CanFields: Consolidating Diffeomorphic Flows for Non-Rigid 4D Interpolation from Arbitrary-Length Sequences

Supplementary Material

A. Organization

This supplementary covers the diffeomorphic flow, velocity field analysis, articulated rigidity, dynamic consolidator architecture, multistage hyperparameters scheduling, complete implementation details, and additional experimental results. It also outlines additional robustness evaluations (missing regions, additional raw scans, sparse frames, sparse points, robustness to noise, topological artifacts, and normals), potential applications (arbitrary mesh discretization, consolidating textured scans, and dynamic texture generation), and limitations (topological changes, rapid motion transitions, and capturing finer geometric details).

Note: Figures, sections, and tables in the supplementary material are prefixed with a letter for distinction, while those without a prefix refer to content in the main paper.

B. Diffeomorphic Flow

We now present the core motivation of our methodology.

Temporal Coherence via Diffeomorphic Flows. 4D Implicit methods [2, 8, 22, 25] represent shapes independently per frame and lose explicit pointwise correspondences over time, resulting in temporally incoherent interpolation, difficulty in handling missing regions (Fig. 9), and noises (Fig. A9). To address this, we explicitly model deformation as a diffeomorphic flow via a continuous velocity field, and factor our representation into two parts:

1. **Canonical Shape** g_c: An implicit spatial function $g_c(x) : \mathbb{R}^3 \to \mathbb{R}$ defined at canonical time $c = 0.5$, implicitly representing the canonical shape $S_c = g_c^{-1}(0)$.
2. **Velocity Field** v: A spatially and temporally varying velocity field $v(x, t) : \mathbb{R}^3 \times \mathbb{R} \to \mathbb{R}^3$ defining a continuous diffeomorphic flow.

Diffeomorphic Flow via ODE Integration. Given the velocity field $v(x, t)$, we numerically integrate an ordinary differential equation (ODE) using standard solvers [5] to obtain flow maps. Specifically, given a point x at time 0, its forward flow map $\phi^{\rightarrow}$ to the position y at time t is defined as:

$$\begin{cases} \frac{\partial \phi^{\rightarrow}(x,\tau)}{\partial \tau} & = v(\phi^{\rightarrow}(x,\tau),\tau), \quad \tau \in [0,t) \\ \phi^{\rightarrow}(x,0) & = x, \\ \phi^{\rightarrow}(x,t) & = y. \end{cases}$$

Similarly, we define the backward flow map $\phi^{\leftarrow}$. Practically, we denote the flow map from arbitrary time t to canonical time $c = 0.5$ as $\phi(x, t \mapsto c)$, and its inverse for evaluation as $\phi(x, c \mapsto t)$. Thus, our deforming shape representation is $f(x, t) = g_c \circ \phi(x, t \mapsto c)$. This implicit-explicit decomposition offers several practical advantages:

- Explicit velocity modeling induces a smooth diffeomorphic flow, naturally ensuring consistent pointwise correspondences without requiring separate forward-backward networks or cycle-consistency constraints (Fig. A12).
- The canonical shape g_c aggregates geometric details from all frames, enabling robust reconstruction even with incomplete, sparse, or noisy data (Fig. 9, Fig. A9, Fig. A8).
- Using an implicit canonical shape flexibly allows the reconstruction of arbitrary topologies.
- Decoupling shape and motion representation, where the canonical shape captures fine geometric details and the velocity field encourages smooth deformation,

C. Velocity Field Analysis

We bias our velocity representation towards a mixture of low- and medium-frequency velocities. With this, we mitigate kinks and abrupt changes in the deformation. In addition to the Fourier encoding in space-time coordinates in Sec. 5.1, we use a mixture of two MLP layers to represent the velocity field itself (see Fig. 6). Consider two fully connected single-layer networks $\zeta_A : \mathbb{R}^d \to \mathbb{R}^h$ and $\zeta_B : \mathbb{R}^h \to \mathbb{R}^h$ and with $h = 512$. A SoftPlus [7] function activates these MLP layers. The purpose of this composition is to use the smooth attenuation bias of neural networks, where $\zeta_B \circ \zeta_A$ serves as a low-frequency component; using it alone (i.e., $w_M = 0$) would result in the loss of small details in the deformation. Using ζ_A alone (i.e., $w_L = 0$), as the medium-frequency component, results in failure to capture global motion. In Fig. A1, we ablate these weights and demonstrate that the average weighting (i.e., $w_L = w_M = 0.5$) is a successful choice, even when compared to the conventional skip connection $w_L = w_M = 1$.

