Hamid Laga, Guanjin Wang, Farid Boussaid, Senior Member, IEEE, Mohammed Bennamoun, Senior Member, IEEE and Anuj Srivastava, Member, IEEE

***Abstract*—We introduce NeuralSRNF, a novel framework for the statistical shape analysis and generation of genus-zero 3D and 4D objects that undergo nonrigid deformations. Traditional methods rely on complex and computationally expensive nonlinear elastic metrics that measure bending and stretching. Recent advances in elastic shape analysis achieve computational efficiency by mapping input 3D shapes to the space of Square Root Normal Fields (SRNFs) where the $\mathbb{L}^2$ metric approximates the partial elastic metric, significantly facilitating the process of computing geodesics and summary statistics. SRNFs, however, are not invertible, and the numerical algorithms used to map SRNFs back to the original space of surfaces remain computationally very expensive and often lead to approximate results. This paper addresses this fundamental SRNF inversion problem using a novel neural representation, termed NeuralSRNF. Unlike the commonly used numerical SRNF, NeuralSRNF is (1) continuous, and thus resolution-agnostic, enabling full functional shape analysis, (2) more accurate, and (3) computationally more efficient as it can compute inverse SRNF maps along a geodesic path in less than 3 s compared to over 10 min for the numerical SRNF. We demonstrate, using various datasets, the utility and efficiency of the proposed NeuralSRNF in multiple elastic 3D and 4D shape analysis tasks such as geodesic computation, deformation transfer, statistical summaries computation, and 3D shape generation. We show that it outperforms competing methods on most evaluated datasets and metrics by a wide margin in both accuracy and computational efficiency. The source code and additional results are available at https://awaisnizamani16.github.io/awais/NeuralSRNF/.**

***Index Terms*—Square Root Normal Field, Statistical summaries, Shape generation, Elastic metric, Neural representation.**

I. Introduction

Shape is an important geometric property of natural and manmade objects. Understanding differences and similarities between the shapes of objects has many applications in computer vision, augmented reality, biology, and anatomy. Shape analysis focuses on quantifying such differences or similarities, and characterizing shape variability within a population of 3D and 4D objects using probability distributions. Traditional techniques achieve this by treating the shape of an object as a point on a Riemannian shape space equipped with a proper elastic metric that measures bending and stretching. This way, the dissimilarity between two shapes can be defined as the optimal amount of bending and stretching that one needs to apply to one shape in order to align it with the other. State-of-the-art methods that follow this general pipeline exhibit two fundamental limitations. **First**, they treat 3D and 4D shapes as discrete surfaces and thus the metric is not invariant to the quality of the discretization. **Second**, the metric, which is nonlinear, is computationally very expensive to evaluate; thus, it becomes impractical for the statistical analysis of large shape datasets.

Recent advances in elastic shape analysis achieve computational efficiency by mapping 3D shapes to the space of Square Root Normal Fields (SRNFs) [16], [17], [21] where the $\mathbb{L}^2$ metric approximates the partial elastic metric, significantly facilitating the process of computing geodesics and summary statistics. SRNFs, however, are not invertible as there is no analytical expression for the inverse SRNF for arbitrary points in $\mathbb{L}^2$, and the injectivity and surjectivity of the SRNF map remain to be determined. Thus, one cannot transfer geodesics and statistical analysis conducted in the SRNF space back to the original space of surfaces. While numerical algorithms have been used for the SRNF inversion [21], they remain computationally very expensive and often lead to approximate results.

This paper addresses this fundamental SRNF inversion problem using a novel neural representation, termed Neural Square Root Normal Fields (NeuralSRNF). Focusing on genus-0 surfaces, which cover a wide range of shape classes, ranging from human bodies to animals, our key insight is to treat such surfaces as continuous functions parameterized using an MLP. This way, SRNFs become continuous functions. We then propose a novel network architecture that learns how to recover surfaces from their SRNF maps using a Conditional MLP. Unlike the commonly used numerical SRNF [21], NeuralSRNF is continuous. Thus, it is resolution-agnostic, enabling full functional shape analysis. The proposed approach is more accurate and computationally more efficient as it can invert SRNF maps along a geodesic path in less than 3 s compared to over 10 min for the numerical SRNF [21], [29].

Awais Nizamani is with Murdoch University, Perth, Australia. E-mail: Awais.Nizamani@murdoch.edu.au

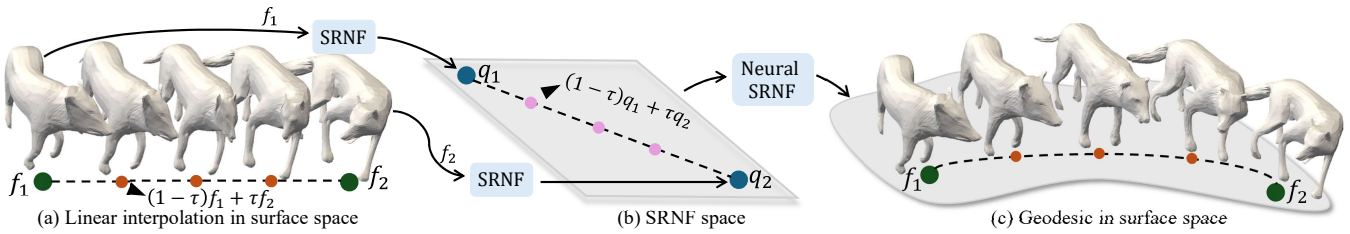
Hamid Laga is with Murdoch University, Perth, Australia. E-mail: H.Laga@murdoch.edu.au.

Guanjin Wang is with Murdoch University, Perth, Australia. E-mail: Guanjin.Wang@murdoch.edu.au.

Farid Boussaid is with The University of Western Australia, Perth, Australia. E-mail: Farid.Boussaid@uwa.edu.au.

Mohammed Bennamoun is with The University of Western Australia, Perth, Australia. E-mail: Mohammed.Bennamoun@uwa.edu.au.

Anuj Srivastava is with The Johns Hopkins University, Maryland, US. E-mail: Anuj.Srivastava@jhu.edu.



(a) Linear interpolation in surface space (b) SRNF space (c) Geodesic in surface space

Fig. 1. Illustration of the proposed NeuralSRNF framework. Given surfaces f_i, we first map them to the SRNF space using $q_i = Q(f_i)$. We then compute geodesics between pairs q_1 and q_2 in the SRNF space, which has an Euclidean structure, and use NeuralSRNF to map their SRNFs back to the surface space. We show **(a)** linear interpolation in the original space of surfaces, **(b)** linear interpolation in the SRNF space, and **(c)** the geodesic computed by SRNF inversion of the linear path in the SRNF space.

In summary, the main contributions are as follows:

- We treat genus-0 surfaces as continuous functions parameterized using MLPs. This provides a resolution-agnostic representation, enabling full functional shape analysis.
- We propose a novel network architecture that provides a practical solution to the theoretical SRNF problem by learning to recover a valid surface from its SRNF representation. We achieve this using a conditional MLP defined using DINOv2 encoded features and Feature-wise Linear Modulation (FiLM). Compared to [21], ours is more than 100 times faster and more accurate.
- We develop a set of computational tools for geodesic computation, deformation transfer, statistical summarization of 3D and 4D shape collections, and 3D shape generation.

We demonstrate, using 3D and 4D human, face, and animal datasets, that the proposed NeuralSRNF framework outperforms the state of the art by a wide margin in terms of accuracy and computational efficiency.

II. Related work

Statistical 3D shape analysis methods treat the shape of a 3D object as a point in a high-dimensional shape space equipped with a metric that measures physical deformations; see [20] for a detailed review of the state-of-the-art. Early approaches that are based on active shape models [9] and morphable models [3], [11] assume that the shape space is Euclidean and use the $\mathbb{L}^2$ metric to compute geodesics and perform statistical analysis using Principal Component Analysis (PCA). These methods, however, are only suitable for rigid objects or objects that undergo small elastic deformations since the $\mathbb{L}^2$ metric does not quantify elastic deformations. Later, more expressive metrics have been introduced to capture and quantify large nonrigid deformations. For example, Kilian *et al.* [18] use a metric that penalizes deviations from rigidity and isometry. Drawing from the elasticity theory in physics, Wirth *et al.* [42] and Zhang *et al.* [45] modeled stretching using the Cauchy-Green strain tensor and bending via differences in shape operators. Windheuser *et al.* [41] measured bending through variations in the mean curvature, while Heeren *et al.* [15] used differences in the second fundamental form for bending and changes in the first fundamental form for stretching.

