

Template-free and Rig-free Animation Transfer using Kinetic Codes

SANJEEV MURALIKRISHNAN*, University College London, UK Dolby Laboratories, India

NILADRI SHEKHAR DUTT*, University College London, UK

NILOY J. MITRA, University College London, UK Adobe Research, UK

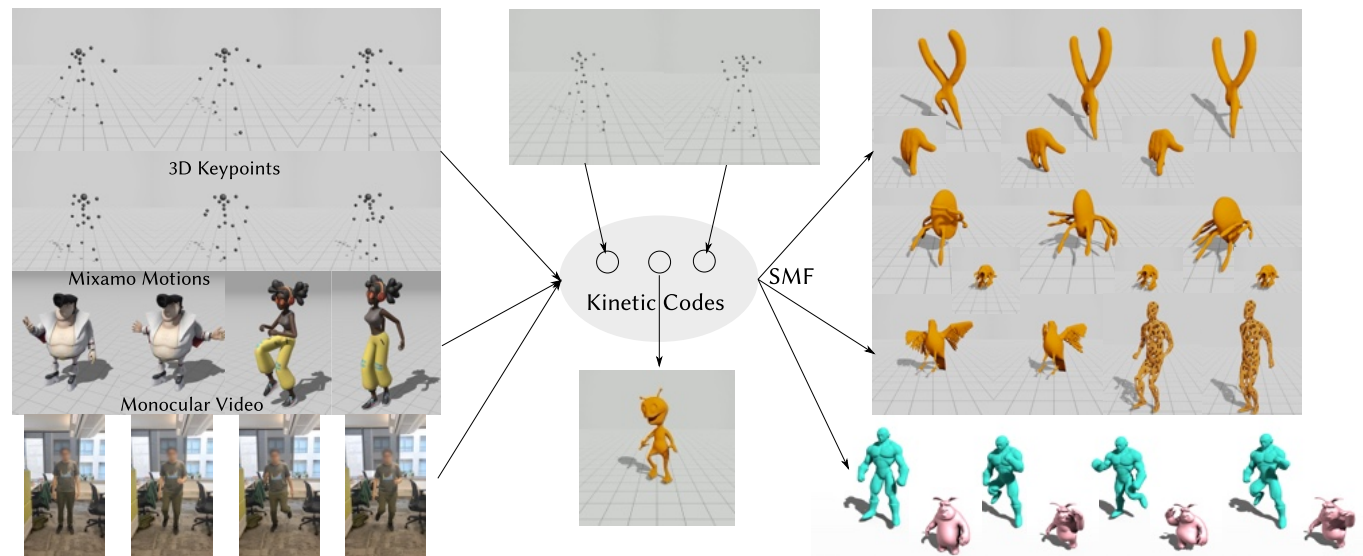


Fig. 1. **Animation Transfer from Unseen Motion to Different Characters.** We present a self-supervised method to transfer coarse motion sequences, embedded in a learned Kinetic Code (KC) space, to full body motion. Samples from the KC space can be consumed by our method, Self-supervised Motion Fields (SMF), to produce mesh animations. Our method is trained with sparse signals and can be used for motion interpolation. We do not assume access to any morphable model, canonical template mesh, or deformation rigs. Left shows various sparse motion inputs (3D keyframes, Mixamo sequences, or monocular video) that can be embedded into the learned KC space and decoded, and consumed by temporally coherent motion prediction via SMF to produce animations for different characters (right), with varying topology and shapes. SMF can faithfully transfer human motion to non-humanoid characters.

Animation retargetting applies sparse motion description (e.g., keypoint sequences) to a character mesh to produce a semantically plausible and temporally coherent full-body mesh sequence. Existing approaches come with restrictions – they require access to template-based shape priors or artist-designed deformation rigs, suffer from limited generalization to unseen motion and/or shapes, or exhibit motion jitter. We propose Self-supervised Motion Fields (SMF), a self-supervised framework that is trained with only sparse motion representations, without requiring dataset-specific annotations, templates, or rigs. At the heart of our method are Kinetic Codes, a novel autoencoder-based sparse motion encoding, that exposes a semantically rich latent space, simplifying large-scale training. Our architecture comprises

dedicated spatial and temporal gradient predictors, which are jointly trained in an end-to-end fashion. The combined network, regularized by the Kinetic Codes’ latent space, has good generalization across both unseen shapes and new motions. We evaluated our method on unseen motion sampled from AMASS, D4D, Mixamo, and raw monocular video for animation transfer on various characters with varying shapes and topology. We report a new SoTA on the AMASS dataset in the context of generalization to unseen motion.

CCS Concepts: • **Computing methodologies** → *Machine learning approaches*; **Shape analysis**; **Motion processing**: Motion capture; *Temporal reasoning*; **Animation**.

ACM Reference Format:

Sanjeev Muralikrishnan, Niladri Shekhar Dutt, and Niloy J. Mitra. 2025. Template-free and Rig-free Animation Transfer using Kinetic Codes. *ACM Trans. Graph.* 44, 6, Article 261 (December 2025), 11 pages. <https://doi.org/10.1145/3763309>

1 Introduction

Motion brings characters to life. To build any digital twin world, be it for scenario planning, games, or movies, adding motion to static characters is a fundamental requirement. Manually authoring full-body motion is tedious, error-prone, and requires significant effort from skilled artists. This quickly becomes expensive when

*Both authors contributed equally to this research.

Authors’ Contact Information: Sanjeev Muralikrishnan, sanjeev.muralikrishnan.20@ucl.ac.uk, University College London, UK and Dolby Laboratories, India; Niladri Shekhar Dutt, [Niladri.dutt.22@ucl.ac.uk](mailto:niladri.dutt.22@ucl.ac.uk), University College London, UK; Niloy J. Mitra, n.mitra@cs.ucl.ac.uk, University College London, UK and Adobe Research, UK.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM 1557-7368/2025/12-ART261

<https://doi.org/10.1145/3763309>

scaling to long sequences or animating many different characters. Hence, researchers seek data-driven solutions.

Traditional approaches investigate this problem with explicit priors — statistical body templates (e.g., SMPL [Loper et al. 2015] for humans, CAFM [Sun and Murata 2020] for animals) to model shape and pose variations or constructing specific character rigs to transfer joint motion to full body motion via skinning weights. These approaches are simple, popular, and robust, but come at the cost of having to first build an expressive shape space along with a corresponding pose space, and are restricted to specific templates.

Automating these workflows with learning-based solutions has gained popularity: learning a space of stick figure character motion (e.g., neural motion field [He et al. 2022]) or phase-based character control [Holden et al. 2017]); enabling deformation transfer from a source-target pair to a new shape via neural Jacobian fields [Aigerman et al. 2022; Muralikrishnan et al. 2024]; or learning skinning weights from a set of annotated rigged characters [Qin et al. 2023; Xu et al. 2020]. These approaches, however, require various levels of intermediate supervision, are limited in their handling of motion or shape variations, and suffer from limited generalization.

Given a coarse motion specification, we transfer animation to a full-character mesh, *without* access to any rig or morphable template, at train or test times. A natural approach is to treat this as a sequence prediction problem. However, this quickly leads to memory issues as the animations' length or the meshes' resolution increases. Increasing the network capacity adversely affects the situation, leading to overfitting as we often have sparse/limited training data. Also, treating the problem at the frame level is efficient, but leads to jittery motion without any temporal coupling.

Inspired by the recent success of latent space diffusion models [Rombach et al. 2021] over pixel space diffusion models, we ask if a similar latent space can be designed for (sparse) motion sequences. To this end, we propose *Kinetic Codes*, a temporally-informed lightweight motion autoencoder, that we train over a collection of sparse (humanoid) motion sequences across all types of motion. (Since we only rely on keypoints, instead of body meshes/template, we call the latent space kinetic instead of kinematic.) Regularized by this latent space, we train a spatial and a temporal gradient predictor network. We couple the networks through differentiable spacetime integration and supervise the framework, in an end-to-end fashion.

By representing source motion using only keypoints, we eliminate the need for geometric constraints such as 2-manifoldness, watertightness, or fixed triangulation in source meshes. Moreover, our motion representation allows for 2D source motion as input, which can be transferred to any stylized character (see Figure 1). This keypoint-based representation simplifies motion capture and facilitates sampling and interpolation of motions.

We evaluate SMF for generalization across diverse shapes and unseen motion, and compare against various alternatives. We evaluate our setup on a range of diverse shapes and motion datasets (e.g., AMASS, D4D, Mixamo, monocular video). In summary, we: (i) propose a self-supervised animation transfer framework regularized by kinetic codes, a learned temporally aware latent space; (ii) develop a rig- and template-free animation transfer framework based on simple keypoints as input that is easy to train with sparse supervision and generalizes robustly to new motions and stylized

characters; and (iii) report a new SoTA on the AMASS dataset and show realistic motion transfer to in-the-wild stylized characters using different 3D as well as 2D coarse space motion specifications.