D. Articulated Rigidity Details

Following [2, 26], physical objects, especially live creatures, tend to move in (softly) rigid semantic parts, such as the limbs between the joints. These kinematic units constitute a quasi-articulated system. For this, we adapt a motion-segmentation module from [26]. For a prescribed

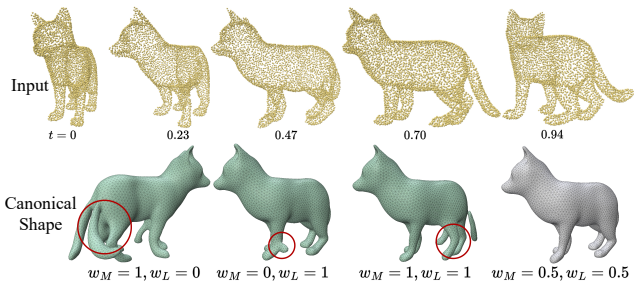

Figure A1. **Additional Velocity Ablation.** Given sequential point cloud inputs, different canonical shapes can be reconstructed when setting different w_M and w_L values. An average weighting scheme works best, and we adopt it throughout our experiments.

number of segments $H = 20$ (See the Supplementary in Zhang et al. [26] for the ablations), we learn a motion-segmentation network that is implemented as a neural field $\zeta_H : \mathbb{R}^3 \to [0,1]^H$, that for each location x in the space of the canonical time c outputs a probability $\zeta_{H,h}(x)$ of the point x belonging to segment h (as a partition of unity: $\sum_h \zeta_{H,h}(x) = 1$). We add a loss term that regularizes the per-part rigidity, by computing a pair of the rotation matrix and translation vector $\{R_{h,i} \in SO(3),\ \tau_{h,i} \in \mathbb{R}^3\}$ per each segment h and time frame i, where we seek that the flow from t_i to c is as-(part-wise)-rigid-as-possible [21], weighted by the probability of h:

$$E_{\text{rigid}}(i) = \mathbb{E}_{x_j \in \mathcal{P}_i} \sum_{h \in [1,H]} \zeta_{H,h}(\phi(x_j, t_i \mapsto c)) \cdot \|(R_{h,i} x_j + \tau_{h,i}) - \phi(x_j, t_i \mapsto c)\|^2, \quad (1)$$

In each iteration, we compute the rigid transformation $\{R_{h,i}, \tau_{h,i}\}$ in closed-form by SVD of the correlation matrix between the segment at time t and that of time c (the classical ARAP local step). As a by-product of this fitting, we get a segmentation of the moving object, by considering the highest probability of each point (Fig. A2).

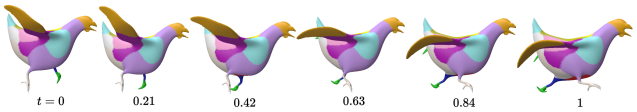

Figure A2. **Motion Segmentation.** The motion-segmentation network learns the articulation of the object and segments the (softly) piecewise-rigidly moving parts (different colors).

E. Experiments

E.1. Experimental Details

Dynamic Consolidator. Our dynamic consolidator (Sec. 4) takes as input a spacetime coordinate $\eta = (x, t)$ and an optimizable latent code θ, and outputs the perturbation δ and confidence score p. As illustrated in Fig. A3, it consists of MLP layers with 512 neurons per layer.

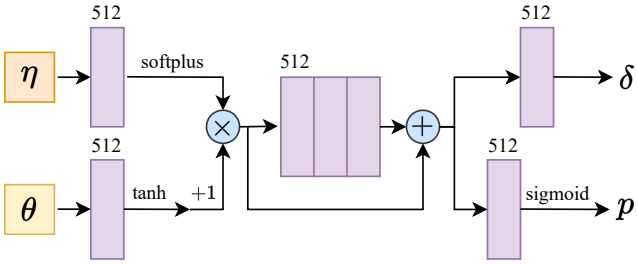

Figure A3. **Dynamic consolidator architecture.**

Multistage Hyperparameters Scheduling. Our experiments are run for 15000 full learning iterations. We implement a scheduling strategy for the velocity-network regularization weights λ_{kill}, λ_{diri}, and λ_{speed} as follows: they are initially set to 0 for the first 7000 iterations (during which we do not compute the unused associated derivatives); by doing so, the velocity field is only guided by $\widehat{E}_{\text{fit}}$, E_{eik} and E_{rigid}. We subsequently increase them as follows: from $\lambda_{\text{kill}} = 5 \times 10^{-5}$, $\lambda_{\text{dir}} = 8 \times 10^{-5}$, and $\lambda_{\text{speed}} = 1 \times 10^{-3}$ to 1.5×10^{-3}, 3.8×10^{-3}, and 2×10^{-2} respectively and uniformly over the next 5000 iterations. We then maintain them at 5×10^{-4}, 6.5×10^{-4}, and 5×10^{-3} respectively for the final 3000 iterations. We set the fixed $\lambda_{\text{rigid}} = 1 \times 10^3$, based on [26], and also set $\lambda_{\text{eik}} = 0.1$. Our loss coefficients λ_{mag}, λ_{var}, and λ_{log} for the dynamic consolidator, unless otherwise specified (e.g., Fig. A10), are initially set to $\lambda_{\text{mag}} = \lambda_{\text{var}} = 0.01$, and $\lambda_{\text{log}} = 0.10$ for the first 3000 iterations. We then increase them to 0.4, 30, and 2.0 respectively and uniformly over the next 9000 iterations, maintaining these final values for the last 3000 iterations.