While being highly expressive, these physically motivated metrics are computationally intensive since geodesic computation and statistical summarization all require solving nonlinear optimization problems. Jermyn *et al.* [16], [17] introduced a more efficient approach by showing that the $\mathbb{L}^2$ metric in the space of SRNFs is equivalent to a partial elastic metric defined as a weighted sum of bending and stretching. This significantly reduces the computational complexity: instead of working in the original space, one can map all the input shapes to the SRNF space, perform the analysis tasks on that space using the $\mathbb{L}^2$ metric, and finally map the results back to the original space of shapes. The main theoretical limitation, however, is that this inverse mapping has no analytical expression, and its injectivity and surjectivity are yet to be established. Laga *et al.* [21] addressed this by using a numerical approach. The method, however, is computationally expensive, and results in approximate solutions. This paper addresses this inversion problem using a learning approach, leading to a significantly faster and more accurate solution than the state-of-the-art.

Note that statistical shape analysis methods require spatially-registered surfaces. Some methods address this jointly with the statistical analysis [16], [17], [21]. Others assume that the registration is pre-established, *e.g.,* by using functional maps [31], [32], [13], [2], [27], [7], [23], [6].

Neural representations [33], [28], [39] leverage neural networks to represent the geometry and appearance of 3D shapes in a resolution-agnostic manner. While they were primarily intended for 3D and 4D reconstruction, and novel view synthesis, some recent papers have begun to explore their potential in statistical shape analysis. In particular, Williamson *et al.* [40] showed how to compute differential properties of neural surfaces and perform geometry processing tasks on spherical neural representations without upfront discretization. Sang *et al.* [37], [38] used neural implicit functions to learn how to deform a source 3D surface to a target one. Their method, however, requires training for every pair of 3D objects and is therefore computationally very expensive. Nizamani *et al.* [29] introduced dynamic neural surface representations for the statistical analysis of 4D shapes. Some recent approaches embed surfaces into a learned latent space and compute geodesics within latent space using ARAP regularizations [12], [10], [44], Laplace-Beltrami operators [22], and path straightening [14]. These geometric constraints are also computationally expensive.

This paper uses the surface-based neural representation of Nizamani *et al.* [29] but focuses on addressing the SRNF inversion problem using a novel neural representation termed NeuralSRNF. This leads to more accurate results while being

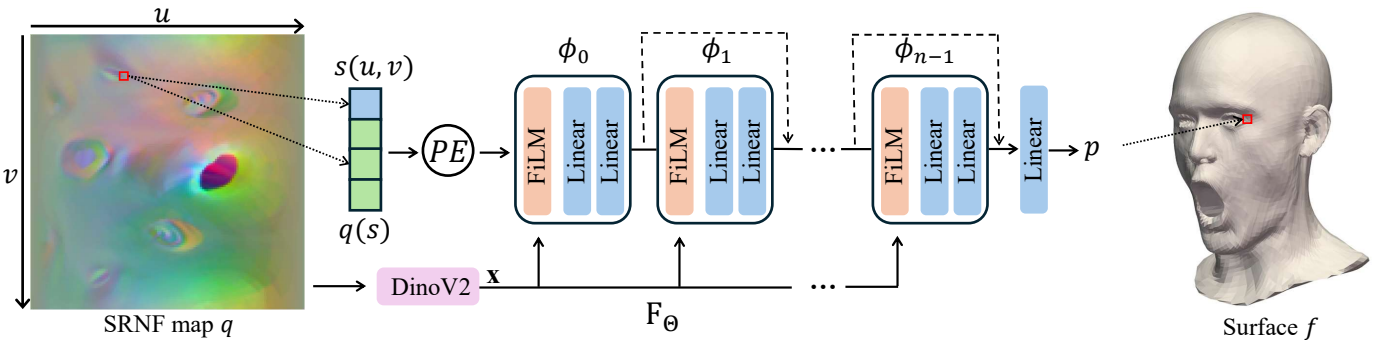


Fig. 2. Network architecture of NeuralSRNF, which takes an SRNF map q and recovers a surface f whose SRNF map is as close as possible to q.

significantly faster (up to 100 times) compared with the state-of-the-art methods.

III. Method

We treat the shape of a closed genus-0 surface $f : \Omega = \mathbb{S}^2 \to \mathbb{R}^3$, after normalization for translation and scaling, as an element of a pre-shape space $\mathcal{F}$. To measure the dissimilarity between two surfaces f_1 and f_2 in $\mathcal{F}$, and subsequently compute geodesic paths, we equip $\mathcal{F}$ with a Riemannian elastic metric that quantifies the amount of bending and stretching one needs to apply to f_1 in order to align it onto f_2. This approach, however, is computationally expensive, making it hard to use in downstream tasks such as the computation of geodesics and statistical summaries. This computational cost stems from two fundamental limitations. **First**, the computational complexity of the metric scales exponentially with the resolution of the discretization of the domain Ω. **Second**, the metric is highly nonlinear. Thus, computing geodesics and summary statistics requires solving complex nonlinear optimizations. To address the computational complexity issue, Jermyn *et al.* [16] map 3D shapes to the space $\mathcal{Q}$ of SRNFs where the $\mathbb{L}^2$ metric in $\mathcal{Q}$ is equivalent to the partial elastic metric in $\mathcal{F}$. This significantly simplifies downstream tasks since these can be performed in the SRNF space, using the $\mathbb{L}^2$ metric, and then mapped back to the space of surfaces for visualization. The fundamental problem, however, is that SRNFs are not invertible, *i.e.,* the mapping from the SRNF space to the space of surfaces has no analytical form. Laga *et al.* [21] approximate this inversion using a numerical approach but it is very slow and often leads to inaccurate results.

This paper jointly addresses these two problems by introducing a novel neural representation termed NeuralSRNF, which represents surfaces $f \in \mathcal{F}$ in a resolution-agnostic manner, enabling full functional shape analysis, and learns how to invert SRNF maps in a computationally efficient manner while achieving significantly better accuracy than numerical methods (Section III-A). We will show how to use the proposed NeuralSRNF in computing geodesics (Section III-B), deformation transfer (Section III-C), and summary statistics and shape generation (Section III-D).

A. The SRNF inversion problem

Given a surface $f \in \mathcal{F}$, its SRNF $q = Q(f)$ is essentially the normal field on the surface scaled by the square root of the norm of the normals. In other words,

$$Q : \mathcal{F} \to \mathcal{Q} \text{ such that } Q(f)(s) = \frac{\mathbf{n}(s)}{\| \mathbf{n}(s) \|^{\frac{1}{2}}}. \quad (1)$$

Here, $\mathcal{Q}$ is the space of SRNFs, $\mathbf{n} = \frac{\partial f}{\partial u} \times \frac{\partial f}{\partial v}$ is the normal vector field to the surface, $\hat{\mathbf{n}}$ is the unit normal field, and $s = (u, v) \in \mathbb{S}^2$. Note that the SRNF is physically motivated since **(1)** changes in the direction of the normal vector $\mathbf{n}(s)$ at a given point s on the surface induce local bending, while **(2)** changes in the norm of a normal vector, denoted by $r(s) = \|\mathbf{n}(s)\|$, which is the square root of the determinant of the first fundamental form g of the surface at s, result in local stretching. Jermyn *et al.* [16] showed that the partial elastic metric $d_{\mathcal{F}}$ that quantifies the weighted sum of bending and stretching between two surfaces f_1 and f_2 in $\mathcal{F}$ is equivalent to the $\mathbb{L}^2$ metric in the space $\mathcal{Q}$ of SRNFs, *i.e.,* ,

$$d_{\mathcal{F}}(f_1, f_2) \equiv \| q_1 - q_2 \|, \text{ where } q_i = Q(f_i). \quad (2)$$

With this formulation, the geodesic, or the shortest path with respect to the metric $d_{\mathcal{F}}$, between f_1 and f_2 is equivalent to the straight line between q_1 and q_2. Thus, working in the SRNF space is computationally very attractive. However, the following questions about the SRNF inversion problem, *i.e.,* how to map results from the SRNF space back to the original space of surfaces, remain unaddressed: Given an SRNF q, or equivalently a unit normal field $\hat{\mathbf{n}} : \mathbb{S}^2 \to \mathbb{S}^2$ and a local area element $r = \| \mathbf{n} \| : \mathbb{S}^2 \to \mathbb{R}_{>0}$,

1) Under what conditions does the pair $(\hat{\mathbf{n}}, r)$ correspond to a valid surface $f : \mathbb{S}^2 \to \mathbb{R}^3$?
2) Does the differential equation $\mathbf{n} = |q|q$ admit a solution, and is it unique up to translation?

In the **first** case, if we are given $(\hat{\mathbf{n}}, g)$, it is known that the representation is injective (up to isometries of $\mathbb{R}^3$), *i.e.,* the original surface can be reconstructed up to translation and rotation [1]. This is not the case with the SRNF, or equivalently $(\hat{\mathbf{n}}, r)$, where r is unable to fully measure the stretching. The **second** case considers the existence of a unique global solution for the inverse of the SRNF q, which clearly does not always exist. For a small patch $q(s)$, one can reconstruct the surface $f(s)$, but it may not be unique. Also, this is not sufficient for shape analysis as one requires a global solution for an entire surface.