Code, weights, and supplementary are available on the project webpage at <https://motionfields.github.io/>.

2 Related Works

Encoding shape deformation. Parameterized deformation approaches represent 2D or 3D shapes through a predetermined function of shared parameters and capture deformations as variations of these parameters. A popular example of such models is morphable models [Egger et al. 2019]. Such techniques encompass cages, explicit formulations [Ju et al. 2005] or neural approaches [Yifan et al. 2020], blendshapes [Lewis et al. 2014], skinned skeletons [Jacobson et al. 2014], Laplacian eigenfunctions [Rong et al. 2008], etc. Linking these parameters to the shapes' surface typically necessitates manual annotation of weights, commonly called weight painting, in 3D authoring tools. Alternatively, with access to sufficient supervision data, data-driven approaches can yield realistic neural rigs, as exemplified by Pinocchio [Baran and Popović 2007], RigNet [Xu et al. 2020], skinning-based motion retargeting [Aberman et al. 2020; Marsot et al. 2023; Zhang et al. 2023], and skeletal articulations with neural blend shapes [Li et al. 2021a]. Unsupervised shape and pose disentanglement [Zhou et al. 2020] proposes learning a disentangled latent representation of shape and pose, facilitating motion transfer using shape codes, dependent on registered meshes and maintaining identical connectivity. To plausibly animate these parameterized shapes over time, the parameters should evolve dynamically, weighing the mesh. Although these methodologies require access to body templates and/or rigs, they can still produce jittery results due to loose coupling of individual frame predictions.

Notably, Skeleton-free pose transfer [Liao et al. 2022] aims to alleviate the need for rigs by treating the character pose as a set of independent part deformations. By learning the skinning weights and deformations associated with each module, it can match the source pose using linear blending of skinning weights. While it achieves impressive results in pose transfer, being a per-frame method, it suffers from artifacts and lacks temporal coherence when applied to animation transfer, as observed in our results. Furthermore, its reliance on both source and target shapes being provided in a rest T-pose presents a practical limitation, as such canonical poses are often difficult to define for non-humanoids. This dependency challenges its classification as a truly rig-free method.

Modeling motion as sequence prediction. Deep recurrent neural networks are capable of modeling time and shape sequences using LSTMs to predict human joints [Fragkiadaki et al. 2015], generate motion in-betweening [Harvey et al. 2020], and to learn a motion field through time [He et al. 2022]. These approaches require access to templates/rigs, and large datasets of joint motion since they are discrete time representations. Qiao et al. [2018] utilize mesh convolutions with LSTMs to deform vertices through time; while, Motion Diffusion [Raab et al. 2023] uses local attention to capture motifs of a single motion and combines it with a diffusion UNet module to produce motion extrapolation and in-betweening. The main challenge is handling long (extrapolation) sequences while still being

able to generalize to unseen motion. In this work, we propose using Augmented Neural ODEs [Dupont et al. 2019], operating through temporally-aware kinetic codes, to model time continuously instead of using discretized sequential networks such as LSTMs.

Modeling motion using morphable templates. Temporal surface effects can be modeled, physically correctly, by simulating the underlying soft tissues with finite element methods [Chadwick et al. 1989; Fan et al. 2014]. However, this direct simulation is often slow and requires artists to design the underlying bone and muscle structure [Abdrashitov et al. 2021]. To overcome the stiffness problem and speed up the simulation, reduced-order models have been proposed [Modi et al. 2020; Park and Hodgins 2008]. When character rigs are available, approaches have been proposed to add soft tissue deformation as an additive per-vertex bump map on top of a primary motion model. Santesteban et al. [2020] use this approach, AMASS [Mahmood et al. 2019] imparts secondary motion using the blending coefficients of the SMPL shape space [Loper et al. 2015], while Dyna [Pons-Moll et al. 2015] learns a data-driven model of soft-tissue deformations using a linear PCA subspace. However, these methods require access to primary motion via a skeleton rig and are restricted to humans registered to a canonical template.

Temporal Residual Jacobian (TRJ) [Muralikrishnan et al. 2024] uses NJF and ODE for motion retargeting, and demonstrates good generalization across shapes. However, TRJ requires motion annotation and also access to SMPL template during training, has to be retrained for different motion classes, produces jitters due to per-frame prediction, and does *not* generalize to unseen motion. Unlike TRJ, which requires a template to learn motion transfer between different shapes, our self-supervised method reconstructs the original motion on the same shape, eliminating the need for annotated data.

3 Algorithm

Our goal is to animate unrigged triangulated meshes of 3D characters, conditioned on coarse motion signal. These motion control parameters are defined at a few keypoints of the body, can be varied in representation (e.g., 2D or 3D), and specify the target pose per frame. We learn to map these coarse keypoint parameters to dense 3D meshes and, as output, generate an animated 3D mesh at every frame described by the input motion. This nullifies the need for any consistent mesh template or fixed triangulation for both the source motion as well as the target character. Additionally, our method generalizes to unseen motion targets and unseen body shapes, learns from sparse datasets containing a mix of motion examples, and can be applied to long motion sequences.

3.1 Overview

Our method is a module named Self-Supervised Motion Fields (*SMF*) that maps inputs describing a target shape, X_0 , and motion, $\{\gamma_k\}$, to per-frame motion as 3D meshes,

$$X_k := \text{SMF}(\dots), \quad (1)$$

where X_k denotes the mesh vertices at frame k .

Input representation. Our inputs are coarse motion parameters, defined as γ_k , which characterize the pose required in each frame k and the target triangulated mesh defined as X_0 . In our experiments,

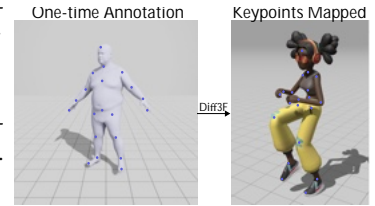
we have tested k ranging from 200 (short) to 4000 (long) sequences. We define γ_k at fixed keypoints extracted automatically from a given body. Specifically, γ_k is a $N_{\text{joints}} \times D$ vector at each time step, where N_{joints} is the number of keypoints/joints and D is the dimensionality of the chosen input representation. This representation can be 3D keypoint locations on a mesh, 2D keypoints defined on a stick figure frame, or 3D relative Euler angles computed according to the kinematic tree of the chosen pose space. In this work, we focus on results with 2D and 3D keypoints, as we observe their performance to be more robust than 3D relative Euler angles. Our method is self-supervised as we automatically extract the keypoints from mesh sequence.

We now define our module from Equation (1) as,

$$X_k := \text{SMF}(\gamma_k, X_0, C), \quad (2)$$

where C describes additional geometric features of the target character such as centroids, normals, and Wave Kernel Signatures [Aubry et al. 2011] computed on the faces of X_0 and encoded per-face using a shallow PointNet [Qi et al. 2016]. These additional features establish correspondence during inference on unseen in-the-wild shapes. We jointly train the PointNet network and other networks in our system. Figure 2 presents an overview of our method.

Keypoint extraction. We semi-automatically extract the keypoints γ_k . We start with a one-time manual annotation of **one** human and **one** animal shape, performed by selecting mesh faces at joint locations. These sparse annotations are then automatically propagated across all characters and motions using point correspondences computed with Diff3F [Dutt et al. 2024]. Our framework is robust to variations between character setups; for example, it bridges the gap even when target keypoint locations (e.g., from Mixamo) differ from the source. This flexibility is possible because our method only requires the number and order of keypoints to be consistent, not their exact spatial positions. This process, where we annotate just two shapes once, allows us to robustly handle diverse character and motion datasets. For extracting 2D keypoints from RGB videos, we use HRNet [Sun et al. 2019], a pretrained pose extractor.



3.2 Kinetic Codes: Temporally-Informed Motion Representation

At the core of our method is the shape deformation module f_D . We observed that naively passing the extracted coarse motion parameters γ_k to f_D impairs training, and leads to poor generalization and artifacts on unseen motions (see ablation in Section 4). This is unsurprising as an individual γ_k does *not* contain any temporal motion information or context. Therefore, we first embed γ_k in the latent space of an autoencoder. By coupling information across time, this representation leads to smoother interpolation, thereby enhancing generalization to unseen motions.