Code and Hardware. We run all our experiments on a single NVIDIA A100 80GB GPU. Our code is based on PyTorch [16] and uses the torchdiff [5] package to implement the ODE solver. We use Polyscope [20] for visualization.

ODE Solver. To integrate the flow ϕ from the velocity v, we use the Dormand–Prince method 'dopri5' [6], setting relative and absolute error tolerances to 1×10^{-3} and 1×10^{-5}, respectively.

E.2. Additional Experiment Results

Additional Qualitative Comparisons. We provide additional qualitative results for 4D interpolation of synthetic animals from DeformingThings4D in Fig. A4.

E.3. Additional Robustness Evaluations

Missing Regions. In addition to qualitative evaluations in Fig. 9, we quantitatively compare our approach with DSR [23] on partially missing inputs from the animal dataset. Metrics in Tab. A1 confirm our method's superior performance.

Figure A4. **Additional Qualitative Comparisons.** Synthetic animal motions from DeformingThings4D [12]: (a) Point clouds, (b) OFlow [14] + NVFi velocity [11], (c) NDF [24], (d) OFlow [14], (e) DSR [23], and (f) Ours. Our method reconstructs natural temporal deformations, preserving geometric details without oversmoothing or introducing topological artifacts.

Addtional Raw Scans. Further interpolation results on raw DFAUST scans using our approach are shown in Fig. A5.

Sparse Frames. In Fig. A6, we show that our method can smoothly reconstruct motions even from very sparse input frames. As illustrated in Fig. A7, the competing method DSR [23] struggles to maintain near-isometric deformations, likely because it follows the network gradient rather than physically plausible motions. Furthermore, this com-

Table A1. **Quantitative Comparison: Inputs with Missing Regions.** Evaluation corresponds to Fig. 9 (animal dataset).

Sequence Name	Method	IoU(%)		CD($\times 10^{-5}$)	
		Mean↑	Min↑	Mean ↓	Max↓
deerFEL_WalkhuntedRM	DSR[23]	80.01	74.95	65.72	168.0
	Ours	**89.06**	**82.19**	**6.819**	**19.39**
bear3EP_WalkrightRM	DSR[23]	83.11	78.71	118.0	441.8
	Ours	**89.38**	**86.21**	**36.80**	**90.90**

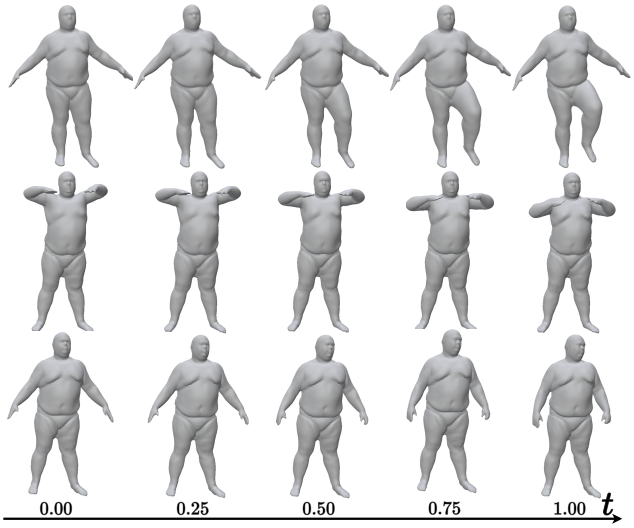

Figure A5. **Additional Raw Scans.** From top to bottom: one_leg_jump, chicken_wings, light_hopping_stiff from DFAUST (Subject ID = 50002).

parison highlights the importance of our proposed speed-consistency loss (E_{speed}), which helps preserve small-scale structures (e.g., rabbit eyes in Fig. A7) and reduces interpolation artifacts when the input frames are limited.

Sparse Points. Fig. A8 tests our robustness to input point sparsity with a fixed sequence of 14 frames. Remarkably, even with only 200 points per frame (a total of 14×200 points consolidated at canonical time), we accurately reconstruct overall geometry and deformation, though fine details are naturally reduced. Increasing the number of points per frame to 20K does not further improve reconstruction quality, indicating a possible saturation due to the underlying SIREN representation. Unlike implicit approaches that independently handle each frame without aggregation, our flow-based method consolidates points from all frames into a canonical shape, significantly boosting robustness.

