BANMo: Building Animatable 3D Neural Models from Many Casual Videos

Gengshan Yang[2*] Minh Vo[3] Natalia Neverova[1] Deva Ramanan[2] Andrea Vedaldi[1] Hanbyul Joo[1]
[1]Meta AI [2]Carnegie Mellon University [3]Reality Labs

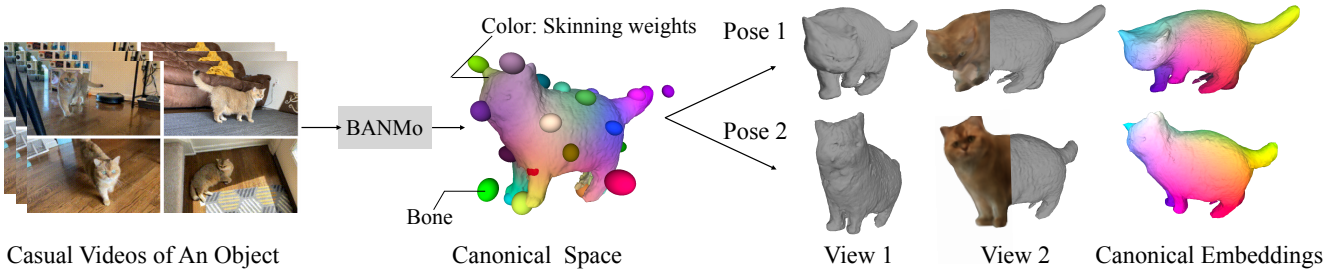

Figure 1. Given multiple casual videos capturing a deformable object ($\approx 10^3$ frames), BANMo reconstructs an animatable 3D model, including an implicit canonical 3D shape, appearance, skinning weights, and time-varying articulations, without pre-defined shape templates or registered cameras. (Left) Input videos; (Middle) 3D shape, bones, and skinning weights (visualized as surface colors) defined in the canonical space; (Right) Posed reconstruction at each time instance with 3D shape, color, and canonical embeddings (correspondences are shown as the same colors).

Abstract

Prior work for articulated 3D shape reconstruction often relies on specialized sensors (e.g., synchronized multi-camera systems), or pre-built 3D deformable models (e.g., SMAL or SMPL). Such methods are not able to scale to diverse sets of objects in the wild. We present BANMo, a method that requires neither a specialized sensor nor a pre-defined template shape. BANMo builds high-fidelity, articulated 3D "models" (including shape and animatable skinning weights) from many monocular casual videos in a differentiable rendering framework. While the use of many videos provides more coverage of camera views and object articulations, they introduce significant challenges in establishing correspondence across scenes with different backgrounds, illumination conditions, etc. Our key insight is to merge three schools of thought; (1) classic deformable shape models that make use of articulated bones and blend skinning, (2) volumetric neural radiance fields (NeRFs) that are amenable to gradient-based optimization, and (3) canonical embeddings that generate correspondences between pixels and an articulated model. We introduce neural blend skinning models that allow for differentiable and invertible articulated deformations. When combined with canonical embeddings, such models allow us to establish dense correspondences across videos that can be self-supervised with cycle consistency. On real and synthetic datasets, BANMo shows higher-fidelity 3D reconstructions than prior works for humans and animals, with the ability to render realistic images from novel viewpoints and poses. Project webpage: banmo-www.github.io.

1. Introduction

We are interested in developing tools that can reconstruct accurate and animatable models of 3D objects from casually collected videos. A representative application is content creation for virtual and augmented reality, where the goal is to 3D-ify images and videos captured by users for consumption in a 3D space or creating animatable assets such as avatars. For rigid scenes, traditional Structure from Motion (SfM) approaches can be used to leverage large collection of uncontrolled images, such as images downloaded form the web, to build accurate 3D models of landmarks and entire cities [1, 40, 41]. However, these approaches do not generalize to deformable objects such as family members, friends or pets, which are often the focus of user content.

We are thus interested in reconstructing 3D deformable objects from *casually collected videos*. However, individual videos may not contain sufficient information to obtain

*Work done when interning at Meta AI

good reconstruction of a given subject. Fortunately, we can expect that users may collect several videos of the same subjects, such as filming a family member over the span of several months or years. In this case, we wish our system to pool information from *all available videos* into a single 3D model, bridging any time discontinuity.