Further, given the sparsity of training data, we find that using the displacements of the motion parameters as inputs, instead of their

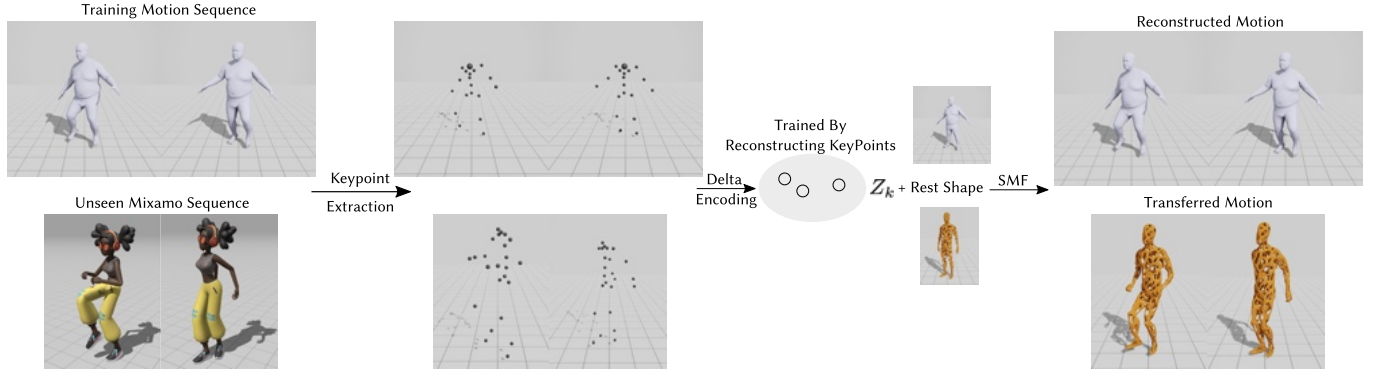


Fig. 2. **Method overview.** We present a self-supervised learning setup to transfer sparse motion information, specified in the form of keypoints over time, to target characters producing full-body motion. *Top:* During training, given a motion dataset we extract sparse keypoints from the meshes and encode them to a novel *Kinetic Code* representation. We then train two networks to map the rest shape and the Kinetic Code to the full body motion, with only mesh-level reconstruction loss. *Bottom:* At inference, we drop in stylized characters (Hole Man) and unseen motion inputs to obtain full-body character animation.

absolute values, significantly boosts generalization to unseen motion. Therefore, we express γ_k as displacement vectors with respect to the first frame motion parameter γ_0 as,

$$\gamma_k^\delta := \gamma_k - \gamma_0. \quad (3)$$

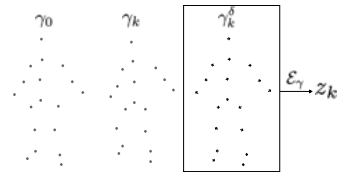
We then train a multi-headed attention auto-encoder with self-attention to reconstruct γ_k^δ as,

$$z_k = \mathcal{E}_Y(\gamma_k^\delta, \{\gamma_1^\delta, \dots, \gamma_k^\delta, \dots, \gamma_{N_f}^\delta\}) \quad (4)$$

$$\hat{\gamma}_k = \mathcal{D}_Y(z_k), \quad (5)$$

where $\mathcal{E}_Y, \mathcal{D}_Y$ are multi-headed attention encoder and decoder networks, respectively; z_k is the per-frame latent motion representation, referred to as *kinetic code*, of the same dimensionality as γ_k , and $\hat{\gamma}_k$ are the decoded motion displacements; N_f refers to number of frames.

The length of the sequence can vary across motion samples. Note that we use attention on the full sequence as context, and hence a single frame z_k has an understanding of the broader motion sequence.



We train the auto-encoder by minimizing the reconstruction loss,

$$L_Y := \sum_{k=1}^{N_f} \|\hat{\gamma}_k - \gamma_k^\delta\|^2. \quad (6)$$

Thus, we obtain a temporally-informed latent motion representation z_k , which is more conducive to interpolation, resulting in improved generalization to unseen motions and unseen shapes. For our deformation module (described next), we only use z_k as the per-frame motion representation, and freeze \mathcal{E}_Y and \mathcal{D}_Y .

3.3 3D Shape Posing via Deformation Module

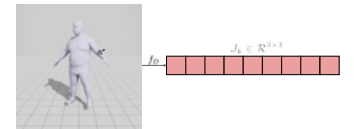
Inputs to our shape deformation module f_D comprise the initial configuration, specifically the shape X_0 in its canonical rest pose,

the learned motion latents z_k , and the geometric features C (derived from X_0). The module f_D is designed to forecast the pose for each time step $k \in [1, N_f]$. While f_D could be trained to directly forecast the target vertex positions, such an approach results in artifacts, including flipped and folded faces. Following neural gradient space processing [Aigerman et al. 2022; Muralikrishnan et al. 2024], we use a simple MLP to predict affine transformation matrices, labeled as Jacobians, at the centroids of the mesh faces, thus encoding relative transforms [Sumner and Popović 2004]. Specifically, we predict final vertex positions using predicted Jacobians, integrated through a differentiable Poisson solver to solve a system of linear equations. We supervise this using vertex-to-vertex and Jacobian losses to ensure accuracy in matching ground-truth data; this frees us from requiring additional annotation data. This improves shape consistency and can model various deformations by predicting affine transformations to mesh faces. Concretely, we define,

$$\begin{aligned} J_k^R &= f_D(J_0, z_k, C) \\ J_k &= J_0 + J_k^R = J_0 + f_D(J_0, z_k, C), \end{aligned} \quad (7)$$

where J_0 is the Jacobian of the initial frame shape X_0 and J_k^R are intermediate residuals predicted by f_D , which are added to J_0 to produce the Jacobian J_k for the k^{th} frame. We found that predicting the Jacobians using the Kinetic codes, coupled with residual connections leads to significantly faster and well-behaved convergence and better generalization. Note that J_0 is the identity transformation projected on the local basis of each face of X_0 by rotation. All Jacobians are predicted in the central coordinate frame defined by the local bases of the faces of X_0 .

We can now directly obtain final vertex positions using the Poisson Solver on the predicted Jacobians. However, as we drift away



from the pose of the initial shape X_0 or when inferring on unseen motion parameters, accumulated error leads to shape inconsistencies and artifacts. Additionally, f_D utilizes only coarse motion parameters and, by itself, is unable to produce temporally coherent and

smooth motions on dense meshes for long sequences. Hence, we use an Augmented ODE based formulation, described next, to enable temporally coherent predictions.

3.4 Temporally Coherent Motion Prediction

Motion prediction. We introduce a second stage that learns to predict a dynamic correction to the initial pose estimate to enforce temporal coherence. We predict the sequences in chunks of consecutive frames, i.e., the given sequence is split into fixed windows, each of size W ($W = 32$ in our tests). We set the initial state of the NODE as the Corrective Jacobians required for the first frame as,

$$J_0^C = \mathbf{0} \in \mathbb{R}^{3 \times 3}. \quad (8)$$

To make this operable with ANODE [Dupont et al. 2019], we augment the initial state with extra dimensions. We follow the recommendations of the original work and set the extra dimensions to $\mathbf{0} \in \mathbb{R}^A$ where A denotes the number of extra (augmented) dimensions to be added. We use $A = 256$ for all our experiments. In essence, these augmented dimensions act as a form of temporal memory for the ODE. By lifting the state space to a higher dimensional space where it can more easily predict the required trajectories, the system is able to maintain a richer state representation of the motion's history. This simplifies the task of learning the corrective dynamics needed to prevent drift. In other words, temporal message passing helps predict the corrective Jacobians. Equation (8) becomes,

$$\hat{J}_0^C = \mathbf{0} \in \mathbb{R}^{(3 \times 3) + A}. \quad (9)$$

We predict the correctives in the augmented space first, which is driven by an MLP f_C that predicts the rate of change of the correctives in time. The function f_C models the rate of change of the drift based on conditioning factors,

$$\frac{\partial \hat{J}_k^C}{\partial t} = f_C(\hat{J}_0, E_{W_n}^P, E_{W_{n-1}}^C, t) \quad (10)$$

where J_0 , as defined previously, are the Jacobians of the first frame; $E_{W_n}^P$ is the attention encoding of the current window of Jacobian predictions J_k from Equation (7) and $E_{W_{n-1}}^C$ is the attention encoding of the previous window of corrective predictions. Intuitively, $E_{W_n}^P$ provides the context of the current motion's structure, while $E_{W_{n-1}}^C$ informs the model about the accumulated error from the previous window. We integrate the local changes over time using Euler's method to obtain J_k^C at each time as,

$$\hat{J}_k^C = \int_0^t \frac{\partial \hat{J}_k^C}{\partial t} dt + \hat{J}_0^C = \int_0^t f_C(\hat{J}_0, E_{W_n}^P, E_{W_{n-1}}^C, t) dt + \hat{J}_0^C. \quad (11)$$

Since \hat{J}_k^C is in the augmented space, we use a final linear projection with learnable weights (W_p) to map it back to the original unaugmented space. Specifically,

$$J_k^C = W_p \hat{J}_k^C \quad (12)$$

where W_p is simply the learnable weights of the last linear layer projecting from $(3 \times 3) + A$ to the (3×3) Jacobian.

Our jointly trained attention encoders are defined as,

$$\begin{aligned} E_{W_n}^P &= E_P(J_{k:k+W}, T_{k:k+W}) \\ E_{W_{n-1}}^C &= E_C(J_{k:k-W}^C, T_{k:k-W}), \end{aligned} \quad (13)$$

where E_P and E_C are multi-head attention networks, $J_{k:k+W}$ and $J_{k:k-W}^C$ are the block of sequential Jacobians in the current window W and the past window $W - 1$, respectively; $T_{k:k+W}$ and $T_{k:k-W}$ are the corresponding blocks of time instances in these windows, which are positionally encoded.