Dataset	Spherical resolution	Numerical SRNF [21] Mean	Std	Median	Time (sec)	NeuralSRNF (Ours) Mean	Std	Median	Time (sec)
COMA [36]	128×128	0.058×10^{-3}	0.057×10^{-3}	0.041×10^{-3}	4.73	$\mathbf{0.002 \times 10^{-3}}$	$\mathbf{0.001 \times 10^{-3}}$	$\mathbf{0.002 \times 10^{-3}}$	**0.10**
CAPE [26]	512×512	5.433×10^{-3}	5.608×10^{-3}	3.275×10^{-3}	67.22	$\mathbf{0.336 \times 10^{-3}}$	$\mathbf{0.361 \times 10^{-3}}$	$\mathbf{0.242 \times 10^{-3}}$	**0.65**
DFAUST [5]	512×512	2.655×10^{-3}	1.885×10^{-3}	2.194×10^{-3}	69.10	$\mathbf{0.043 \times 10^{-3}}$	$\mathbf{0.191 \times 10^{-3}}$	$\mathbf{0.036 \times 10^{-3}}$	**0.60**
Animal (w tail) [43]	128×128	2.728×10^{-3}	1.861×10^{-3}	2.350×10^{-3}	15.93	$\mathbf{0.078 \times 10^{-3}}$	$\mathbf{0.112 \times 10^{-3}}$	$\mathbf{0.042 \times 10^{-3}}$	**0.11**
Animal (w/o tail) [43]	128×128	1.208×10^{-3}	1.192×10^{-3}	0.812×10^{-3}	12.77	$\mathbf{0.030 \times 10^{-3}}$	$\mathbf{0.074 \times 10^{-3}}$	$\mathbf{0.012 \times 10^{-3}}$	**0.11**

TABLE I

COMPARISON OF THE PROPOSED NEURALSRNF INVERSION WITH NUMERICAL SRNF [21]. THE EVALUATION IS PERFORMED ON 100 TEST SAMPLES FROM FIVE DATASETS. WE MEASURE THE MEAN SQUARED $\mathbf{L}^2$ DISTANCE AND REPORT THE MEAN, STANDARD DEVIATION, AND MEDIAN INVERSION ERRORS, TOGETHER WITH THE AVERAGE COMPUTATION TIME FOR BOTH METHODS.

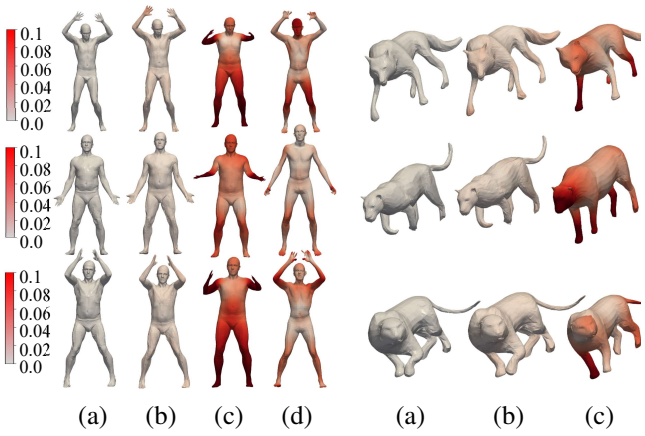


Fig. 3. Accuracy of the SRNF inversion on 3D humans and animals. **(a)** Ground-truth surfaces. **(b)** Reconstructed surfaces using our NeuralSRNF method. **(c)** Reconstructed surfaces using the numerical SRNF [21]. **(d)** Reconstructed surfaces using the spectral autoencoder [22]. The pointwise error is shown as a heatmap for (b), (c), and (d).

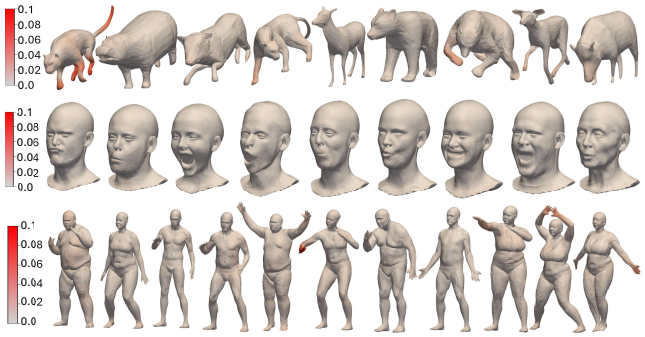


Fig. 4. We show the qualitative results of the proposed NeuralSRNF inversion on animals, faces, and humans. The error is shown as a heatmap computed as the pointwise distance between the ground-truth and reconstructed surface using our NeuralSRNF method.

1) Neural Square Root Normal Fields: In the absence of a theoretical solution, we reformulate the SRNF inversion problem as an optimization problem, *i.e.,* given an SRNF $q \in \mathcal{Q}$, we seek to find a surface $f^* \in \mathcal{F}$ whose SRNF is as close as possible to q:

$$f^* = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \|q - Q(f)\|^2. \tag{3}$$

When dealing with non-degenerate closed surfaces, Laga *et al.* [21] solved this optimization using a numerical approach. However, the approach is slow and requires a multiresolution and multiscale representation of the surfaces, in addition to a careful initialization of the optimization procedure. These significantly limit its applicability.

In this paper, we adopt a different approach. Instead of seeking to solve the optimization problem of Eqn. (3), we propose to explicitly learn the inverse mapping that maps the subset of $\mathcal{Q}$ that corresponds to valid surfaces to their corresponding surfaces in $\mathcal{F}$. We treat this inverse mapping as a set of functions F_Θ, parameterized by the set of parameters Θ, of the form: $F_\Theta : (s, q(s); \mathbf{x}) \rightarrow f(s)$. They take a point $s \in \mathbb{S}^2$ and its corresponding SRNF $q(s)$, and find its corresponding point $f(s)$ on the surface. By conditioning F_Θ on the shape-specific latent code $\mathbf{x}$, one can represent many shapes with a single function.

Formally, we represent F_Θ using an MLP, conditioned on the latent code $\mathbf{x}$, and learn its parameters Θ from input SRNFs $\{q_1, \ldots, q_n\}$ paired with their corresponding ground-truth surfaces $\{f_1, \ldots, f_n\}$. The MLP is composed of eight residual blocks, each containing two fully connected layers with 1024 hidden units, and a FiLM conditioning layer [34] to allow the network to adapt its intermediate representations in each residual block, similar to the feature modulation strategy used in pi-GAN [8]. Instead of using the ReLU activation function, which results in sharp edges, we employ the SoftPlus activation [46] to ensure a smooth, continuous, and differentiable surface representation. Lastly, a linear projection layer maps the network output to a 3D point. Figure 2 summarizes this architecture.

Inspired by NeRF [28], we apply positional encoding to the SRNF point $q(s)$ and to the coordinates s. To enable the network to invert any SRNF map, we condition it on the shape latent code $\mathbf{x}$. Unlike DeepSDF [33], which learns the latent code in an auto-decoding fashion, our latent codes are pre-computed from the SRNF map of each shape using a pretrained DINOv2 [30] encoder. We found that this approach leads to faster computation time as the latent codes do not require finetuning during inference. The Supplementary Material provides a detailed ablation study of the network architecture, including the role of FiLM-based conditioning and the importance of DINOv2 features as latent codes.

2) Training and losses: Let $\{q_1, \ldots, q_n\}$ be a set of SRNFs, $\{f_1, \ldots, f_n\}$ their corresponding spherically-parameterized ground-truth surfaces, and B the batch size. Each SRNF q_j is first encoded using DINOv2 encoder to obtain a latent code $\mathbf{x}_j$. We follow the DINOv2 input format and normalize q_j to keep it within the range of $[0, 1]$ (the normalization is only applied to generate the latent code $\mathbf{x}_j$). To create a continuous representation, we sample K spherical coordinates

$\{s_i \in \mathbb{S}^2\}_{i=1}^K$. For each spherical coordinate s_i, we obtain its corresponding points $p_{ij} = f_j(s_i)$ on the input surfaces, and their respective SRNF values $q_j(s_i)$. Next, we train the MLP F_Θ by minimizing the mean squared error (MSE) between the ground-truth point p_{ij} and the predicted point p^*_{ij}:

$$L_{\text{MSE}} = \frac{1}{B}\frac{1}{K}\sum_{j=1}^{B}\sum_{i=1}^{K} \parallel F_\Theta(s_i, q_j(s_i), \mathbf{x}_j) - p_{ij} \parallel^2 . \quad (4)$$

One advantage of the NeuralSRNF is that it is a continuous and resolution-agnostic function. Thus, one can compute all its differential properties, including the surface normals, using automatic differentiation.