In this paper, we present **BANMo**, a **B**uilder of **A**nimatable 3D **N**eural **Mo**dels from multiple casual RGB videos. By consolidating the 2D cues from thousands of images into a fixed canonical space, BANMo learns a high-fidelity neural implicit model for appearance, 3D shape, and articulations of the target non-rigid object. The articulation of the output model of BANMo is expressed by a neural blend skinning, similar to [5, 54, 55], making the output *animatable* by manipulating bone transformations. As shown in NRSfM [4], reconstructing a freely moving non-rigid object from monocular video is a highly under-constrained task where epipolar constraints are not directly applicable. In our approach, we address three core challenges: (1) how to represent 3D appearance and deformation of the target model in a canonical space; (2) how to find the mapping between canonical space to each individual frame; (3) how to find 2D correspondences across images under view and light changes, and object deformations.

Concretely, we utilize neural implicit functions [26] to represent color and 3D surface in the canonical space. This representation enables higher-fidelity 3D geometry reconstruction compared to approaches based on 3D meshes [54, 55]. The use of neural blending skinning in BANMo provides a way to constrain the deformation space of the target object, allowing better handling of pose pose variations and deformations with unknown camera parameters, compared to dynamic NeRF approaches [5, 19, 30, 35]. We also present a module for fine-grained registration between pixels and the canonical space by matching to an implicit feature volume. To jointly optimize over a large number of video frames with a manageable computational cost, we actively sample pixels locations based on uncertainty. In a nutshell, BANMo presents a way to merge the recent non-rigid object reconstruction approaches [54, 55] in a dynamic NeRF framework [5, 19, 30, 35], to achieve higher-fidelity non-rigid object reconstruction. We show experimentally that BANMo produces higher-fidelity 3D shape details than previous state-of-the art approaches [55], by taking better advantage of the large number of frames in multiple videos.

Our contributions are summarized as follows: (1) BANMo allows for high-fidelity reconstruction of animatable 3D models from casual videos containing thousands of frames; (2) To handle large articulated body motions, we propose a novel neural blend skinning model (Sec. 3.2); (3) To register pixels from different frames, we introduce a method for canonical embedding matching and learning (Sec. 3.3); (4) To robustify the optimization, we propose an pipeline for rough root body pose initialization (Sec. 3.4); (5) Finally, to reconstruct detailed geometry, we introduce an active sampling strategy for inverse rendering (Sec. 3.4).

2. Related work

Human and animal body models. A large body of work in 3D human and animal reconstruction uses parametric shape models [22, 32, 51, 61, 62], which are built from registered 3D scans of real humans or toy animals, and serve to recover 3D shapes given a single image and 2D annotations or predictions (2D keypoints and silhouettes) at test time [2, 3, 12, 12, 60]. Although parametric body models achieve great success in reconstructing categories for which large amounts of ground-truth 3D data are available (mostly in the case of human reconstruction), it is challenging to apply the same methodology to categories with limited 3D data, such as animals and humans in diverse sets of clothing.

Category reconstruction from image/video collections. A number of recent methods build deformable 3D models of object categories from images or videos with weak 2D annotations, such as keypoints, object silhouettes, and optical flow, obtained from human annotators or predicted by off-the-shelf models [6, 9, 13, 17, 18, 50, 58]. Such methods often rely on a coarse shape template [15, 46], and are not able to recover fine-grained details or large articulations.

Category-agnostic video shape reconstruction. Non-rigid structure from motion (NRSfM) methods [4, 7, 14, 16, 38] reconstruct non-rigid 3D shapes from a set of 2D point trajectories in a class-agnostic way. However, due to difficulties in obtaining accurate long-range correspondences [37, 44], they do not work well for videos in the wild. Recent efforts such as LASR and ViSER [54, 55] reconstruct articulated shapes from a monocular video with differentiable rendering. As our results show, they may still produce blurry geometry and unrealistic articulations.

Neural radiance fields. Prior works on NeRF optimize a continuous scene function for novel view synthesis given a set of images, often assuming the scene is rigid and camera poses can be accurately registered to the background [8, 20, 24–26, 49]. To extend NeRF to dynamic scenes, recent works introduce additional functions to deform observed points to a canonical space or over time [19, 30, 31, 35, 45]. However, they heavily rely on background registration, and fail when the motion between objects and background is large. Moreover, the deformations cannot be explicitly controlled by user inputs. Similar to our goal, some recent works [21, 29, 33, 34, 42] produce pose-controllable NeRFs, but they rely on a human body model, or synchronized multi-view video inputs.