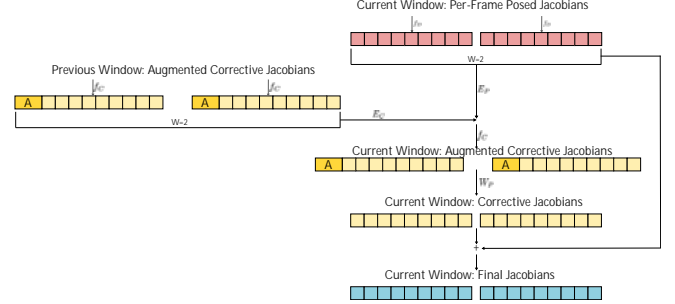


Fig. 3. **Windowed Jacobian Prediction.** We use attention encodings of the current window's posed Jacobians (Eq 7) and the previous window's augmented corrective Jacobians (Eq 10) to predict the current window's augmented corrective Jacobians. These are projected to predict the current window's corrective Jacobians (Eq 12). These correctives are then added to the posed Jacobians to obtain the current window's final Jacobians. We use window size $W = 32$.

These attention networks encode a window of Jacobians into a single encoding as shown in Figure 3. Fixing the encoding sizes to a constant size enables handling any arbitrary window/sequence length without overflowing memory. The encoders distill the current window of Jacobian predictions $J_{k:k+W}$ and previously predicted window of Corrective Jacobians $J_{k:k-W}^C$. We pass the output of the attention networks as conditioning to Equation (10) to integrate and obtain the correctives in Equation (11) in the augmented space, before projecting them in Equation (12) to obtain the final correctives. We add the predicted correctives to the posed Jacobians J_k from Equation (7). Finally, the predicted Jacobians J_k are spatially integrated using a differentiable Poisson solve [Nicolet et al. 2021], in the coordinate frame of the first frame, to obtain the predicted shape X_k at frame number k .

Loss terms. We train *end-to-end* using only a shape loss over vertices of X_k and a Jacobian loss. No extra annotation is required for supervision. Our final objective function is,

$$\mathcal{L}_{\text{vertex}} = \|X_k - X_k^{GT}\|^2 \quad \text{and} \quad \mathcal{L}_{\text{Jacobian}} = \|J_k - J_k^{GT}\|^2, \quad (14)$$

aggregated together into the total loss as $\mathcal{L} = \mathcal{L}_{\text{vertex}} + \alpha \mathcal{L}_{\text{Jacobian}}$ with $\alpha = 0.05$ in our experiments.

4 Evaluation

We test SMF along multiple axes: (i) generalization to unseen motion; (ii) animation transfer to different characters (shapes and topology; meshes and scans); (iii) motion specification using 3D keypoints, using monocular videos, or sampling/interpolating in the Kinetic Code (latent) space. No rigs or templates were used for any training or tests. We also test motion transfer from models trained on humanoid characters to non-humanoids.

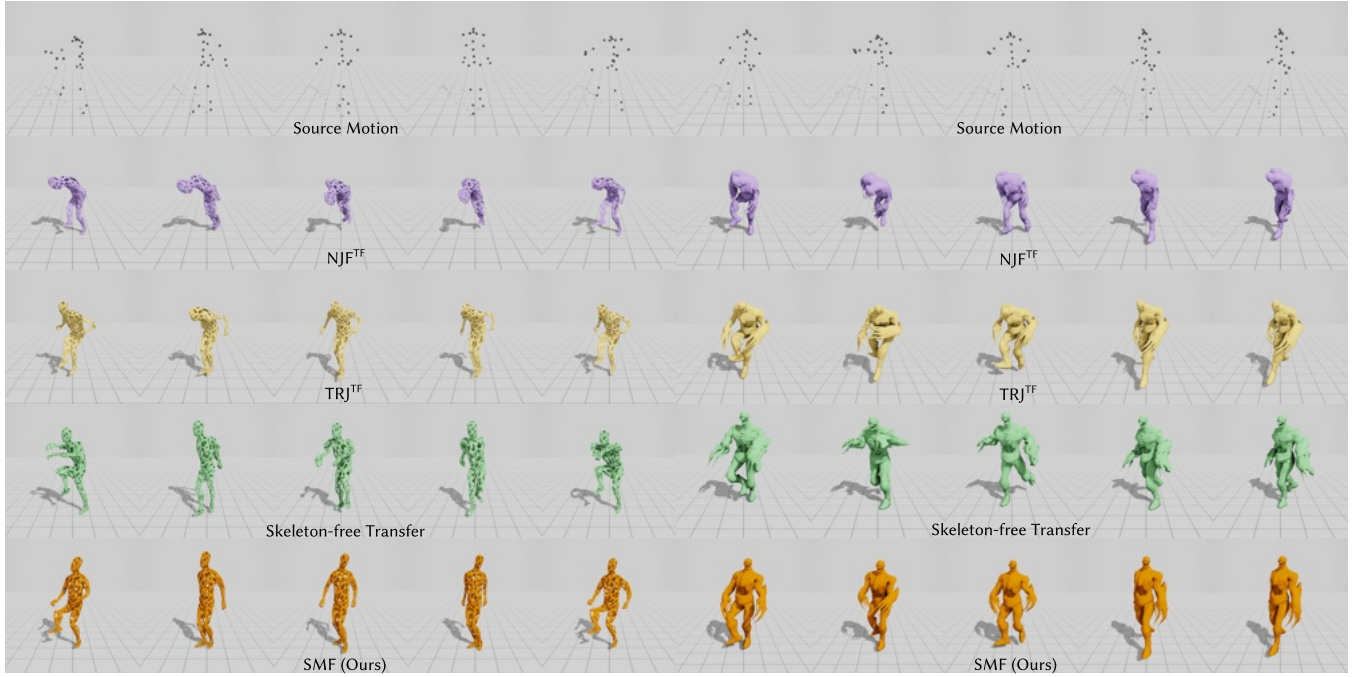


Fig. 4. **Unseen motion from Out-of-Distribution dataset (Mixamo) applied to in-the-wild shapes.** We compare *SMF* with *NJF*, *TRJ*, and *Skeleton-free transfer* on unseen dance motions (left: Hiphop; right: Shuffle) sampled from the out-of-distribution Mixamo dataset, applied to a 3D character found in-the-wild (hole man, left) and a Mixamo character (zombie, right). We modified *NJF*, *TRJ* to use keypoints instead, indicated by superscript *TF*. Competing methods exhibit distortion artifacts while attempting to follow the sampled source motion, while *SMF* (Ours) more accurately follows the sampled motion.

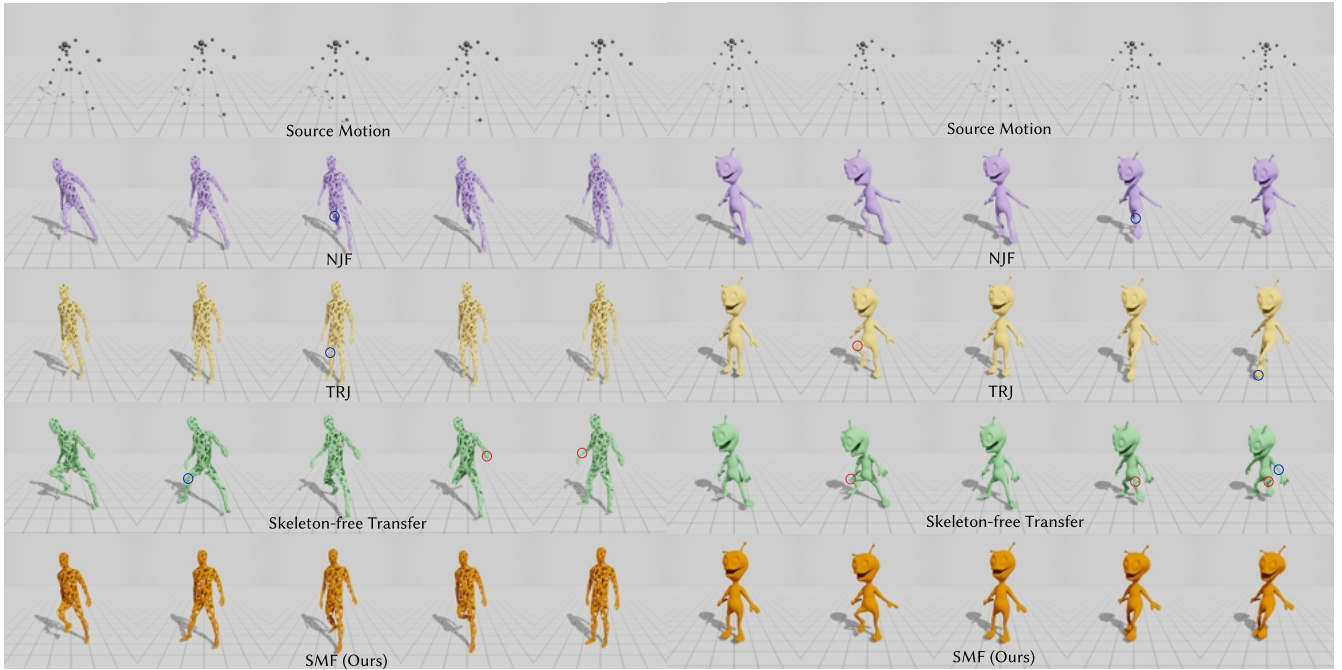


Fig. 5. **Unseen motion applied to in-the-wild shapes.** We compare *SMF* with *NJF*, *TRJ*, and *skeleton-free transfer* on unseen motion (left: Leg Backward Rotation; right: One Leg Jump), applied to in-the-wild 3D characters. Baselines often do not adhere to source motion (circled in blue) or exhibit distortion artifacts (circled in red). Our method transfers motion more accurately with far fewer shape distortion artifacts, while closely following the target motion.