B. Geodesics

We define a path in $\mathcal{F}$ between two surfaces f_1 and f_2 as a curve $\alpha_{f_1 \to f_2} : [0,1] \to \mathcal{F}$ such that $\alpha(0) = f_1$, $\alpha(1) = f_2$, and $\alpha(\tau) \in \mathcal{F}$ for all $\tau \in [0,1]$. The geodesic between the two surfaces, *i.e.,* the minimum amount of bending and stretching one needs to apply to f_1 in order to align it onto f_2, is then the shortest path under the elastic metric between the two surfaces. However, finding this shortest path using algorithms such as path straightening requires solving a highly nonlinear optimization problem since the metric is nonlinear. To address this problem in a computationally efficient manner, we first map the two surfaces to the SRNF space, resulting in $q_1 = Q(f_1)$ and $q_2 = Q(f_2)$. Since the $\mathbb{L}^2$ metric in $\mathcal{Q}$ is equivalent to the elastic metric in the original space $\mathcal{F}$ of surfaces, one only needs to compute the straight line α_q between q_1 and q_2:

$$\alpha_q(\tau) = (1-\tau)q_1 + \tau q_2, \quad \tau \in [0,1]. \quad (5)$$

We then map $\alpha_q(\tau) \in \mathcal{Q}$ for all $\tau \in [0,1]$ back to the original space $\mathcal{F}$ of surfaces using the proposed NeuralSRNF, *i.e.,* $\alpha_f(\tau) = Q^{-1}(\alpha_q(\tau))$. Figure 1-(c) shows an example of such a geodesic path between f_1 and f_2.

C. Deformation transfer

The proposed framework can be used to transfer deformations across surfaces. This problem can be defined as follows: we are given a source deformation $f_1 \to h_1$, *i.e.,* a surface f_1 that deforms into the surface h_1, and we would like to deform a surface f_2 so that it assumes the same pose as h_1, obtaining another surface h_2. Traditionally, this is solved using parallel transport, *i.e.,* we first compute the geodesic between f_1 and h_1, and parallel transport it to f_2. This involves computing geodesics in $\mathcal{F}$ and parallel-transporting them, thus it is computationally expensive. With the proposed NeuralSRNF, this parallel transport problem can be solved in a very efficient manner. We first map the surfaces to the SRNF space $\mathcal{Q}$, which has an $\mathbb{L}^2$ structure. Then, the SRNF of h_2 is given by:

$$Q(h_2) = Q(f_2) + (Q(h_1) - Q(f_1)). \quad (6)$$

Here, $(Q(h_1) - Q(f_1))$ represents the deformation field v, which is added to $Q(f_2)$ to obtain the deformed target surface $Q(h_2)$. $Q(h_2)$ can then be inverted back to the space of surfaces using the proposed NeuralSRNF inversion.



(a) Before registration. **(b)** After registration.



(c) Geodesic after registration.

Fig. 5. Joint registration and geodesic computation in the SRNF space.



(a) Linear path.



(b) Geodesic using numerical SRNF [21] (118 s).



(c) Geodesic using our NeuralSRNF (2 s).

Fig. 6. Examples of geodesics generated with our method and the state of the art. Observe that the intermediate surfaces of our approach are more natural.

Note also that one can define a path $\alpha_q(\tau) = Q(f_2) + \tau(Q(h_1) - Q(f_1))$. When $\tau \in [0,1]$, it is equivalent to transporting the entire geodesic between f_1 and h_1 to f_2. For $\tau > 1.0$, this is equivalent to extrapolating the deformation beyond the given one.

D. Summary statistics and shape generation

Given a collection of 3D shapes $\{f_1, \ldots, f_n\}$, the goal of statistical analysis is to compute the mean shape μ and the modes of variation, characterize the shape variability in the collection using a probability distribution, *e.g.,* a multivariate Gaussian, and finally sample from the distribution to generate novel 3D shapes. Since the pre-shape space $\mathcal{F}$ is equipped with a nonlinear metric $d_{\mathcal{F}}$, the mean shape can be computed by finding the shape μ that is as close as possible, with respect



(a) Geodesic using numerical method [21] (118 s).



(b) Geodesic using our NeuralSRNF (2 s).

Fig. 7. Geodesics comparison of numerical SRNF [21] with the proposed NeuralSRNF. Notice that the head and tail movement in **(b)** looks more natural compared to **(a)**.

to the metric $d_{\mathcal{F}}$, to all the input samples:

$$\mu = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \sum_{i=1}^{n} d_{\mathcal{F}}\left(f, f_i\right)^2. \tag{7}$$

This is computationally very expensive to solve as it involves shooting geodesics from f to every sample f_i a number of times (depending on the number of iterations required by the optimization algorithm). We address this computational issue by first mapping the input surfaces to their SRNF representations $\{q_1, \ldots, q_n\}$, performing Principal Component Analysis (PCA) in the SRNF space, and then mapping the computed statistical summaries back to the original space using the proposed NeuralSRNF framework. For instance, the mean μ_q in the SRNF space is given by $\mu_q = \frac{1}{n}\sum_{i=1}^{n} q_i$. Similarly, the modes of variation are the eigenvectors of the covariance matrix $M = \frac{1}{n-1}\sum_{i=1}^{n}(q_i - \mu_q)(q_i - \mu_q)^\top$. Let us denote by Λ_i and λ_i the $i-$th leading eigenvector and its corresponding eigenvalue. With this, the principal geodesic path G_i that corresponds to the principal component Λ_i is given by:

$$G_i(\tau) = Q^{-1}(\mu_q + \tau\Lambda_i), \quad \tau \in \mathbb{R}. \tag{8}$$

Here, Q^{-1} refers to the SRNF inversion using the proposed NeuralSRNF. Also, with this formulation, any surface $f \in \mathcal{F}$ can be written as:

$$f = Q^{-1}\left(\mu_q + \sum_{i=1}^{k} a_i \sqrt{\lambda_i}\Lambda_i\right), \text{ with } a_i \in \mathbb{R}. \tag{9}$$

Here, k is the number of leading eigenvectors. Thus, one can generate a novel 3D surface by first sampling random values $\{a_i\}_{i=1}^{k}$ and using Eqn. (9) to generate a new surface f. To further control the quality of the generated surface, one can enforce the weights $\{a_i\}_{i=1}^{k}$ to be within a certain range, *e.g.,* $[-1, 1]$ to ensure that the generated surfaces are within one standard deviation around the mean.

FAUST [4]	**Mean ↓**	**Std**	**Median**
LIMP [10]	0.239	0.192	0.179
NeuroMorph [12]	0.167	0.162	0.112
Ours	**0.087**	**0.075**	**0.081**
DFAUST [5]	**Mean ↓**	**Std**	**Median**
SAE [22]	0.199	0.166	0.153
Ours	**0.028**	**0.028**	**0.019**

TABLE II

WE MEASURE THE PAIRWISE GEODESIC ERROR ON FIVE PAIRS FROM THE FAUST [4] AND DFAUST [5] DATASET AS THE $\mathbb{L}^1$ DISTANCE BETWEEN SOURCE AND INTERMEDIATE SHAPE.

	FAUST	DFAUST	COMA
LIMP (Geodesic error)	3.82×10^{-3}	$\mathbf{2.89 \times 10^{-4}}$	7.51×10^{-4}
Ours (Geodesic error)	$\mathbf{2.96 \times 10^{-3}}$	1.14×10^{-3}	$\mathbf{6.98 \times 10^{-4}}$

TABLE III

WE MEASURE THE GEODESIC ERROR ON 100 PAIRS FROM THE FAUST [4], DFAUST [5], AND COMA [36] DATASETS USING THE LIMP [10] EVALUATION METRIC.



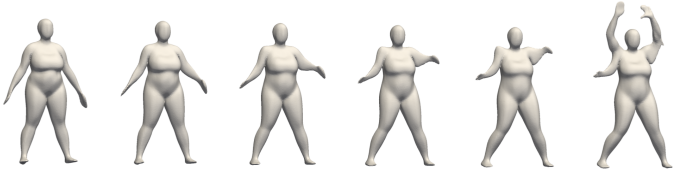
(a) Linear path.



(b) Geodesic using numerical SRNF [21] (818 s).



(c) Geodesic using implicit neural surface [37] (authors reported a time of 20 min; time taken on our machine is 2 h).



(d) Geodesic using 4Deform [38] (authors reported a time of 7 min; time taken on our machine is 1 h).

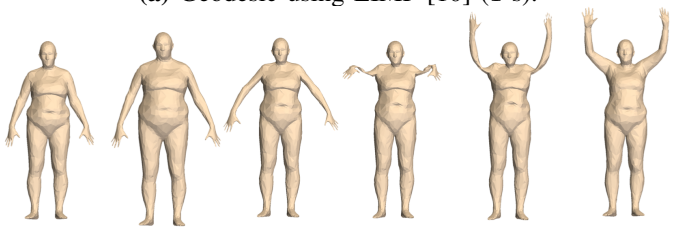


(e) Geodesic using our NeuralSRNF inversion (3 s).