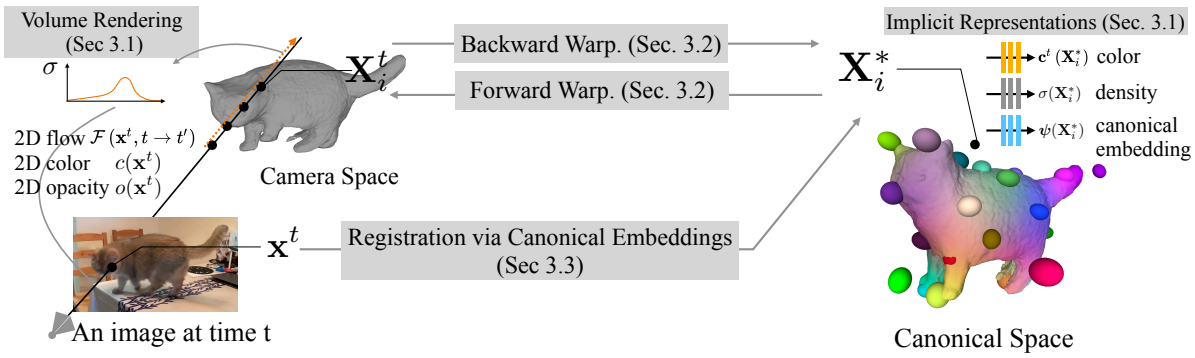

Figure 2. **Method overview.** BANMo optimizes a set of shape and deformation parameters (Sec. 3.1) that describe the video observations in pixel colors, silhouettes, optical flow, and higher-order features descriptors, based on a differentiable volume rendering framework. BANMo uses a neural blend skinning model (Sec. 3.2) to transform 3D points between the camera space and the canonical space, enabling handling large deformations. To register pixels across videos, BANMo jointly optimizes an implicit canonical embedding (CE) (Sec. 3.3).

3. Method

We model the deformable object in a canonical time-invariant space, i.e. the "rest" body pose space, that can be transformed to the "articulated" pose in the camera space at each time instance with forward mappings, and transform back with backward mappings. We use implicit functions to represent the 3D shape, color, and dense semantic embeddings of the object. Our neural 3D model can be deformed and rendered into images at each time instance via differentiable volume rendering, and optimized to ensure consistency between the rendered images and multiple cues in the observed images, including color, silhouette, optical flow, and 2D pixel feature embeddings. We refer readers to an overview in Fig. 2 and a list of notations in the supplement.

We employ neural blend skinning to express object articulations similarly to [15, 54] but modify it for implicit surface representations rather than meshes. Our self-supervised semantic feature embedding produces dense pixelwise correspondences across frames of different videos, which is critical for optimization on large video collections.

3.1. Shape and Appearance Model

We first represent shape and appearance of deformable objects in a canonical time-invariant *rest pose* space and then model time-dependent deformations by a neural blend skinning function (Sec. 3.2).

Canonical shape model. In order to model the shape and appearance of an object in a canonical space, we use a method inspired by Neural Radiance Fields (NeRF) [26]. A 3D point $\mathbf{X}^* \in \mathbb{R}^3$ in a canonical space is associated with three properties: color $\mathbf{c} \in \mathbb{R}^3$, density $\sigma \in [0,1]$, and a learned canonical embedding $\boldsymbol{\psi} \in \mathbb{R}^{16}$. These properties are predicted by the Multilayer Perceptron (MLP) networks:

$$\mathbf{c}^t = \mathbf{MLP}_{\mathbf{c}}(\mathbf{X}^*, \mathbf{v}^t, \boldsymbol{\omega}_e^t), \quad (1)$$

$$\sigma = \Gamma_\beta(\mathbf{MLP}_{\text{SDF}}(\mathbf{X}^*)), \quad (2)$$

$$\boldsymbol{\psi} = \mathbf{MLP}_{\boldsymbol{\psi}}(\mathbf{X}^*). \quad (3)$$

As in NeRF, color $\mathbf{c}^t$ depends on a time-varying view direction $\mathbf{v}^t \in \mathbb{R}^2$ and a learnable environment code $\boldsymbol{\omega}_e^t \in \mathbb{R}^{64}$ that is designed to capture environment illumination conditions [24], shared across frames in the same video.