Table 1. **Quantitative evaluation unseen motion category, unseen shape.** We compare SMF (ours) performance on unseen *motion* categories, against multiple competing methods as well as ablated versions of our method. We note that SMF consistently produces results with the lowest errors, when compared against ground truth full body target meshes. Please refer to the supplemental webpage for videos.

Method	One Leg Jump				Chicken Wings				Walk				Shake Shoulders				Knees				Shake Arms				Shake Hips			
	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N
Skeleton-free transfer [Liao et al. 2022]	4.60	0.67	0.44	13.67	3.27	0.65	0.50	13.78	5.27	0.78	0.65	19.27	5.27	0.72	0.46	13.97	4.71	0.62	0.40	12.82	3.74	0.70	0.52	15.39	3.83	0.75	0.56	16.19
SMF (3D) w/o γ_k encoding	4.56	0.70	0.32	12.56	4.22	0.68	0.35	12.86	5.84	0.71	0.34	12.92	4.19	0.68	0.30	11.85	5.20	0.77	0.36	13.79	5.21	0.80	0.44	19.79	4.48	0.71	0.35	13.26
SMF (2D keypoints)	5.74	0.72	0.33	12.42	6.50	0.80	0.44	19.43	6.87	0.79	0.36	13.49	6.77	0.87	0.38	15.63	8.33	0.88	0.39	14.81	8.81	0.96	0.50	22.39	7.74	0.89	0.43	17.06
SMF (3D keypoints)	2.79	0.51	0.25	9.82	3.74	0.61	0.34	12.72	3.73	0.54	0.27	10.24	3.09	0.58	0.28	11.44	3.78	0.63	0.30	11.27	4.41	0.69	0.41	18.45	3.40	0.58	0.31	12.28

Table 2. **Quantitative evaluation on Mixamo motion (unseen dataset), unseen stylized character.** We compare SMF (ours) against competing methods and report averaged errors (over 3 characters per motion) against rig-based Mixamo [Mixamo 2025] meshes as groundtruth. To work with Mixamo, we modified TRJ and NJF to use keypoints instead of SMPL parameters, indicated by the superscript TF.

Method	Hiphop				Shuffling				Surprised				Shaking Hands				Arguing			
	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N	L2-V	L2- δV	L2-J	L2-N
Skeleton-free transfer [Liao et al. 2022]	12.41	4.61	1.08	32.28	13.72	3.81	0.83	25.51	13.24	2.79	0.70	21.14	17.44	2.75	0.67	20.48	14.35	2.60	0.69	20.85
NJF ^{TF} [Aigerman et al. 2022]	25.57	9.03	1.13	45.94	25.38	9.02	1.11	45.67	26.27	8.96	1.10	45.24	31.80	10.68	1.19	45.80	29.22	9.71	1.14	45.26
TRJ ^{TF} [Muralikrishnan et al. 2024]	15.23	5.86	0.87	35.09	16.91	6.62	0.85	33.54	14.22	5.20	0.64	24.95	15.99	5.94	0.73	30.25	15.17	5.39	0.70	28.62
SMF (3D keypoints)	8.35	3.58	0.62	24.38	7.97	3.50	0.60	23.46	4.60	2.00	0.28	10.36	5.78	2.31	0.43	17.86	5.98	2.27	0.40	16.03

Table 3. **Generalization of Kinetic Codes.** We compare the reconstruction error of the kinetic codes of seen and unseen motion. The minimal variation in reconstruction errors indicates the generalizability of our codes to varied unseen motions.

	Seen Motion				Unseen Motion						
	Running	Jumping Jacks	Punching	Jiggle on Toes	Shake Hips	Hips	Shake Shoulders	One Leg Jump	Shake Arms	Walk	Chicken Wings
Error (in 10^{-3} cm)	7.73	11.88	12.03	11.13	6.88	8.36	8.34	8.51	7.95	8.96	7.89

Motion datasets. We train our method (and baselines) on a single dataset comprising of 5 human motion categories from the AMASS dataset [Mahmood et al. 2019] with each category containing approximately 6-7 motion sequences. We evaluate our method on unseen motion categories from AMASS, sampled motions from Mixamo, animal motion sequences from DeformingThings4D dataset [Li et al. 2021b], and in-the-wild monocular video recordings. The AMASS dataset utilizes the SMPL body model [Loper et al. 2015], which enables generation of motion sequences for diverse body-shapes by varying the shape parameter, β . Note that ours does *not* use this SMPL information during training or inference.

To show that our method works on animals, we train our method on motions from the DeformingThings4D dataset [Li et al. 2021b], which provides animal 4D meshes as deforming sequences and test it on unseen motion sequences.

We train SMF on 9 motion categories from AMASS, each consisting of sequences performed by 6 humans. The number of frames varies from 150-800. For 2D keypoints, we use 3 YouTube videos (2 characters in total), totaling 1hr. Although the videos are noisy as the camera angles change, SMF successfully learns due to the Kinetic codes setup.

4.1 Baselines

We compare our method against recent per frame (pose transfer) methods: NJF [Aigerman et al. 2022] and Skeleton-Free Pose Transfer [Liao et al. 2022]. Due to unavailability of pre-processing code for Skeleton-Free Pose Transfer, we use their pretrained models which were trained on substantially more data including AMASS and Mixamo motions as well as stylized characters. Additionally, it requires the full mesh sequence along with T pose meshes for source and target characters. In comparison, ours only takes in sparse keypoints as input. We also compare ours with animation

transfer methods: TRJ [Muralikrishnan et al. 2024] (which uses template) and an ablated version of our method without the Kinetic Code (γ_k) encoding. In the table below, we highlight the differences between SMF and baseline methods.

Methods	Rig-Free	Template-Free	Temporally-Coherent	Self-Supervised
NJF [Aigerman et al. 2022]	3	3	\times	\times
Skeleton Free [Liao et al. 2022]	\times	3	\times	\times
TRJ [Muralikrishnan et al. 2024]	3	\times	3	\times
SMF (ours)	3	3	3	3
Method	Input Requirements			
NJF [Aigerman et al. 2022]	SMPL Pose Parameters			
Skeleton Free [Liao et al. 2022]	T-Pose Source & Target characters, full body source motion			
TRJ [Muralikrishnan et al. 2024]	SMPL Pose and Bodyshape Parameters			
SMF (ours)	Sparse Keypoints			
NJF ^{TF}	Modified & trained with Keypoints			
TRJ ^{TF}	Modified & trained with Keypoints			

4.2 Metrics and Target Shapes

We evaluate our animation transfer with four main metrics:

- **Vertex-to-vertex error (L2-V):** Measures the Euclidean distance between ground-truth and predicted mesh vertices, indicating how well the global motion is captured.
- **Velocity error (L2- δV):** Quantifies differences in vertex velocities across frames, capturing the temporal smoothness and cohesion of the animation.
- **Jacobian error (L2-J):** Assesses deviations in local transformations, revealing unintended deformations.
- **Angular error of surface normals (L2-N):** Calculates the angle between predicted and ground-truth normals, indicating preservation of local surface orientations.

Target Shapes. We evaluate our method on diverse target shapes varied body shapes sampled from SMPL models, human scans from the FAUST dataset [Bogo et al. 2014], characters from the Mixamo library (e.g., skeleton zombie, triceratop, wolf), and in-the-wild

meshes from online 3D repositories (e.g., alien, hole-man). As pre-processing, when applicable, we fixed non-manifold meshes.

4.3 Qualitative Results

Generalization to unseen motion and shape. We present video results on various unseen motion and unseen shapes on our supplementary webpage. Our method produces consistently better generalization to unseen motion categories compared to NJF and TRJ, both of which result in shape distortion and erroneous displacements as they struggle to follow the input (unseen) motion as seen in Figure 4. This is particularly highlighted in scenarios with large displacements, e.g., feet and hands are widened or stretched thin. Note that unlike in the original TRJ [Muralikrishnan et al. 2024], where specialized models were separately trained for each motion type, we retrained a single TRJ, across all the motion types.