Fig. 8. Comparison of the geodesic computed using the proposed Neural-SRNF with state-of-the-art methods.



(a) Geodesic using LIMP [10] (1 s).



(b) Geodesic using NeuroMorph [12] (1 s).



(c) Geodesic using our NeuralSRNF inversion (3 s).

Fig. 9. Examples of geodesics generated using the discrete methods **(a)** LIMP [10] and **(b)** NeuroMorph [12] and compared with **(c)** our continuous NeuralSRNF.

IV. Results

Datasets. We have tested the NeuralSRNF framework on a wide range of 3D and 4D datasets covering faces, humans, and animal surfaces from MPI COMA [36], MPI FAUST [4], MPI DFAUST [5], MPI CAPE [26], and Animal3D [43] datasets. We spherically parameterize all the 3D surfaces using Kurtek *et al.*'s implementation [19] of the approach of Hoppe and Praun [35].

Implementation details. The NeuralSRNF framework was trained on an NVIDIA GeForce RTX 4090 GPU with an Intel Core i9 2.4 GHz processor. The proposed SRNF inversion network was optimized using AdamW [24] with a CosineAnnealing [25] learning rate scheduler starting from $1e^{-4}$ and gradually reduced to $1e^{-6}$. We train the network for $1,000$ epochs with a batch size of 16, which takes around 8 to 24 hours to train, depending on the complexity of the dataset. For each batch, we sample a 16×16 patch of 3D points to update the network parameters. We follow this for the human, animal, and face datasets.

A. SRNF inversion

We first evaluate, quantitatively and qualitatively, the accuracy and performance of the SRNF inversion procedure proposed in this paper and compare it against the numerical inversion procedure proposed by Laga *et al.* [21]. For this experiment, we use test surfaces from the COMA, CAPE, DFAUST, Animal with tail, and Animal without tail datasets. For each dataset, we randomly sample 100 test surfaces and convert each surface to the SRNF representation. We then apply the inversion frameworks and report the average of the mean squared distance $\mathbb{L}^2$ between the ground-truth and the recovered surfaces using our method and the numerical SRNF method [21].

Table I reports the mean, median, and standard deviation of the inversion error and inference time for the five datasets. As one can see, our inversion framework achieves superior performance on all datasets across the three metrics and is able to perform SRNF inversion within 0.10 to 0.65 s per surface compared to 4.73 to 69 s for [21]. Thus, our proposed framework is 50 to 100 times faster than the traditional numerical SRNF [21] while achieving significantly better accuracy.

Figure 3 demonstrates the qualitative results of the inversion on 3D humans and animals. In this figure, we display **(a)** the ground-truth 3D surface, **(b)** the inversion of SRNF computed using the proposed NeuralSRNF, **(c)** the inversion of SRNF computed using the numerical inversion [21], and **(d)** the inversion of spectral basis computed using the SAE [22]. We also plot the reconstruction error, *i.e.,* the pointwise error between the original surface and the reconstructed surface obtained by inversion. As shown in Figure 3-**(b)**, the inversion error of our method is signifcantly lower than that of the state of the art across all regions of the surface, including at the extremities (such as fingers) and non-convex regions.

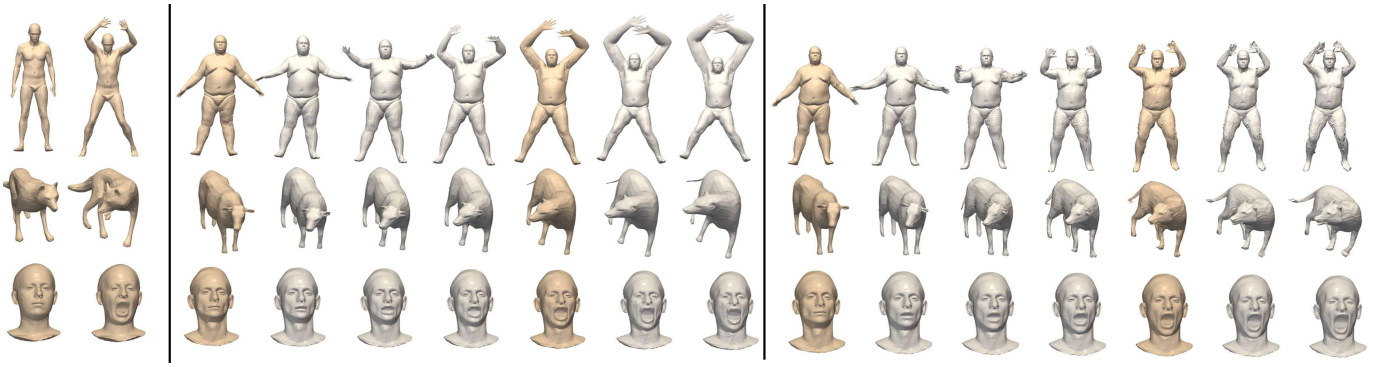
Figure 4 shows additional qualitative results of the proposed NeuralSRNF inversion on 3D models of human bodies, faces, and animals. We show the error as a heatmap computed using the approach discussed in Figure 3.

B. Geodesics

We evaluate quantitatively and qualitatively the quality of the geodesics and the computation time of the proposed method against state-of-the-art methods.

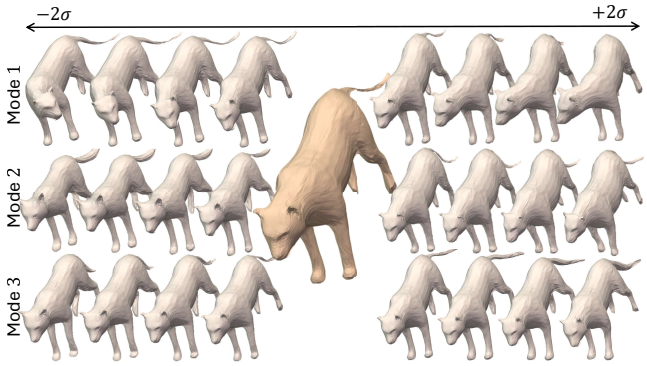
Table II reports the pairwise geodesic error. For each pair of source and target 3D shapes, we first compute a geodesic path using LIMP [10], NeuroMorph[12], SAE[22], and the proposed NeuralSRNF. We then take each of the 3D shapes along the geodesic, sample 1000 points, and compute the pairwise geodesic distances (*i.e.,* distances along the shape surface) between these points. We then define the error as the $\mathbb{L}^1$ distance between the pairwise geodesics of the source surface and the intermediate surfaces along the geodesic path. A lower error indicates a better geodesic path. As shown in the table, our method achieves lower mean, median, and standard deviation errors compared with the state-of-the-art methods. For a fair comparison, we evaluate each method on the datasets supported by the authors' released code. LIMP [10] and NeuroMorph [12] implementations provide training code for FAUST [4], while SAE [22] provides code for DFAUST [5]. Therefore, we compare each method only on the supported datasets.

Table III reports the geodesic error using the evaluation metric proposed in LIMP [10] on 100 pairs from the FAUST [4], DFAUST [5], and COMA [36] datasets. Since LIMP [10] provides evaluation code only for FAUST [4], we use the geodesic error results reported in its paper (Table 1) for this

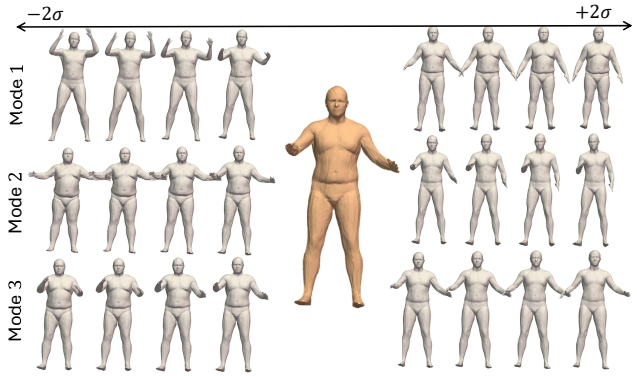


(a) Source. (b) Using numerical SRNF [21]. (c) Using the proposed method.

Fig. 10. Examples of deformation transfer for humans, animals, and faces. **(a)** The source deformation, **(b)** deformation transfer by linear interpolation and extrapolation in the space of SRNF $\mathcal{Q}$ followed by SRNF inversion of $\mathcal{F}$ using numerical SRNF [21], and **(c)** using the proposed NeuralSRNF.



(a) Summary statistics of animals.



(b) Summary statistics of humans.

Fig. 11. Summary statistics computed with the proposed NeuralSRNF on the animal and human datasets.