The shape is given by $\mathbf{MLP}_{\text{SDF}}$, computing the Signed-Distance Function (SDF) of a point to the surface. For rendering, we follow [48, 56] and convert the SDF to a density value $\sigma \in [0,1]$ using $\Gamma_\beta(\cdot)$, the cumulative distribution function of the Laplace distribution with zero mean and β scale. β is a single learnable parameter that controls the solidness of the object, approaching zero for solid objects. Compared to learning σ directly, this provides a principled way of extracting surface as the zero level-set of the SDF.

Finally, the network $\boldsymbol{\psi}$ maps points to a feature descriptor (or canonical embedding) that can be matched by pixels from different viewpoints, enabling long-range correspondence across frames and videos. This feature can be interpreted as a variant of Continuous Surface Embeddings (CSE) [27] but defined volumetrically and fine-tuned in a self-supervised manner (described in Sec 3.3).

Space-time warping model. We consider a pair of time-dependent warping functions: *forward warping function* $\mathcal{W}^{t,\rightarrow} : \mathbf{X}^* \rightarrow \mathbf{X}^t$ mapping canonical location $\mathbf{X}^*$ to camera space location $\mathbf{X}^t$ at current time and the *backward warping function* $\mathcal{W}^{t,\leftarrow} : \mathbf{X}^t \rightarrow \mathbf{X}^*$ for inverse mapping.

Prior work such as Nerfies [30] and Neural Scene Flow Fields (NSFF) [19] learn deformations assuming that (1) camera transformations are given, and (2) the residual object deformation is small. As detailed in Sec. 3.2 and Sec. 3.4, we do not make such assumptions; instead, we adopt a neural blend-skinning model that can handle large deformations, but without assuming a pre-defined skeleton.

Volume rendering. To render images, we use the volume rendering in NeRF [26], but modified to warp the 3D ray to account for the deformation [30]. Specifically, let $\mathbf{x}^t \in \mathbb{R}^2$ be the pixel location at time t, and $\mathbf{X}_i^t \in \mathbb{R}^3$ be the i-th 3D point sampled along the ray emanates from $\mathbf{x}^t$. The color $\mathbf{c}$ and the opacity $o \in [0, 1]$ of the pixel are given by:

$$\mathbf{c}(\mathbf{x}^t) = \sum_{i=1}^{N} \tau_i \mathbf{c}^t \left(\mathcal{W}^{t,\leftarrow}\left(\mathbf{X}_i^t\right)\right), \quad o(\mathbf{x}^t) = \sum_{i=1}^{N} \tau_i,$$

where N is the number of samples, τ_i is the probability $\mathbf{X}_i^t$ visible to the camera and is given by $\tau_i = \prod_{j=1}^{i-1} p_j(1-p_i)$. Here $p_i = \exp\left(-\sigma_i \delta_i\right)$ is the probability that the photon is transmitted through the interval δ_i between the i-th $\mathbf{X}_i^t$ sample and the next, and $\sigma_i = \sigma\left(\mathcal{W}^{t,\leftarrow}\left(\mathbf{X}_i^t\right)\right)$ is the density from Eq. 2. Note that we *pull back* the ray points in the observed space to the canonical space by using the warping $\mathcal{W}^{t,\leftarrow}$, as the color and density are only defined in the canonical space.

We obtain the expected 3D location $\mathbf{X}^*$ of the intersection of the ray with the canonical object:

$$\mathbf{X}^*(\mathbf{x}^t) = \sum_{i=1}^{N} \tau_i \left(\mathcal{W}^{t,\leftarrow}(\mathbf{X}_i^t)\right), \tag{4}$$

which can be mapped to another time t' via forward warping $\mathcal{W}^{t,\rightarrow}$to find its projected pixel location:

$$\mathbf{x}^{t'} = \Pi^{t'}\left(\mathcal{W}^{t,\rightarrow}(\mathbf{X}^*(\mathbf{x}^t))\right), \tag{5}$$

where $\Pi^{t'}$ is the projection matrix of a pinhole camera. We optimize video-specific $\Pi^{t'}$ given a rough initialization. With this, we compute a 2D flow rendering as:

$$\mathcal{F}\left(\mathbf{x}^t, t \rightarrow t'\right) = \mathbf{x}^{t'} - \mathbf{x}^t. \tag{6}$$

3.2. Deformation Model via Neural Blend Skinning

We define mappings $\mathcal{W}^{t,\rightarrow}$ and $\mathcal{W}^{t,\leftarrow}$ based on a neural blend skinning model approximating articulated body motion. Defining invertible warps for neural deformation representations is difficult [5]. Our formulation is conceptually similar to SCANimate [36] that represents 3D warps as compositions of weighted *rigid-body transformations*, each of which is differentiable and invertible.