SMF also generalizes to new shapes of varied body types, including non-humans despite being trained only on humans. This correspondence from coarse keypoints to a dense mapping across diverse shapes is learned during the self-supervised motion transfer setup and proves to be even capable of generalizing to multi-legged creatures. It preserves the source motion and target shape and the resultant motion is realistic and free from jitters/artifacts. We show comparative results on unseen motion transfer to in-the-wild target meshes in Figure 4. For high-genus shapes, such as the mesh with holes (right half of Figure 4), NJF distorts the shape; TRJ fares comparatively better, it is still riddled with artifacts. Our Kinetic codes not only preserve the target shape but also more faithfully adhere to the motion. See supplemental webpage for video results.

Utilizing a latent representation for motion encoding with smooth interpolation properties leads to improved generalization. Hence, we do not see any significant artifacts even when operating on in-the-wild unseen target shapes for unseen motion. Moreover, we notice without our motion encoding, regions around joints may show melting (see around feet in Figure 6). The same holds true for more accurate bending of the joints (see Figure 4). We note that the AMASS motions contain foot-skating artifacts, and ours faithfully reproduces them. However, we also tested ours on high-quality Mixamo motions as shown in Figure 4 and supplemental webpage, where foot skating is less pronounced. Despite being trained only on AMASS, our model successfully transfers these clean Mixamo motions without introducing any additional skating artifacts. This demonstrates the model’s generalization not only to unseen motions but also to different levels of motion quality. This further suggests that our self-supervised setting is promising and training on higher-quality motion samples would likely yield better results.

Figure 1 and Figure 8 show more animation transfer examples to different humanoids and non-humanoids target meshes, sourced from online in-the-wild meshes and the SHREC’07 dataset [Giorgi et al. 2007]. Moreover, Figure 1 showcases support for different types of coarse motion specifications. In Figure 7, we show retargetting results on target animal shapes from the D4D dataset.

Generalization to Mixamo sampled motions on stylized characters. To transfer motion from Mixamo [Mixamo 2025], we map its joint locations (3D keypoints) to our system, (note the mapping is not perfect due to misalignment between joint locations, requiring

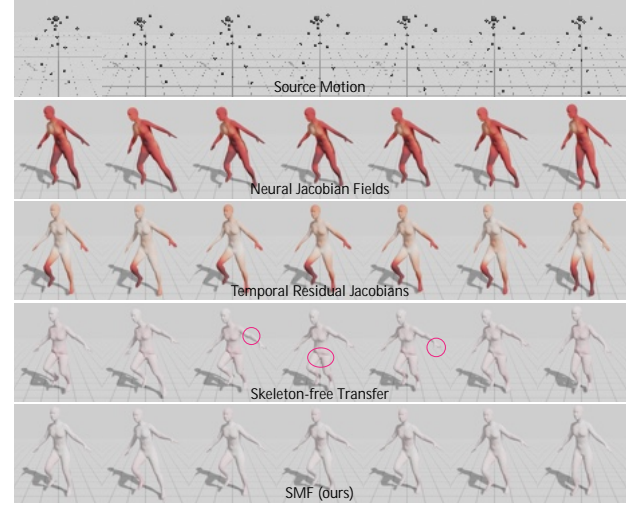


Fig. 6. **Comparison of SMF with baselines.** We compare *SMF* with Neural Jacobian Fields [Aigerman et al. 2022], Temporal Residual Jacobians [Muralikrishnan et al. 2024], and template-free skeleton-free transfer [Liao et al. 2022]. We measure the vertex-to-vertex error with ground truth and color-code the results according to the measured error. Darker red indicates higher error. *SMF* accurately transfers the motion to the target mesh, while baselines struggle to follow the input motion and exhibit distortion artifacts.

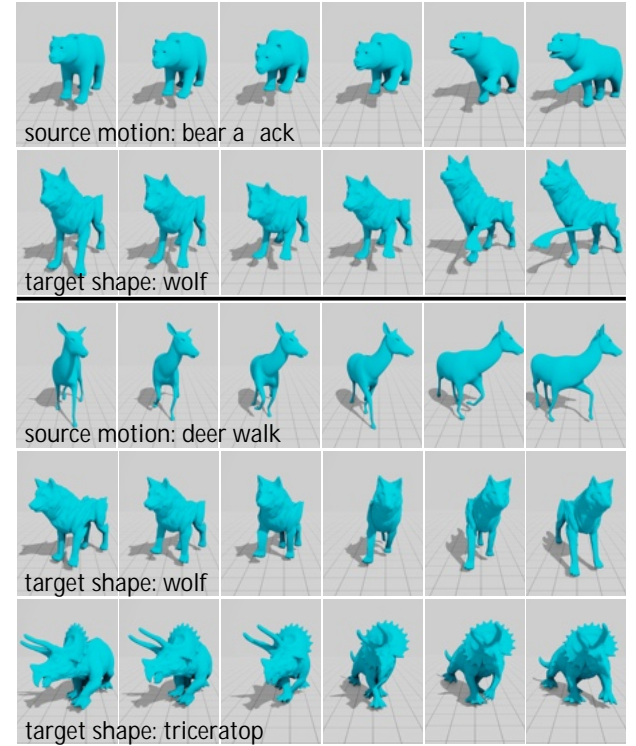


Fig. 7. **Unseen motion to animal shapes.** We use *SMF* for animation transfer to animal shapes (wolf and triceratop) for different source motions (top: bear attack; bottom: deer walk). Our method transfers motion faithfully while closely following the target motion (see supplemental videos).

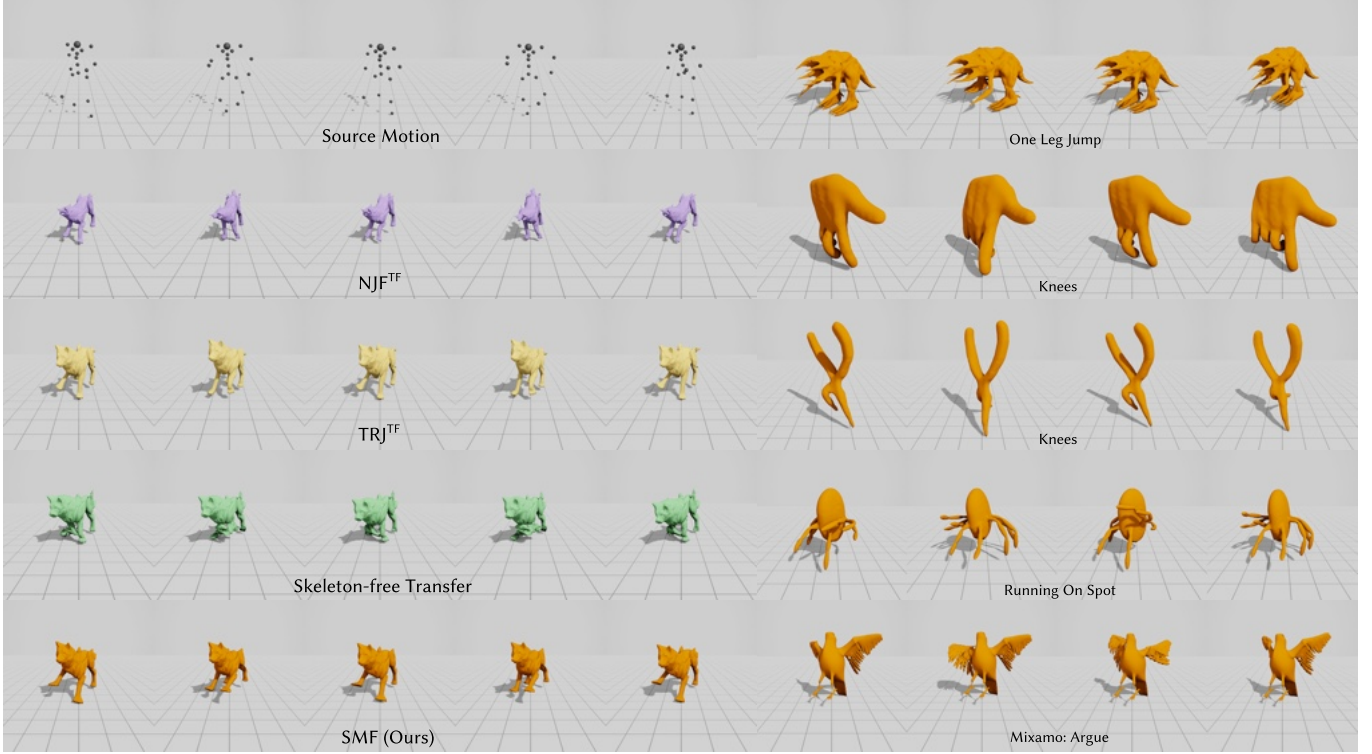


Fig. 8. **Human to non-human motion transfer** *Left*: We compare *SMF* with NJF, TRJ, and skeleton-free transfer on the task of transferring human motion to non-human characters. Since bodyshape parameters are unavailable for these animals, we utilize modified NJF and TRJ models trained on keypoints, as indicated by superscript TF. *SMF*, without using SMPL bodyshape parameters, generalizes to wider range of shapes without distortion artifacts. *Right*: *SMF* transfers varied unseen motions to several Out-of-Distribution characters. Unlike template-based methods, *SMF* does not use any body-shape descriptor.