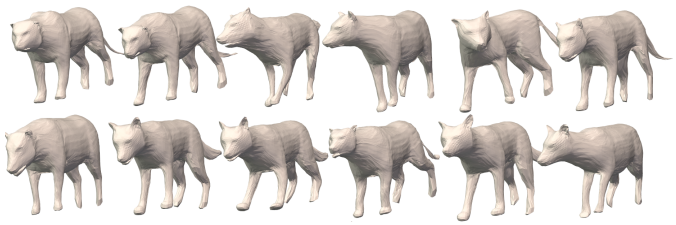
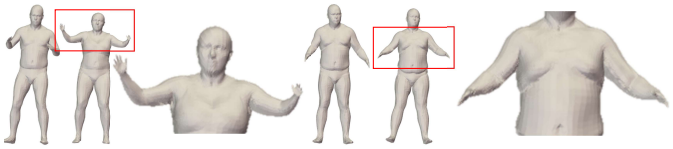
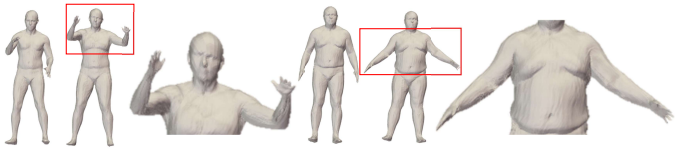


Fig. 12. 12 randomly generated 3D animals with NeuralSRNF.



(a) Randomly generated humans using numerical SRNF [21].



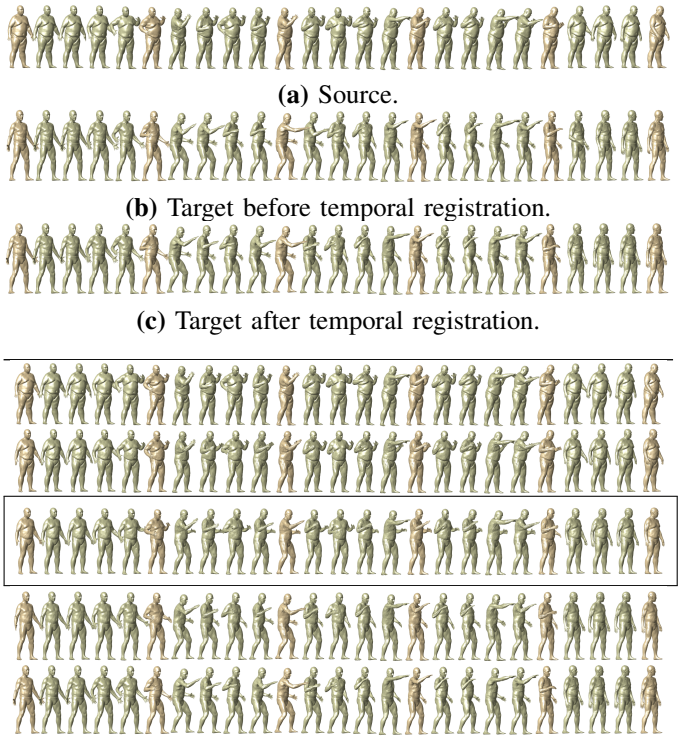
(b) Randomly generated humans using our NeuralSRNF.

Fig. 13. Generated 3D shapes with two close-up views using **(a)** numerical SRNF [21] and **(b)** our NeuralSRNF.

comparison. We follow a similar protocol to LIMP [10]; we compute geodesic error over both training and test samples, allowing us to evaluate the geodesic error on 100 pairs. Note that, during training, LIMP [10] directly optimizes this geodesic error. Although our method is not optimized on this geodesic error, it is able to achieve a lower error on FAUST [4] and COMA [36] than LIMP [10].

Figure 5 shows an example of a geodesic path between two animals computed with our method after registration. Originally, the 3D shapes come with ground-truth registration. Thus, we first randomly reparameterize the lion f_2 to simulate the case where they are not in spatial correspondence; see Figure 5-**(a)**. To bring the surfaces in correspondence, we use the DSNS method [29] to estimate the spatial reparameterization of f_2 and align it to f_1 using the $\mathbb{L}^2$ metric in the SRNF space. Figure 5-**(b)** shows the surfaces after registration. We then compute the geodesic between the two registered surfaces in the SRNF space and reconstruct the surfaces using the proposed NeuralSRNF. Figure 5-**(c)** shows the geodesic path from the top-view.

Figure 6 compares the quality of the geodesic shown in Figure 5 (shown from a front view) against linear interpolation in the original space $\mathcal{F}$ of surfaces (*i.e.,* $(1-t)f_1+tf_2, t \in [0,1]$)



(a) Source.

(b) Target before temporal registration.

(c) Target after temporal registration.

Fig. 14. Example of the temporal registration and geodesic generated after temporal alignment of source and target 4D surfaces using the proposed NeuralSRNF framework. The mean 4D surface is highlighted with a black rectangle. Note that the geodesics look more natural and are generated within 15 s, which is 180 times faster than the numerical SRNF [21]; see the Supplementary Material for additional results.

and the geodesic in $\mathcal{Q}$ but computed using the numerical SRNF inversion procedure of [21]. From this experiment, we can clearly see that the linear interpolation results have unnatural shrinkage, especially in parts such as the head, which undergoes substantial bending. This is expected since linear interpolation assumes that the surfaces are elements of a Euclidean shape space equipped with the $\mathbb{L}^2$ metric, which is not the case for 3D shapes. The geodesic computed using the numerical method of Laga *et al.* [21] shows improvement in quality but still exhibits unnatural shrinkage along the geodesic path. On the other hand, the geodesic computed using the proposed NeuralSRNF is natural: observe, for example, how the head of the animal in Figure 6-**(c)** bends naturally, instead of shrinking as in Figure 6-**(c)**. Figure 7 shows an additional example rendered from the top-view. Here we can see that the tail movement looks more natural between the two foxes in Figure 7-**(b)**, whereas the numerical method [21] in Figure 7-**(a)** exhibits unnatural shrinkage.

Figures 8 and 9 compare the quality of the geodesics between 3D humans computed using linear interpolation, numerical SRNF inversion [21], LIMP [10], NeuroMorph [12], 4Deform [38], the implicit neural surface approach of Sang *et al.* [37], and with the proposed NeuralSRNF. In this experiment, we considered the geodesic failure case reported in [37]. We can clearly see that, in the case of our method (Figures 8-**(e)** and 9-**(c)**), the arm bends and stretches naturally, whereas previous methods (Figures 8-**(a)**, **(b)**, **(c)**, **(d)**, and Figures 9-**(a)**, **(b)**) fail to recover plausible intermediate geodesics. The Supplementary Material provides additional comparisons.

In terms of computation time, our approach computes the geodesic of Figure 6 in 2 s, whereas the numerical method [21] takes 118 s. As a result, our method achieves an approximately 50-fold speedup. In the case of Figure 8 and 9, our approach takes 3 s compared to 7 min for 4Deform [38], 20 min for the neural implicit surface [37], 1 sec for the LIMP [10], and 1 sec for NeuroMorph [12]. LIMP [10] and NeuroMorph [12], however, are discrete approaches and reconstruct meshes on a fixed set of vertices, limiting reconstruction quality, whereas our method learns a continuous and resolution-agnostic representation that results in higher quality meshes and does not require mesh connectivity. Note that implicit methods [37], [38] require training for each pair of shapes, whereas our approach is trained on the collection of 3D shapes and only requires inference to compute geodesics.

We compare our method qualitatively and quantitatively with the most closely related approaches of 3D geodesics. However, we could not include [44] in this comparison as no official implementation, pretrained model, or generated outputs were publicly available at the time of submission[1].

C. Deformation transfer

Figure 10 shows examples of deformation transfer across human, animal, and facial shapes obtained using the approach presented in Section III-C. The source deformations, *i.e.,* the surface f_1 that deforms into the surface h_1, are shown in Figure 10-(a). Figures 10-(b) and (c) show the transfer of these deformations to another surface using linear extrapolation in the SRNF space, starting from $Q(f_2)$ to $\alpha_q(\tau) = Q(f_2) + \tau Q(v)$. With a slight abuse of notation, let $Q(v) = Q(h_1) - Q(f_1)$. We then map them to the space of surfaces $\mathcal{F}$ using the numerical SRNF inversion method [21] (Figure 10-(b)) and the proposed NeuralSRNF (Figure 10-(c)). In both cases, we show results for $\tau \in [0, 1.5]$, at intervals of 0.25. $\tau = 1$ corresponds to $h_2 = \alpha(1)$ should have the same pose as h_1. The cases of $\tau > 1$ correspond to extrapolation beyond the range of the source deformation. As one can see in Figure 10-(b), deformation transfer using numerical SRNF [21] results in unnatural poses where the limbs unnaturally shrink. This shrinkage becomes significant as we extrapolate further away from the target pose. On the other hand, our method (Figure 10-(c)) obtains natural deformation transfer. As one can see, $h_2 = \alpha(1)$ corresponds exactly to the same pose as h_1. We can also observe that extrapolating beyond $\tau = 1$, using our NeuralSRNF framework, still results in natural poses.