Blend skinning deformation. Given a 3D point $\mathbf{X}^t$ at time t, we wish to find its corresponding 3D point $\mathbf{X}^*$ in the canonical space. Conceptually, $\mathbf{X}^*$ can be considered as points in the "rest" pose at a fixed camera view point. Our formulation finds mappings between $\mathbf{X}^t$ and $\mathbf{X}^*$ by blending the rigid transformations of B bones (3D coordinate systems). Let $\mathbf{G}^t \in SE(3)$ be a root body transformation of the object from canonical to time t, and $\mathbf{J}_b^t \in SE(3)$ be a rigid transformation that moves the b-th bone from its canonical to deformed state t, then we have

$$\mathbf{X}^t = \mathcal{W}^{t,\rightarrow}(\mathbf{X}^*) = \mathbf{G}^t\left(\mathbf{J}^{t,\rightarrow}\mathbf{X}^*\right), \tag{7}$$

$$\mathbf{X}^* = \mathcal{W}^{t,\leftarrow}(\mathbf{X}^t) = \mathbf{J}^{t,\leftarrow}\left((\mathbf{G}^t)^{-1}\mathbf{X}^t\right) \tag{8}$$

The soft-blending of $\mathbf{J}_t$ is computed similar to linear blend skinning (LBS) deformation with B rigid transformations:

$$\mathbf{J}^{t,\rightarrow} = \sum_{b=1}^{B} \mathbf{W}_b^{t,\rightarrow}\mathbf{J}_b^t, \quad \mathbf{J}^{t,\leftarrow} = \sum_{b=1}^{B} \mathbf{W}_b^{t,\leftarrow}(\mathbf{J}_b^t)^{-1}, \tag{9}$$

where the $\mathbf{J}^{t,\rightarrow}$ and $\mathbf{J}^{t,\leftarrow}$ define weighted averages of B rigid-body transformations and $\mathbf{W}_b^{t,\rightarrow}$ and $\mathbf{W}_b^{t,\leftarrow}$ represent linear blend skinning (LBS) weights for the point $\mathbf{X}^*$ and $\mathbf{X}^t$ relative to the b-th bone (described further below). We represent $\mathbf{G}^t$ and $\mathbf{J}_b^t$ using angle-axis rotations and 3D translations. These are regressed from two MLPs with 128-dimensional latent frame codes each: $\mathbf{G}^t = \mathbf{MLP}_{\mathbf{G}}(\boldsymbol{\omega}_r^t)$ and $\{\mathbf{J}_b^t\}_{b\in B} = \mathbf{MLP}_{\mathbf{J}}(\boldsymbol{\omega}_b^t)$, where $\boldsymbol{\omega}_r^t$ and $\boldsymbol{\omega}_b^t$ are the root pose and body pose latent code at frame t respectively. We find such over-parameterized representations empirically easier to optimize with SGD.

Skinning weights. Given a 3D point $\mathbf{X}$ and a body pose code $\boldsymbol{\omega}_b$, we define a function $\mathcal{S}$ to compute the skinning weights $\mathbf{W} \in \mathbb{R}^B$ that assigns $\mathbf{X}$ to bones. The weights consist of an explicit component $\mathbf{W}_\sigma$ and a delta component $\mathbf{W}_\Delta$ to model coarse and fine assignments respectively:

$$\mathbf{W} = \mathcal{S}(\mathbf{X}, \omega_b) = \sigma_{\text{softmax}}\left(\mathbf{W}_\sigma + \mathbf{W}_\Delta\right). \tag{10}$$