some approximation). Our results demonstrate strong generalization, transferring these motions even to different characters. Notably, TRJ [Muralikrishnan et al. 2024] and NJF [Aigerman et al. 2022] are unable to operate on Mixamo motion, as they strictly depend on the SMPL template, which significantly limits their flexibility. Therefore, we modify their architecture to handle 3D coordinates and remove the shape parameter (β) module from TRJ. Please see supplemental videos on the webpage for comparison. We further evaluate it on 5 different motions on 3 stylized characters with ground truth generated from Mixamo. As seen in Table 2, we see large improvements over existing methods. Except Skeleton-free transfer (additionally trained on Mixamo data and stylized characters), all other baselines have been trained solely on the AMASS dataset with human characters.

Generalization to monocular capture. *SMF* can directly transfer raw motion sequences from 2D input videos captured on handheld cameras to a 3D target shape. We use our formulation based on 2D keypoints to directly operate on 2D input frames. This is a particularly challenging problem as we are lifting motion from 2D to a 3D shape *without* any explicit supervision or template shape. For our 2D training setup, we additionally train only the autoencoder on in-the-wild videos of exercise videos totaling 1 hour. We note this is noisy but this helps the γ_k generalize to RGB captures better

as all our 2D renderings of AMASS sequences have a similar camera setup. We extract 2D keypoints using HRNet [Sun et al. 2019], which computes 2D keypoints frame by frame. We use Savitzky-Golay filter to smoothen the 2D signal as there is inter-frame noise.

Self-supervised training. Our self-supervised setup enables us to learn motion transfer without requiring a template shape and using only coarse motion parameters extracted at joints from the motion dataset. This setup is the key to enabling learning from coarse input motion (2D/3D keypoints) and transferring motion from 2D monocular captures and Mixamo sampled motion to 3D target shapes.

4.4 Quantitative Comparison

We present quantitative results on unseen motion category and unseen motion sequence (within the same category) to unseen shape in Tables 1 and 2. Generalization to unseen motion is a particularly challenging problem as the Poisson solve in NJF and TRJ facilitates shape preservation for unseen shape by following the coordinates of the source shape. However, there is no such aid when generalizing to unseen motion, which is an extrapolation problem. Our method significantly outperforms NJF and TRJ for both motion accuracy as well as detail preservation. The benefits of our γ_k are highlighted here as we see significantly lower errors against a baseline of *SMF* without γ_k encoding. Compared to the per-frame pose transfer performed

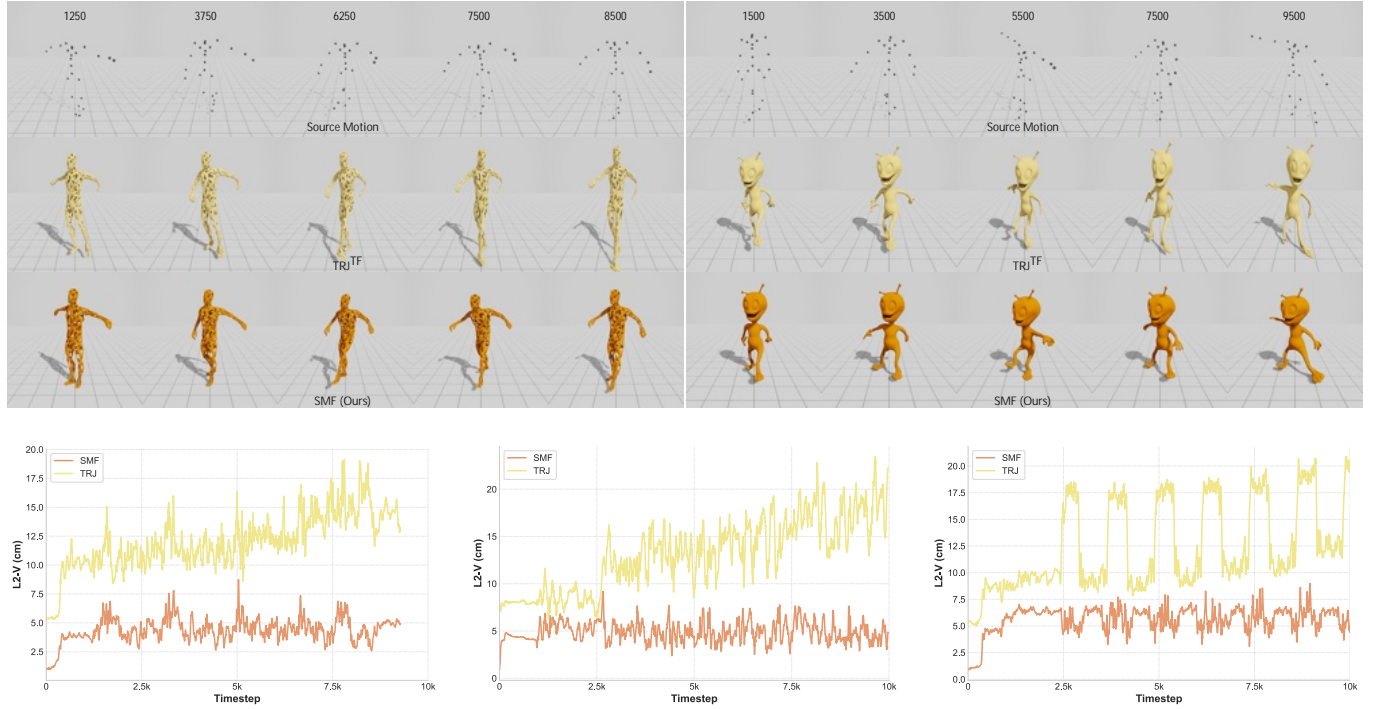


Fig. 9. **Temporal stability and comparison on very long motion sequences.** We compare *SMF* and *TRJ* on various *unseen* folk dance sequences (10000 frames) from the AMASS dataset. **Top.** Qualitative comparison on two unseen dance sequence. As the sequence progresses (frame numbers at top), *TRJ* accumulates significant error, exacerbating distortion artifacts. In contrast, *SMF* follows the source motion more closely. **Bottom.** Per-frame vertex-to-vertex (L2-V) error plots for three unseen dance sequences. *TRJ* displays a monotonically increasing error trend due to accumulation, while *SMF* maintains a consistently low error profile. This demonstrates *SMF*'s robustness to temporal drift and its suitability for generating long motion sequences.

by Skeleton-Free Pose Transfer which has numerous artifacts and unnatural deformations, *SMF* also ensures temporal consistency as seen in lower velocity error (L2- δV), measuring motion smoothness.

Error accumulation on long motion transfer. We choose 3 unseen, complex, and long dance sequences (10000 frames) to analyze error accumulation. We report per frame vertex-to-vertex (L2-V) error in Figure 9. Compared to the only other temporal method, *TRJ*, *SMF* does not exhibit a trend of monotonically increasing error demonstrating its robustness to temporal drift and its suitability for generating long motion sequences. The initial spike in error is due to the motion moving away from the easy-to-model rest pose.

4.5 Performance

SMF's lightweight architecture enables inference at >30 FPS on a single RTX 4090 for a mesh with 30k faces (for high number of faces this will be slower). This makes our system suitable for real-time applications, following a one-time preprocessing step to compute features on the target mesh. Furthermore, when using 2D keypoints, the system could potentially be extended for real-time 2D-to-3D animation transfer, as keypoint extraction with HRNet [Sun et al. 2019] is highly efficient.

5 Conclusion

We have presented *Self-supervised Motion Fields* that convert sparse motion into full-body character motion. We enable this by first creating a temporally-informed motion latent space – Kinetic codes – and then utilizing it to train spatial and temporal gradient predictor networks jointly. The gradient signals are coupled via spatial and temporal integration, and trained using full-body mesh sequences for supervision. *SMF* does *not* require additional motion annotation, is simple and robust to train, and generalizes across unseen motion and character shape variations. We also do not use any template model or rig, either during training or inference. We establish a new SOTA on the AMASS dataset.

Limitations. While our formulation captures some secondary motion, we do not model secondary dynamics in any physically correct way. Neither do we explicitly support collision handling to prevent self-intersection (see the triceratop leg crossing in Figure 7). An interesting future direction would be to implicitly handle collision detection and simultaneously model character-garment interactions, possibly using a transformer-based attention mechanism to capture non-local interactions between body parts. Similar to single human bodies [Frühstück et al. 2022], it will be interesting to combine different motion generators for different parts of a character, allowing mixing of motion modes. Finally, temporal sketching sequences [Fu et al. 2011] can also be modeled with such a generative framework.