D. Summary statistics and 3D shape generation

We evaluate the performance of the proposed framework in computing summary statistics and generating novel 3D shapes. Given a set of 3D shapes, we first normalize them for translation and scale, and convert each to its SRNF representation. We then compute the statistical summaries in the SRNF space

[1]We contacted the authors to request access, but did not receive a response. We tried independent reimplementation, but due to a lack of training and inference protocols, we were unable to reproduce visual qualitative comparisons.



(a) Source.



(b) Target before temporal registration.



(c) Target after temporal registration.



(d) Mean (with numerical SRNF [21], after temporal registration, time taken 45 min).



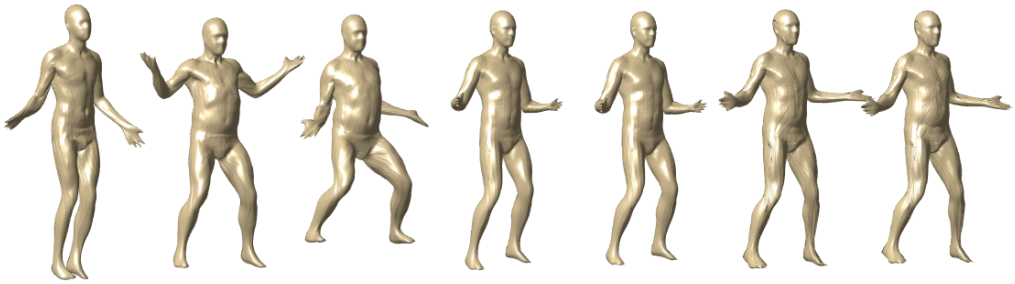
(e) Mean (with numerical SRNF [21], before temporal registration, time taken 45 min).



(f) Mean (with NeuralSRNF, after temporal registration, time taken 15 s).



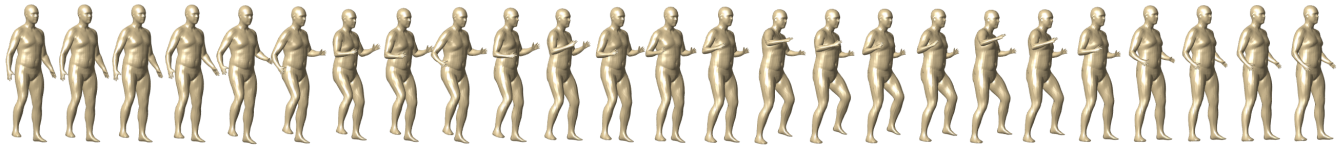
(g) Mean (with NeuralSRNF, before temporal registration, time taken 15 s).



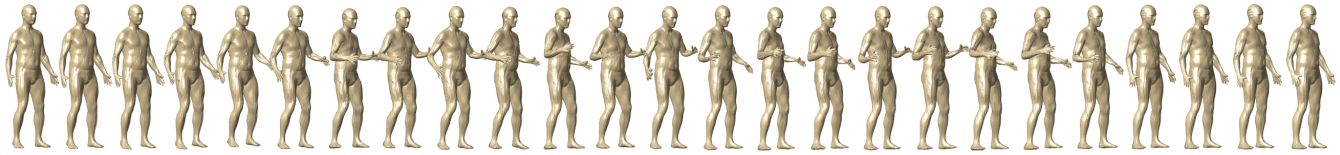
(a) Source | (b) Target unregistered | (c) Target registered | (d) Numerical [21] registered | (e) Numerical [21] unregistered | (f) Ours registered | (g) Ours unregistered

Fig. 15. Comparison of computational efficiency and inversion quality of mean 4D surface obtained using the numerical SRNF [21], [29] and the proposed NeuralSRNF. The bottom row, which highlights a 3D instance from each 4D sequence, shows that the proposed NeuralSRNF produces more natural mean shapes while achieving a $180\times$ speedup; see the Supplementary Material for additional results.





(a) Mean shape computed in the SRNF space and inverted using numerical SRNF [21] (45 min).



(b) Mean shape computed in the SRNF space and inverted using the proposed NeuralSRNF (15 sec).

Fig. 16. Example of the mean 4D surface computed on six 4D surfaces from the DFAUST dataset after temporal registration. We show the mean computed using **(a)** numerical SRNF [21] and **(b)** the proposed NeuralSRNF method. Here, the first four rows perform the same punching action and the following two perform a one-leg jump action. Note that the proposed NeuralSRNF is 180 times faster than the numerical method (15 s against 45 min).

following the algorithm presented in Section III-D, and map them back to the original space $\mathcal{F}$ for visualization, using our proposed method. We also use the computed mean shape and modes of variation to generate random surfaces in the SRNF space, using Eqn. (9), and invert them to the shape space $\mathcal{F}$ using our SRNF inversion approach. Figure 11 shows the statistical mean and the first three modes of variation computed on the animal and human dataset. Figure 12 shows some 3D animals generated using the proposed NeuralSRNF. Figure 13 compares the quality of 3D human models generated using numerical SRNF [21] (Figure 13-**(a)**) with those generated using our method (Figure 13-**(b)**). We can clearly see in the close-up views that the random shapes computed with our method capture all the characteristic features of the dataset. On the other hand, those computed using the numerical SRNF [21] have unnatural artifacts such as missing and shrunk limbs.

E. Application to 4D surfaces

We now demonstrate the applicability of the proposed NeuralSRNF to 4D surfaces. A 4D (or 3D+time) surface is



Fig. 17. Example of the two principal modes of variations computed on 4D faces in the SRNF space and inverted using the proposed NeuralSRNF.

a 3D surface that evolves over time, *e.g.,* a moving 3D body shape, a facial surface that performs some expressions, or a surface of an anatomical organ that grows over time (due to the aging process). Following [29], we represent each 3D surface along a 4D sequence using its SRNF. Thus, a 4D surface becomes curves in the SRNF space. The problem of temporal alignment and geodesics and summary statistics computation between 4D surfaces becomes that of temporal alignment and geodesics and summary statistics computation between curves in the SRNF space, which has an $\mathbb{L}^2$ structure, and then mapping the results back to the space of surfaces using SRNF inversion.

Figure 14 shows an example of joint temporal alignment (using DSNS [29]) and geodesics computation between two 4D surfaces performing a punching action. We compute the geodesic between the top row (source 4D surface) and the bottom row (target 4D surface) in the SRNF space and invert the geodesic path to the original surface space using the proposed NeuralSRNF. Compared to the numerical SRNF [21] used in DSNS [29], which takes around 45 min to invert one 4D surface, the proposed NeuralSRNF is 180 times faster (around 15 s per 4D surface) and more accurate.

Figure 15 compares the computation time and accuracy of the mean 4D surfaces obtained using the numerical SRNF [21], [29] and the proposed NeuralSRNF method. Figures 15-**(a)** and **(b)** show two 4D surfaces that perform a jumping action before temporal alignment. We align the source 4D surface (Figure 15-**(a)**) to the target surface in Figure 15-**(c)**. We then compute the mean 4D surfaces in the SRNF space and invert them back to the surface space using numerical SRNF [21] and NeuralSRNF. Figures 15-**(d)** and **(e)** show the mean 4D surfaces obtained using numerical SRNF after and before temporal registration, respectively. Note that it takes 45 minutes to compute the inversion using numerical SRNF [21], [29]. On the other hand, Figures 15-**(f)** and **(g)** show the mean 4D surface obtained using the proposed NeuralSRNF after and before temporal registration, respectively. Notice that NeuralSRNF only takes 15 s to generate the entire 4D sequence while being more accurate than the numerical method. Also, we show a close-up view of a selected 3D instance from the 4D sequences at the bottom. As one can see, the proposed NeuralSRNF is able to generate a plausible mean shape compared to the numerical SRNF [21], [29].

Figure 16 shows an example of temporal co-registration of multiple 4D surfaces, from the DFAUST dataset after temporal alignment, and their corresponding mean 4D surface. Here, the first four rows are performing a punching action and the last two rows are performing a one-leg jump action. We temporally co-aligned all these 4D surfaces using DSNS [29] approach and show in Figure 16-**(a)** the 4D mean surface obtained using the numerical SRNF [21], [29] approach, which takes 45 minutes and results in arm shrinkage. In Figure 16-**(b)**, we show the 4D mean surface obtained using the proposed NeuralSRNF. We can see that the latter, which is generated in less than 15 s, looks more natural.

Figure 17 shows an example of the mean and the first two principal modes of variation of 4D faces from COMA dataset [36]. As one can see, the 4D mean is able to capture the main facial features.