Similar to [54], the explicit component $\mathbf{W}_\sigma$ is computed as the distance between $\mathbf{X}$ and each bone center $\mathbf{C}_b$:

$$\mathbf{W}_\sigma = -\alpha_w ||\mathbf{X} - \mathbf{C}_b||^2, \tag{11}$$

where α_w is a learnable scalar that controls the peakness of the assignment. Bone centers at time t are computed as $\mathbf{C}_b^t = \mathbf{J}_b^t\mathbf{C}_b^*$, where $\mathbf{C}_b^*$ are learnable bone centers at the rest pose. To model skinning weights for complex geometry, we additionally learn delta skinning weights $\mathbf{W}_\Delta = \mathbf{MLP}_\Delta(\mathbf{X}, \boldsymbol{\omega}_b)$. With the skinning function, the forward and backward skinning weights in Eq. 9 are computed as $\mathbf{W}^{t,\rightarrow} = \mathcal{S}(\mathbf{X}^*, \boldsymbol{\omega}_b^*)$ and $\mathbf{W}^{t,\leftarrow} = \mathcal{S}(\mathbf{X}^t, \boldsymbol{\omega}_b^t)$, where $\boldsymbol{\omega}_b^*$ and $\boldsymbol{\omega}_b^t$ are the body pose code for the rest pose and the time t pose. By construction, the skinning weights are forced to be dependent on only pose status. Thus, our formulation regularizes the space of skinning weights, and handles large deformations better than purely implicitly-defined ones.

3.3. Registration via Canonical Embeddings

To register pixel observations at different time instances, BANMo maintains a canonical feature embedding that encodes semantic information of 3D points in the canonical

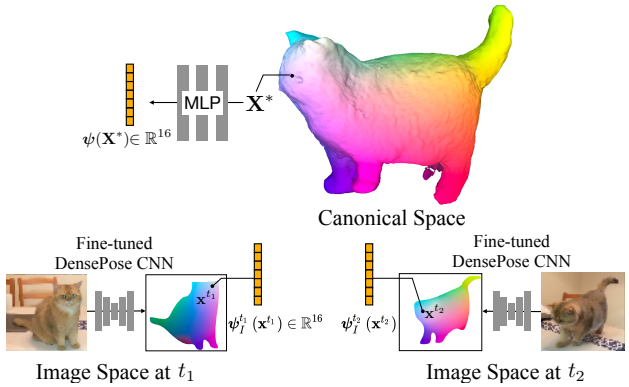

Figure 3. **Canonical Embeddings.** We jointly optimize an implicit function to produce canonical embeddings from 3D canonical points that match to the 2D DensePose CSE embeddings [27].

space, which can be uniquely matched by the pixel features, and provide strong cues for registration via a joint optimization of shape, articulations, and embeddings (Sec. 3.4).

Canonical embeddings matching. Given a pixel at $\mathbf{x}^t$ of frame t, our goal is to find a point $\mathbf{X}^*$ in the canonical space whose feature embedding $\psi(\mathbf{X}^*) \in \mathbb{R}^{16}$ best matches the pixel feature embedding $\psi_I^t(\mathbf{x}^t) \in \mathbb{R}^{16}$. The pixel embeddings ψ_I^t (of frame t) are computed by a CNN. Different from ViSER [55] that learns embeddings from scratch, we initialize pixel embeddings with CSE [27, 28] that produces consistent features for semantically corresponding pixels, and optimize pixel and canonical embeddings jointly. Recall that the embedding of a canonical 3D point is computed as $\psi(\mathbf{X}^*) = \mathbf{MLP}_\psi(\mathbf{X}^*)$ in Eq. 3. Intuitively, $\mathbf{MLP}_\psi$ is optimized to ensure the output 3D descriptor matches 2D descriptors of corresponding pixels across multiple views. To compute the 3D surface point corresponding to a 2D point $\mathbf{x}^t$, we apply soft argmax descriptor matching [10, 23]:

$$\hat{\mathbf{X}}^*(\mathbf{x}^t) = \sum_{\mathbf{X} \in \mathbf{V}^*} \tilde{\mathbf{s}}^t(\mathbf{x}^t, \mathbf{X})\mathbf{X}, \tag{12}$$

where $\mathbf{V}^*$ are sampled points in a canonical 3D grid, whose size is dynamically determined during optimization (see supplement), and $\tilde{\mathbf{s}}$ is a normalized feature matching distribution over the 3D grid: $\tilde{\mathbf{s}}^t(\mathbf{x}^t, \mathbf{X}) = \sigma_{\text{softmax}}\Big(\alpha_s \big\langle \psi_I^t(\mathbf{x}^t), \psi(\mathbf{X}^*) \big\rangle\Big)$, where α_s is a learnable scaling to control the peakness of the softmax function and $\langle ., . \rangle$ is the cosine similarity score.