Robustness to Noise. We evaluate our framework under noisy conditions in Fig. A9. Gaussian noise is added to 4 randomly selected frames within a 13-frame input sequence. Our method, especially the consolidator mod-

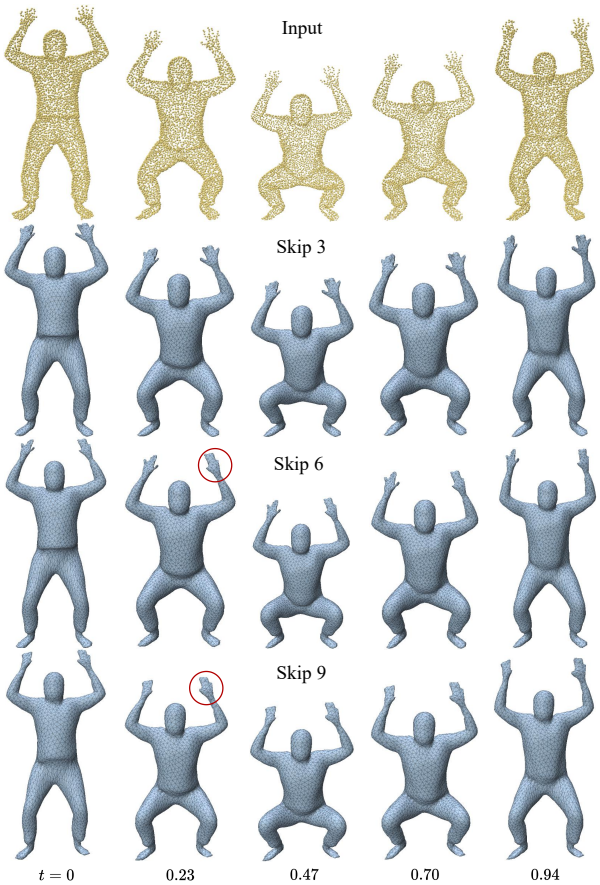

Figure A6. **Robustness to Frame Sparsity.** Interpolation results at identical time frames using varying input sparsity. "Skip" denotes temporal sampling intervals from the original sequence (e.g., a skip of 3 means using frames $1, 4, 7, \ldots$). Our method accurately reconstructs motions despite substantial frame skipping.

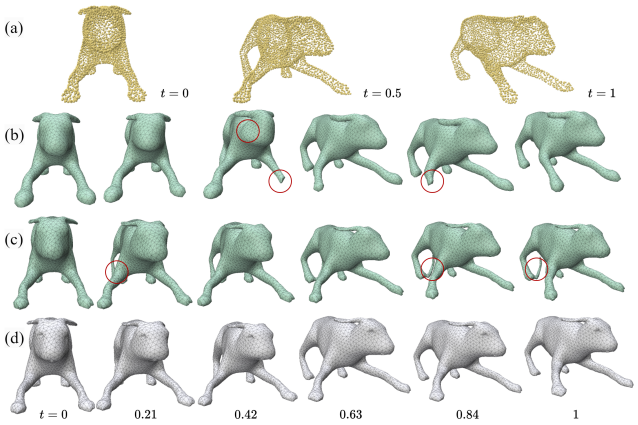

Figure A7. **Comparison under Extremely Sparse Input.** (a) Three-frame Input, (b) DSR, (c) Ours without speed-consistency loss (E_{speed}), (d) Ours (full). Our full approach achieves better interpolation quality and fewer visual artifacts (circled).

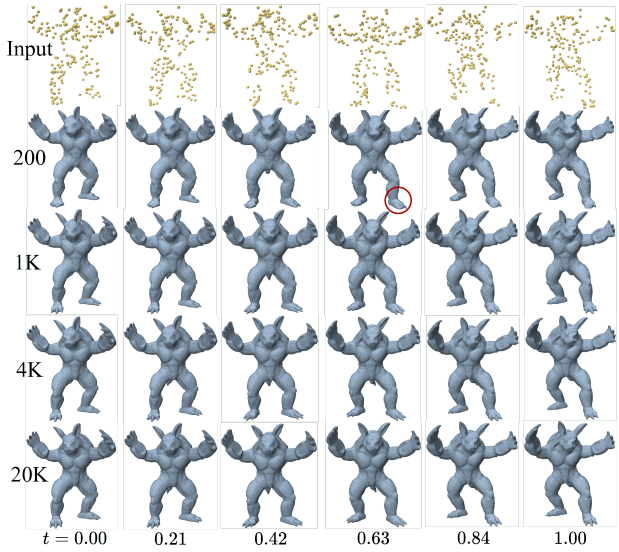

Figure A8. **Robustness to Sparse Point Input.** Using as few as 200 points per frame maintains shape reconstruction, with gradual detail loss. Results using 4K and 20K points per frame are nearly indistinguishable.

ule, effectively filters noise-induced artifacts compared to DSR [23], which produces overly smoothed geometry due to global sequence fitting.