References

- Rinat Abdrashitov, Seungbae Bang, David Levin, Karan Singh, and Alec Jacobson. 2021. Interactive Modelling of Volumetric Musculoskeletal Anatomy. *ACM Transactions on Graphics* 40, 4 (2021).
- Kfir Aberman, Peizhuo Li, Dani Lischinski, Olga Sorkine-Hornung, Daniel Cohen-Or, and Baoquan Chen. 2020. Skeleton-aware networks for deep motion retargeting. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 62–1.
- Noam Aigerman, Kunal Gupta, Vladimir G Kim, Siddhartha Chaudhuri, Jun Saito, and Thibault Groueix. 2022. Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes. *SIGGRAPH* (2022).
- Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. 2011. The wave kernel signature: A quantum mechanical approach to shape analysis. In *2011 IEEE international conference on computer vision workshops (ICCV workshops)*. IEEE, 1626–1633.
- Ilya Baran and Jovan Popović. 2007. Automatic rigging and animation of 3D characters. *ACM Trans. Graph.* 26, 3 (2007), 72–es.
- Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. 2014. FAUST: Dataset and evaluation for 3D mesh registration. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3794–3801.
- John E Chadwick, David R Haumann, and Richard E Parent. 1989. Layered construction for deformable animated characters. *ACM Siggraph Computer Graphics* 23, 3 (1989), 243–252.
- Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. 2019. Augmented neural odes. *Advances in neural information processing systems* 32 (2019).
- Niladri Shekhar Dutt, Sanjeev Muralikrishnan, and Niloy J Mitra. 2024. Diffusion 3d features (diff3f): Decorating untextured shapes with distilled semantic features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4494–4504.
- Bernhard Egger, William A. P. Smith, Ayush Tewari, Stefanie Wührer, Michael Zollhöfer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, Christian Theobalt, Volker Blanz, and Thomas Vetter. 2019. 3D Morphable Face Models - Past, Present and Future. *CoRR* abs/1909.01815 (2019). arXiv:1909.01815 <http://arxiv.org/abs/1909.01815>
- Ye Fan, Joshua Litven, and Dinesh K Pai. 2014. Active volumetric musculoskeletal systems. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–9.
- K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik. 2015. Recurrent Network Models for Human Dynamics. In *ICCV*.
- Anna Frühstück, Krishna Kumar Singh, Eli Shechtman, Niloy J. Mitra, Peter Wonka, and Jingwan Lu. 2022. InsetGAN for Full-Body Image Generation. In *CVPR*. 7723–7732.
- Hongbo Fu, Shizhe Zhjou, Ligang Liu, and Niloy Mitra. 2011. Animated Construction of Line Drawings. *ACM Transactions on Graphics* 30, 6 (2011).
- Daniela Giorgi, Silvia Biasotti, and Laura Paraboschi. 2007. Shape retrieval contest 2007: Watertight models track. *SHREC competition* 8, 7 (2007), 7.
- Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. 2020. Robust Motion In-Betweening. 39, 4 (2020).
- Chenghan He, Jun Saito, James Zachary, Holly Rushmeier, and Yi Zhou. 2022. NeMF: Neural Fields for Kinematic Animation. In *NeurIPS*.
- Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-functioned neural networks for character control. *ACM Trans. Graph.* 36, 4, Article 42 (jul 2017), 13 pages.
- Alec Jacobson, Zhigang Deng, Ladislav Kavan, and J. P. Lewis. 2014. Skinning: Real-time Shape Deformation. In *SIGGRAPH Courses*.
- Tao Ju, Scott Schaefer, and Joe Warren. 2005. Mean Value Coordinates for Closed Triangular Meshes. In *SIGGRAPH*.
- J. P. Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Fred Pighin, and Zhigang Deng. 2014. Practice and Theory of Blendshape Facial Models. *Eurographics State of the Art Reports* (2014).
- Peizhuo Li, Kfir Aberman, Rana Hanocka, Libin Liu, Olga Sorkine-Hornung, and Baoquan Chen. 2021a. Learning Skeletal Articulations with Neural Blend Shapes. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 130.
- Yang Li, Hikari Takehara, Takafumi Taketomi, Bo Zheng, and Matthias Nießner. 2021b. 4dcomplete: Non-rigid motion estimation beyond the observable surface. *IEEE International Conference on Computer Vision (ICCV)* (2021).
- Zhouyingcheng Liao, Jimei Yang, Jun Saito, Gerard Pons-Moll, and Yang Zhou. 2022. Skeleton-free Pose Transfer for Stylized 3D Characters. In *European Conference on Computer Vision (ECCV)*. Springer.
- 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.
- Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. 2019. AMASS: Archive of Motion Capture as Surface Shapes. In *International Conference on Computer Vision*. 5442–5451.
- Mathieu Marsot, Rim Rekik, Stefanie Wührer, Jean-Sébastien Franco, and Anne-Hélène Olivier. 2023. Correspondence-free online human motion retargeting. arXiv:2302.00556 [cs.CV]
- Mixamo. 2025. Mixamo Animation Service. <https://www.mixamo.com/> Accessed: 2025-03-08.
- V. Modi, L. Fulton, A. Jacobson, S. Sueda, and D.I.W. Levin. 2020. EMU: Efficient Muscle Simulation in Deformation Space. *Computer Graphics Forum* (Dec 2020). doi:10.1111/cgf.14185
- Sanjeev Muralikrishnan, Niladri Shekhar Dutt, Siddhartha Chaudhuri, Noam Aigerman, Vladimir Kim, Matthew Fisher, and Niloy J Mitra. 2024. Temporal Residual Jacobians For Rig-free Motion Transfer. *ECCV 2024 preprint arXiv:2407.14958* (2024).
- Baptiste Nicolet, Alec Jacobson, and Wenzel Jakob. 2021. Large Steps in Inverse Rendering of Geometry. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 40, 6 (Dec. 2021). doi:10.1145/3478513.3480501
- Sang Il Park and Jessica K Hodgins. 2008. Data-driven modeling of skin and muscle deformation. In *ACM SIGGRAPH 2008 papers*. 1–6.
- Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J Black. 2015. Dyna: A model of dynamic human shape in motion. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–14.
- Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2016. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *arXiv preprint arXiv:1612.00593* (2016).
- Yi-Ling Qiao, Lin Gao, Yu-Kun Lai, and Shihong Xia. 2018. Learning Bidirectional LSTM Networks for Synthesizing 3D Mesh Animation Sequences. arXiv:1810.02042 [cs.GR]
- Dafei Qin, Jun Saito, Noam Aigerman, Groueix Thibault, and Taku Komura. 2023. Neural Face Rigging for Animating and Retargeting Facial Meshes in the Wild. In *SIGGRAPH 2023 Conference Papers*.
- Sigal Raab, Inbal Leibovitch, Guy Tevet, Moab Arar, Amit H Bermano, and Daniel Cohen-Or. 2023. Single motion diffusion. *arXiv preprint arXiv:2302.05905* (2023).
- Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2021. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752 [cs.CV]
- Guodong Rong, Yan Cao, and Xiaohu Guo. 2008. Spectral mesh deformation. *The Visual Computer* 24 (2008).
- Igor Santesteban, Elena Garces, Miguel A Otaduy, and Dan Casas. 2020. SoftSMPL: Data-driven Modeling of Nonlinear Soft-tissue Dynamics for Parametric Humans. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 65–75.
- Robert W. Sumner and Jovan Popović. 2004. Deformation Transfer for Triangle Meshes. *ACM Trans. Graph.* 23, 3 (2004), 399–405.
- Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. 2019. Deep High-Resolution Representation Learning for Human Pose Estimation. In *CVPR*.
- Yifan Sun and Noboru Murata. 2020. CAFM: A 3D Morphable Model for Animals. In *2020 IEEE Winter Applications of Computer Vision Workshops (WACVW)*. 20–24. doi:10.1109/WACVW50321.2020.9096941
- Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. 2020. RigNet: Neural Rigging for Articulated Characters. *ACM Trans. on Graphics* 39 (2020).
- Wang Yifan, Noam Aigerman, Vladimir G. Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. 2020. Neural Cages for Detail-Preserving 3D Deformations. In *CVPR*.
- Jiaxu Zhang, Junwu Weng, Di Kang, Fang Zhao, Shaoli Huang, Xuefei Zhe, Linchao Bao, Ying Shan, Jue Wang, and Zhigang Tu. 2023. Skinned motion retargeting with residual perception of motion semantics & geometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13864–13872.
- Keyang Zhou, Bharat Lal Bhatnagar, and Gerard Pons-Moll. 2020. Unsupervised Shape and Pose Disentanglement for 3D Meshes. In *European Conference on Computer Vision (ECCV)*.