Self-supervised canonical embedding learning. As describe later in Eq. 14-15, the canonical embedding is self-supervised by enforcing the consistency between feature matching and geometric warping. By jointly optimizing the shape and articulation parameters via consistency losses, canonical embeddings provide strong cues to register pixels from different time instance to the canonical 3D space, and enforce a coherent reconstruction given observations from multiple videos, as validated in ablation studies (Sec. 4.3).

3.4. Optimization

Given frames from multiple videos, we optimize all parameters described above, including MLPs, $\mathbf{MLP}_\mathbf{c}$, $\mathbf{MLP}_{\text{SDF}}$, $\mathbf{MLP}_\psi$, $\mathbf{MLP}_\mathbf{G}$, $\mathbf{MLP}_\mathbf{J}$, $\mathbf{MLP}_\Delta$, learnable codes $\boldsymbol{\omega}_e^t$, $\boldsymbol{\omega}_r^t$, $\boldsymbol{\omega}_b^t$, and $\boldsymbol{\omega}_b^*$, and pixel embeddings ψ_I.

Losses. The model is learned by minimizing three types of losses: reconstruction losses, feature registration losses, and a 3D cycle-consistency regularization loss:

$$\mathcal{L} = \underbrace{\big(\mathcal{L}_{\text{sil}} + \mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{OF}}\big)}_{\text{reconstruction losses}} + \underbrace{\big(\mathcal{L}_{\text{match}} + \mathcal{L}_{\text{2D-cyc}}\big)}_{\text{feature registration losses}} + \mathcal{L}_{\text{3D-cyc}}.$$

Reconstruction losses are similar to those in existing differentiable rendering pipelines [26, 57]. Besides color reconstruction loss $\mathcal{L}_{\text{rgb}}$ and silhouette reconstruction loss $\mathcal{L}_{\text{sil}}$, we further compute flow reconstruction losses $\mathcal{L}_{\text{OF}}$ by comparing the rendered $\mathcal{F}$ defined in Eq. 6 with the observed 2D optical flow $\hat{\mathcal{F}}$ computed by an off-the-shelf flow network:

$$\mathcal{L}_{\text{rgb}} = \sum_{\mathbf{x}^t} \big\|\mathbf{c}(\mathbf{x}^t) - \hat{\mathbf{c}}(\mathbf{x}^t)\big\|^2, \ \mathcal{L}_{\text{sil}} = \sum_{\mathbf{x}^t} \big\|\mathbf{o}(\mathbf{x}^t) - \hat{\mathbf{s}}(\mathbf{x}^t)\big\|^2,$$

$$\mathcal{L}_{\text{OF}} = \sum_{\mathbf{x}^t, (t,t')} \Big\|\mathcal{F}\left(\mathbf{x}^t, t \to t'\right) - \hat{\mathcal{F}}\left(\mathbf{x}^t, t \to t'\right)\Big\|^2, \tag{13}$$

where $\hat{\mathbf{c}}$ and $\hat{\mathbf{s}}$ are observed color and silhouette.

Additionally, we define registration losses to enforce 3D point prediction via canonical embedding $\hat{\mathbf{X}}^*(\mathbf{x}^t)$ (Eq. 12) to match the prediction from backward warping (Eq. 4):

$$\mathcal{L}_{\text{match}} = \sum_{\mathbf{x}^t} \Big\|\mathbf{X}^*(\mathbf{x}^t) - \hat{\mathbf{X}}^*(\mathbf{x}^t)\Big\|_2^2, \tag{14}$$

and a 2D cycle consistency loss [15, 55] that forces the image projection after forward warping of $\hat{\mathbf{X}}^*(\mathbf{x}^t)$ to land back on its original 2D coordinates:

$$\mathcal{L}_{\text{2D-cyc}} = \sum_{\mathbf{x}^t} \Big\|\Pi^t\Big(\mathcal{W}^{t,\rightarrow}(\hat{\mathbf{X}}^*(\mathbf{x}^t))\Big) - \mathbf{x}^t\Big\|_2^2. \tag{15}$$

Similar to NSFF [19], we regularize the deformation function $\mathcal{W}^{t,\rightarrow}(\cdot)$ and $\mathcal{W}^{t,\leftarrow}(\cdot)$ by a 3D cycle consistency loss, which encourages a sampled 3D point in the camera coordinates to be backward deformed to the canonical space and forward deformed to its original location:

$$\mathcal{L}_{\text{3D-cyc}} = \sum_i \tau_i \Big\|\mathcal{W}^{t,\rightarrow}\Big(\mathcal{W}^{t,\leftarrow}(\mathbf{X}_i^t, t), t\Big) - \mathbf{X}_i^t\Big\|_2^2, \tag{16}$$

where τ_i is the opacity that weighs the sampled points so that a point near the surface receives heavier regularization.