Topological Artifacts. Flow-based methods are sensitive to incorrect canonical shape inference, motivating our consolidator module. In Fig. A10, we illustrate topology inference errors, such as mistakenly merging the hand into the torso. Compared to OFlow [14]+NVFi [11], which infers incorrect topology, and our framework without the consolidator (correct topology but geometric artifacts), our full approach accurately preserves topology and geometry throughout deformation. Here we adjust the default weight w_{log} to 0.005 to slightly reduce early reliance on raw points.

Normals. For input point clouds without provided normals, we substitute E_{fit} with the SALD loss term [1], which fits the unsigned distance function. Fig. A11 reveals that this change significantly compromises geometric reconstruction, demonstrating the importance of normals in our formulation. Future research will explore alternative loss functions to address the absence of normal data.

E.4. Applications

Arbitrary Mesh Discretization. Our method outputs an implicit canonical representation g_c, allowing flexibility in mesh discretization. Once a desired mesh (representing S_c) is extracted, vertex positions are directly updated via the learned velocity field to generate deformations. In

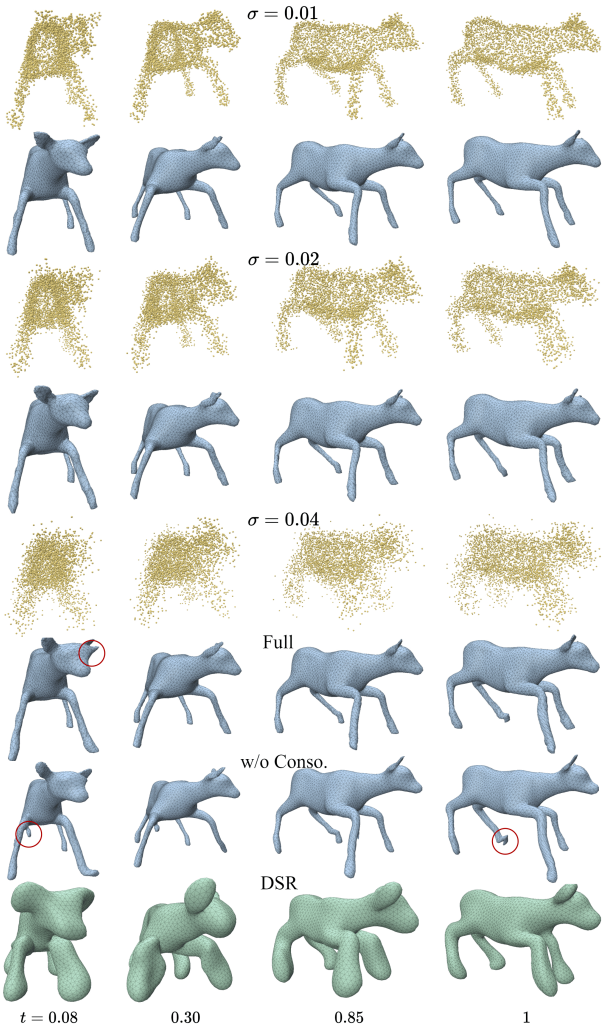

Figure A9. **Robustness to Gaussian Noise.** Our method (blue) maintains robustness across various noise levels. At high noise ($\sigma = 0.04$), removing the consolidator leads to visible artifacts. DSR (green) is particularly prone to noise, as highlighted by circled artifacts.

Fig. A12, we exemplify this versatility using both triangular and quadrilateral meshes, demonstrating intuitive deformations without significant distortion. The quadrilateral mesh is obtained using Instant Meshes [10].

Consolidating Textured Scans. Our explicit modeling of flow naturally supports the temporal propagation of vertex attributes such as textures. In Fig. A13, we demonstrate this capability using CAPE [17] raw scans. We sample 4K textured points from input scans, associate these textures directly with vertices on the canonical shape, and consistently advect textures through the velocity field to other time frames. Compared to NVFi, our approach yields more ac-