V. Conclusion

We propose NeuralSRNF, a novel continuous surface-based neural representation that provides a practical solution to the ill-posed SRNF inversion problem. Our solution is computationally very efficient while achieving significantly better accuracy than traditional methods. We further demonstrate how core statistical analysis tasks can be performed in the SRNF space, where the complex elastic metric reduces to an $\mathbb{L}^2$ metric, facilitating downstream analysis tasks such as geodesic computation, deformation transfer, summary statistics, and shape generation. Our approach achieves state-of-the-art results for 3D shape analysis across human body, face, and animal datasets. Importantly, it is up to 100 times faster than existing methods while achieving improved accuracy. Although efficient, our approach requires registered 3D surfaces, needs to be trained separately for each shape class, and is limited to genus-0 surfaces.

Acknowledgment

This work is supported by the Australian Research Council (ARC) Discovery Projects no. DP210101682, DP220102197, and DP260101891, and ARC Future Fellowship FT250100448.

REFERENCES

[1] V. I. Arnold. *Arnold's Problems*. Springer-Verlag, 2004.

[2] S. Attaiki and M. Ovsjanikov. Understanding and improving features learned in deep functional maps. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023.

[3] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 187–194, 1999.

[4] F. Bogo, J. Romero, M. Loper, and M. J. Black. Faust: Dataset and evaluation for 3d mesh registration. In *IEEE CVPR*, pp. 3794–3801, 2014.

[5] F. Bogo, J. Romero, G. Pons-Moll, and M. J. Black. Dynamic FAUST: Registering human bodies in motion. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[6] D. Cao, M. Eisenberger, N. Amrani, D. Cremers, and F. Bernard. Spectral meets spatial: Harmonising 3d shape matching and interpolation. In *CVPR 2024*, pp. 3658–3668, 06 2024. doi: 10.1109/CVPR52733.2024.00351

[7] D. Cao, P. Roetzer, and F. Bernard. Unsupervised learning of robust spectral shape matching. *ACM Transactions on Graphics (TOG)*, 2023. doi: 10.1145/3592107

[8] E. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[9] T. Cootes, C. Taylor, D. Cooper, and J. Graham. Active shape models-their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995. doi: 10.1006/cviu.1995.1004

[10] L. Cosmo, A. Norelli, O. Halimi, R. Kimmel, and E. Rodolà. Limp: Learning latent shape representations with metric preservation priors. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III*, pp. 19–35. Springer-Verlag, Berlin, Heidelberg, 2020. doi: 10.1007/978-3-030-58580-8_2

[11] B. Egger, W. A. Smith, A. Tewari, S. Wuhrer, M. Zollhoefer, T. Beeler et al. 3D morphable face models—past, present, and future. *ACM TOG*, 39(5):1–38, 2020.

[12] M. Eisenberger, D. Novotny, G. Kerchenbaum, P. Labatut, N. Neverova, D. Cremers et al. Neuromorph: Unsupervised shape interpolation and correspondence in one go. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7473–7483, 2021.

[13] D. Ginzburg and D. Raviv. Cyclic functional mapping: Self-supervised correspondence between non-isometric deformable shapes. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V*, pp. 36–52. Springer-Verlag, Berlin, Heidelberg, 2020. doi: 10.1007/978-3-030-58558-7_3

[14] F. Hartwig, J. Sassen, J. Braunsmann, M. Rumpf, and B. Wirth. Geodesic calculus on implicitly defined latent manifolds. In *International Conference on Machine Learning (ICML)*, 2026.

[15] B. Heeren, M. Rumpf, M. Wardetzky, and B. Wirth. Time-discrete geodesics in the space of shells. *Computer Graphics Forum*, 31:1755–1764, 08 2012. doi: 10.1111/j.1467-8659.2012.03180.x

[16] I. H. Jermyn, S. Kurtek, E. Klassen, and A. Srivastava. Elastic shape matching of parameterized surfaces using square root normal fields. In *European conference on computer vision*, pp. 804–817. Springer, 2012.

[17] I. H. Jermyn, S. Kurtek, H. Laga, and A. Srivastava. Elastic shape analysis of three-dimensional objects. *Synthesis Lectures on Computer Vision*, 12(1):1–185, 2017.

[18] M. Kilian, N. J. Mitra, and H. Pottmann. Geometric modeling in shape space. *ACM SIGGRAPH 2007 papers*, 26(3):64–es, July 2007. doi: 10.1145/1276377.1276457

[19] S. Kurtek, A. Srivastava, E. Klassen, and H. Laga. Landmark-guided elastic shape analysis of spherically-parameterized surfaces. In *Computer graphics forum*, vol. 32, pp. 429–438. Wiley Online Library, 2013.

[20] H. Laga. A survey on nonrigid 3D shape analysis. *Academic Press Library in Signal Processing, Volume 6*, pp. 261–304, 2018.

[21] H. Laga, Q. Xie, I. H. Jermyn, and A. Srivastava. Numerical inversion of srnf maps for elastic shape analysis of genus-zero surfaces. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2451–2464, 2017.

[22] C. Lemeunier, F. Denis, G. Lavoué, and F. Dupont. Representation learning of 3d meshes using an autoencoder in the spectral domain. *Computer Graphics*, 107(C):131–143, 2022. doi: 10.1016/j.cag.2022.07.011

[23] H. Liu, X. Zhan, Z. Yan, Z. Luo, Y. Wen, and X. Han. Stable-score: A stable registration-based framework for 3d shape correspondence. *Conference on Computer Vision and Pattern Recognition*, 2025.

[24] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

[25] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR)*, 2017.

[26] Q. Ma, J. Yang, A. Ranjan, S. Pujades, G. Pons-Moll, S. Tang et al. Learning to Dress 3D People in Generative Clothing. In *Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[27] R. Magnet and M. Ovsjanikov. Scalable and efficient functional map computations on dense meshes. In *Computer Graphics Forum*, vol. 42, pp. 89–101, 2023. doi: 10.1111/cgf.14746

[28] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*, pp. 405–421. Springer, 2020.

[29] A. Nizamani, H. Laga, G. Wang, F. Boussaid, M. Bennamoun, and A. Srivastava. Dynamic neural surfaces for elastic 4d shape representation and analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[30] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov et al. Dinov2: Learning robust visual features without supervision. In *Transactions on Machine Learning Research (TMLR)*, 2023.

[31] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Trans. Graph.*, 31(4), art. no. 30, 11 pp., July 2012. doi: 10.1145/2185520.2185526

[32] M. Ovsjanikov, E. Corman, M. Bronstein, E. Rodolà, M. Ben-Chen, L. Guibas et al. Computing and processing correspondences with functional maps. In *SIGGRAPH ASIA*, pp. 1–60, 11 2016. doi: 10.1145/2988458.2988494

[33] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *IEEE CVPR*, pp. 165–174, 2019.

[34] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. C. Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

[35] E. Praun and H. Hoppe. Spherical parametrization and remeshing. *ACM transactions on graphics (TOG)*, 22(3):340–349, 2003.

[36] A. Ranjan, T. Bolkart, S. Sanyal, and M. J. Black. Generating 3D faces using convolutional mesh autoencoders. In *European Conference on Computer Vision (ECCV)*, pp. 725–741, 2018.

[37] L. Sang, Z. Canfes, D. Cao, F. Bernard, and D. Cremers. Implicit Neural Surface Deformation with Explicit Velocity Fields. In *ICLR*, 2025.

[38] L. Sang, Z. Canfes, D. Cao, R. Marin, F. Bernard, and D. Cremers. 4Deform: Neural Surface Deformation for Robust Shape Interpolation. In *CVPR*, 2025.

[39] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *NeurIPS*, 2021.

[40] R. Williamson and N. J. Mitra. Neural Geometry Processing via Spherical Neural Surfaces. In *arXiv:2407.07755*, 2024.

[41] T. Windheuser, U. Schlickewei, F. R. Schmidt, and D. Cremers. Geometrically consistent elastic matching of 3d shapes: A linear programming solution. In *2011 International Conference on Computer Vision*, pp. 2134–2141. IEEE, 2011.

[42] B. Wirth, L. Bar, M. Rumpf, and G. Sapiro. A continuum mechanical approach to geodesics in shape space. *International Journal of Computer Vision*, 93(3):293–318, 2011.

[43] J. Xu, Y. Zhang, J. Peng, W. Ma, A. Jesslen, P. Ji et al. Animal3d: A comprehensive dataset of 3d animal pose and shape. *arXiv preprint arXiv:2308.11737*, 2023.

[44] H. Yang, B. Sun, L. Chen, A. Pavel, and Q. Huang. Geolatent: A geometric approach to latent space design for deformable shape generators. *ACM Trans. Graph.*, 42(6), art. no. 242, 20 pp., Dec. 2023. doi: 10.1145/3618371

[45] C. Zhang, B. Heeren, M. Rumpf, and W. A. Smith. Shell pca: statistical shape modelling in shell space. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1671–1679, 2015.

[46] H. Zheng, Z. Yang, W. Liu, J. Liang, and Y. Li. Improving deep neural networks using softplus units. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–4, 2015. doi: 10.1109/IJCNN.2015.7280459