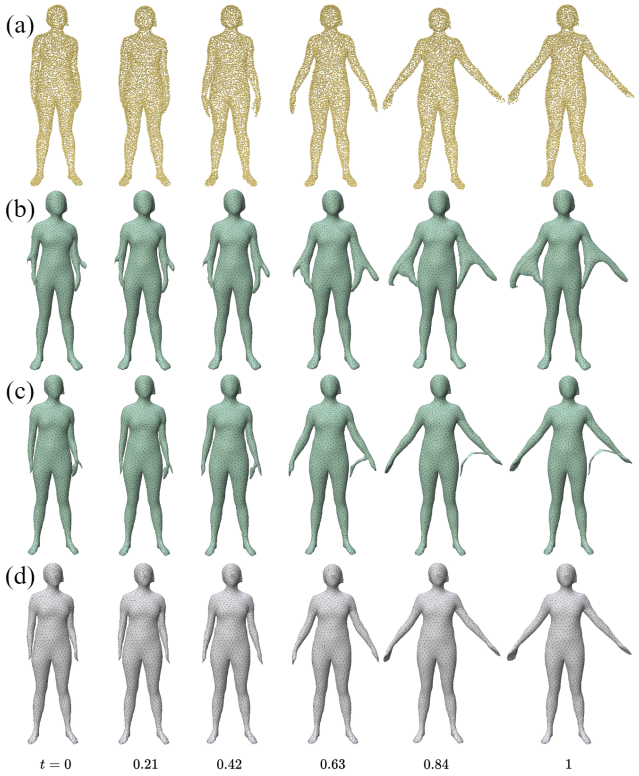

Figure A10. **Robustness to Topological Errors.** (a) Input sequence; (b) OFlow [14]+NVFi velocity [11]; (c) Ours w/o consolidator; (d) Ours (full), which accurately reconstructs motion with minimal artifacts.

curate and visually coherent textured mesh sequences.

Dynamic Texture Generation. Our approach readily integrates with mesh texturing workflows. We illustrate in Fig. A14 how we first apply Meshy [13] to generate textures on our canonical shape based on textual prompts, then propagate these textures dynamically using our learned velocity field.

E.5. Limitations

Despite the promising performance of our method, several challenges remain open for future exploration:

Topological Changes. Our approach assumes consistent topology under diffeomorphic deformation, making it well-suited to applications such as motion capture or studying individual object deformation. However, unlike purely implicit methods (e.g., DSR), which inherently allow topological changes such as splitting or merging, our method cannot model scenarios like protein recombination or molecular bond-breaking (see supplementary video).

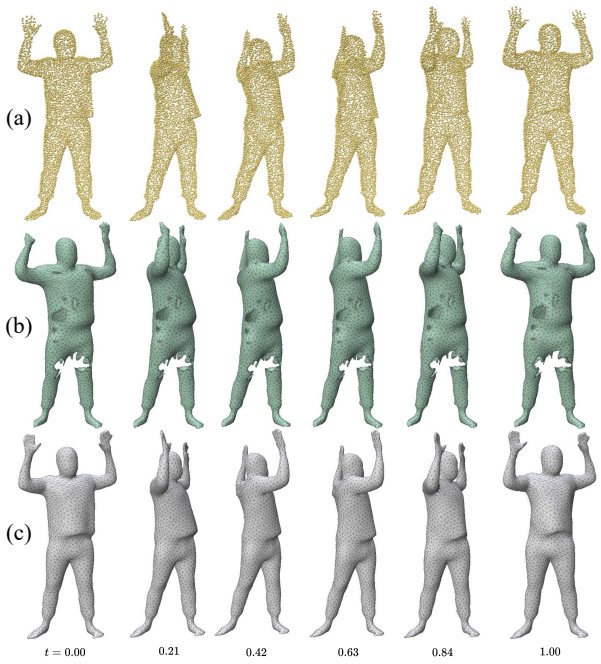

Figure A11. **Effect of Missing Normals.** (a) Input; (b) Ours without normals; (c) Ours (full). Normals are crucial for accurate geometry reconstruction.

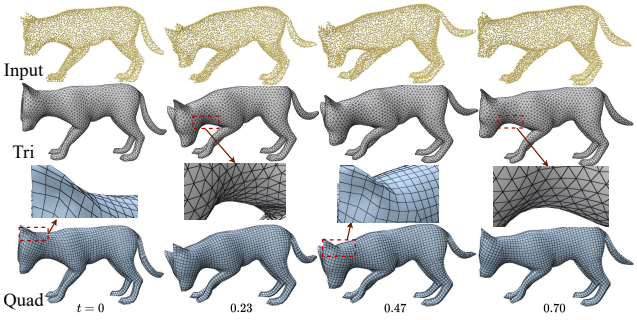

Figure A12. **Mesh Representation Flexibility.** Our implicit canonical representation allows natural deformation on both triangular and quadrilateral meshes, without introducing noticeable tangential distortions.

Rapid Motion Transitions. Our approach may introduce geometric artifacts when motions between consecutive frames exhibit large variations, even if the deformation remains nearly isometric and topology consistent. Fig. A15 illustrates such a scenario. Although fine-tuning regularization terms might reduce this issue, we leave this investigation as future work

Capturing Finer Geometric Details. While our method captures geometric details better than many existing approaches, reconstructing extremely fine-scale features re-

Figure A13. **Consolidating Textured Scans.** (a) Input point clouds (4K points per frame) sampled from raw textured scans. (b) Result from NVFi. (c) Our result, demonstrating accurate geometry and texture preservation.

mains challenging (see Fig. A8). Improving high-frequency detail reconstruction continues to be an important open problem in geometry processing.

E.6. Questions and Answers

Using fewer frames for large deformation. To demonstrate our robustness, we interpolate the result of **??** Bottom using just 3 frames (Fig. A16), maintaining robustness. To compare with source–target shape deformation, due to unavailable code for 4Deform [19] and issues with training scripts and pretrained models in Implicit-Surf-Deformation (ISD) [18], we adopt Neural Implicit Surface Evolution (NISE) [15], which is a common SOTA baseline under the same experimental settings (by implicitly fitting two shapes and building pairwise correspondences). We also compare to reconstructing the three frames by Poisson reconstruction which is sensitive to noise and outliers, and generating correspondence via Unsupervised Learning of Robust Spectral Shape Matching [3]. The result is visibly poor, not meeting the prerequisites for methods [4, 9].

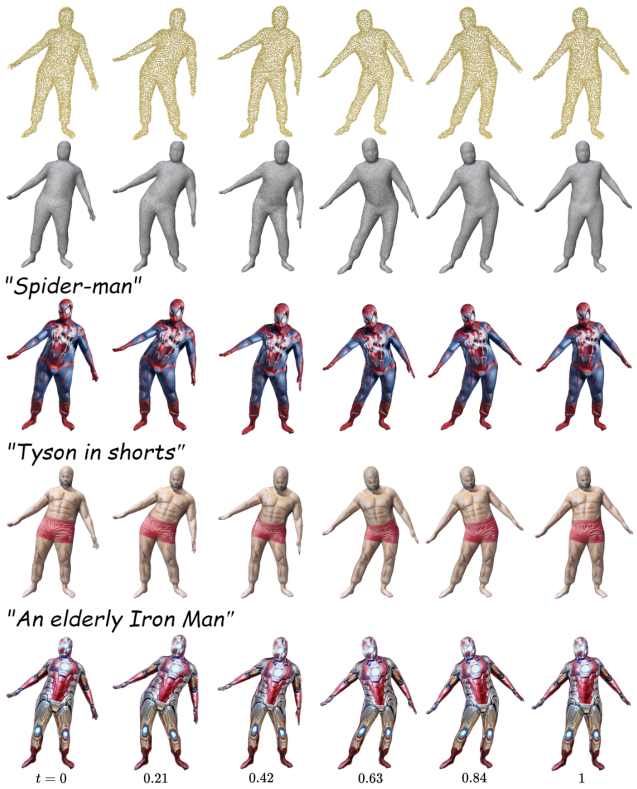

Figure A14. **Dynamic Texture Generation.** We generate textures on the canonical frame (left) and consistently advect them over time, achieving temporally coherent textured animations.

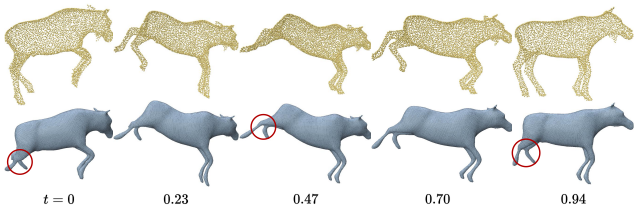

Figure A15. **Failure with Rapid Motions.** Fast movements between frames can cause geometric artifacts (circled) due to insufficient temporal resolution.

Missing Regions. We reran (Fig. A17) the experiment from **??**, using point clouds with significantly more missing regions ratios (ratios 0.35–0.63, mean 0.47). Even under these extreme conditions, our method completes well and remains robust.

Loss of Details. SIREN-based implicit functions tend to oversmooth fine structures like fingers. We show this in an

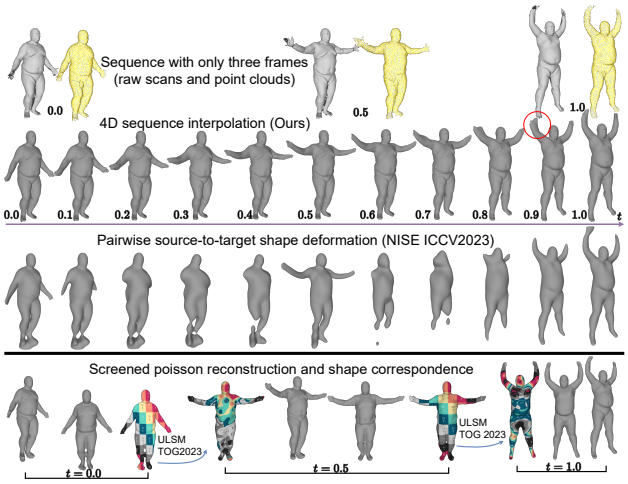

Figure A16. Interpolating only 3 frames with a large deformation.

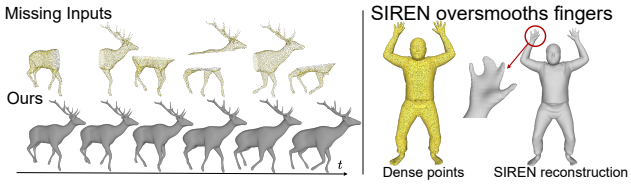

Figure A17. Left: Ours restores regions; Right: SIREN oversmooths.

"upper-bound" static reconstruction with the same number of samples as in *the entire sequence* (Fig. A6 and Fig. A17 Right). Mitigating this oversmoothing is beyond our scope as a valuable direction for future work.

References

[1] Matan Atzmon and Yaron Lipman. Sal: Sign agnostic learning of shapes from raw data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2565–2574, 2020. 5

[2] Matan Atzmon, David Novotny, Andrea Vedaldi, and Yaron Lipman. Augmenting implicit neural shape representations with explicit deformation fields. *arXiv preprint arXiv:2108.08931*, 2021. 1

[3] Dongliang Cao, Paul Roetzer, and Florian Bernard. Unsupervised learning of robust spectral shape matching. *ACM Transactions on Graphics (TOG)*, 42(4):1–15, 2023. 7

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

[5] Ricky T. Q. Chen. torchdiffeq, 2018. 1, 2

[6] John R Dormand and Peter J Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980. 2

[7] Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. *Advances in neural information processing systems*, 13, 2000. 1

[8] I. Eckstein, J.-P. Pons, Y. Tong, C.-C. J. Kuo, and M. Desbrun. Generalized surface flows for mesh processing. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, page 183–192, Goslar, DEU, 2007. Eurographics Association. 1

[9] Marvin Eisenberger, Zorah Lähner, and Daniel Cremers. Divergence-free shape correspondence by deformation. In *Computer Graphics Forum*, pages 1–12. Wiley Online Library, 2019. 7

[10] Wenzel Jakob, Marco Tarini, Daniele Panozzo, Olga Sorkine-Hornung, et al. Instant field-aligned meshes. *ACM Trans. Graph.*, 34(6):189–1, 2015. 5

[11] Jinxi Li, Ziyang Song, and Bo Yang. Nvfi: Neural velocity fields for 3d physics learning from dynamic videos. *Advances in Neural Information Processing Systems*, 36, 2023. 3, 5, 6

[12] Yang Li, Hikari Takehara, Takafumi Taketomi, Bo Zheng, and Matthias Nießner. 4dcomplete: Non-rigid motion estimation beyond the observable surface. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12706–12716, 2021. 3

[13] Meshy LLC. Create stunning 3d models with ai, 2024. 6

[14] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5379–5389, 2019. 3, 5, 6

[15] Tiago Novello, Vinicius Da Silva, Guilherme Schardong, Luiz Schirmer, Helio Lopes, and Luiz Velho. Neural implicit surface evolution. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14279–14289, 2023. 7

[16] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 2

[17] Gerard Pons-Moll, Sergi Pujades, Sonny Hu, and Michael J Black. Clothcap: Seamless 4d clothing capture and retargeting. *ACM Transactions on Graphics (ToG)*, 36(4):1–15, 2017. 5

[18] Lu Sang, Zehranaz Canfes, Dongliang Cao, Florian Bernard, and Daniel Cremers. Implicit neural surface deformation with explicit velocity fields. In *The Thirteenth International Conference on Learning Representations*, 2025. 7

[19] Lu Sang, Zehranaz Canfes, Dongliang Cao, Riccardo Marin, Florian Bernard, and Daniel Cremers. 4deform: Neural surface deformation for robust shape interpolation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2025. 7

[20] Nicholas Sharp et al. Polyscope, 2019. www.polyscope.run. 2

[21] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, pages 109–116. Citeseer, 2007. 2

[22] Jos Stam and Ryan Schmidt. On the velocity of an implicit surface. *ACM Transactions on Graphics (TOG)*, 30(3):1–7, 2011. 1

[23] Daiwen Sun, He Huang, Yao Li, Xinqi Gong, and Qiwei Ye. Dsr: Dynamical surface representation as implicit neural networks for protein. *Advances in Neural Information Processing Systems*, 36, 2023. 2, 3, 4, 5

[24] Shanlin Sun, Kun Han, Deying Kong, Hao Tang, Xiangyi Yan, and Xiaohui Xie. Topology-preserving shape reconstruction and registration via neural diffeomorphic flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20845–20855, 2022. 3

[25] Michael Tao, Justin Solomon, and Adrian Butscher. Near-isometric level set tracking. In *Computer Graphics Forum*, pages 65–77. Wiley Online Library, 2016. 1

[26] Baowen Zhang, Jiahe Li, Xiaoming Deng, Yinda Zhang, Cuixia Ma, and Hongan Wang. Self-supervised learning of implicit shape representation with dense correspondence for deformable objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14268–14278, 2023. 1, 2