Our optimization is highly non-linear with local minima, and we consider two strategies for robust optimization.

Root pose initialization. Due to ambiguities between object geometry and root poses, we find it helpful to provide a rough per-frame initialization of root poses ($\mathbf{G}^t$ in Eq. 7), similar to NeRS [59]. Specifically, we train a separate network PoseNet , which is applied to every test video frame. Similar to DenseRaC [52], PoseNet takes a DensePose CSE [27] feature image as input and predicts the root pose $\mathbf{G}_0^t = \text{PoseNet}(\boldsymbol{\psi}_I^t)$, where $\boldsymbol{\psi}_I^t \in \mathbb{R}^{112\times112\times16}$ is the embedding output of DensePose CSE [27] from an RGB image I_t. We train PoseNet by a synthetic dataset produced offline. See supplement for details on training. Given the pre-computed $\mathbf{G}_0^t$, BANMo only needs to compute a delta root pose via MLP:

$$\mathbf{G}^t = \mathbf{MLP}_{\mathbf{G}}(\boldsymbol{\omega}_r^t)\mathbf{G}_0^t. \tag{17}$$

Active sampling over (x, y). Inspired by iMAP [43], our sampling strategy follows an easy-to-hard curriculum. At the early iterations, we randomly sample a batch of $N^p = 16384$ pixels for volume rendering and compute reconstruction losses. At the same time, we optimize a compact 5-layer MLP to represent the uncertainty over image coordinates and frame index, by comparing against the color reconstruction errors in the current batch. More details can be found in the supplement. After a half of the optimization steps, we select $N^a = 2048$ additional *active* samples from pixels with high uncertainties, and combine with the uniform samples. Empirically, active samples dramatically improves reconstruction fidelity, as shown in Fig. 8.

4. Experiments

Implementation details. Our implementation of implicit shape and appearance models follows NeRF [26], except that our shape model outputs SDF, which is transformed to density for volume rendering. To extract the rest surface, we find the zero-level set of SDF by running marching cubes on a 256^3 grid. To obtain articulated shapes at each time instance, we articulate points on the rest surface with forward deformation $\mathcal{W}^{t,\rightarrow}$. We use $B = 64$ bones, whose centers are initialized around an approximation of object surface.
Optimization details. In a single batch, we sample $N^b = 32$ pairs of images and sub-sample $N^p = 16384$ pixels for volume rendering. Empirically, we found the number of iterations to achieve the same level of reconstruction quality scales with the number of input frames. To determine the number of iterations, we use an empirical equation: $N^{opt} = 2000\frac{\#\text{ frames}}{N^b}$, roughly 60k iterations for 1000 frames (15 hours on 8 V100 GPUs). Please find a complete list of hyper-parameters for optimization in supplement.

4.1. Reconstruction from Casual Videos

Casual videos dataset. We demonstrate BANMo's ability to reconstruct 3D models on collections of casual videos, including 20 videos of a British Shorthair cat and 10 videos of a human, denoted by `casual-cat` and `casual-human` respectively. Object silhouette and two-frame optical flow images (used for reconstruction losses Eq. 13) are extracted from off-the-shelf models, PointRend and VCN-robust respectively [11, 53]. Two special challenges arise from the casual nature of the video captures. First, each video collection contains around 2k images, orders of magnitudes larger those used in prior work [19, 26, 30, 55], which requires the method to handle reconstructions at a larger scale. Second, the dataset makes no assumption about camera movement or object movement, making registration difficult. In particular, objects freely moves in a video and background changes across videos, posing challenges to standard SfM pipelines. We show results on `casual-cat` below and `casual-human` in the supplement.

Comparison with Nerfies. Nerfies [30] is designed for a single continuously captured video, assuming object root body does not move and root poses can be registered by SfM. However, in our setup object root body moves and SfM does not provide registration of root poses, causing Nerfies to fail. To make a fair comparison, we provide Nerfies with rough initial root poses (obtained from our PoseNet, Sec. 3.4), and optimize Nerfies on a per-video basis. Meshes are extracted by running marching cubes on a 256^3 grid. We first compare single-video BANMo with Nerfies in Fig. 4. Although Nerfies reconstructs reasonable 3D shapes of moving objects given rough initial root pose, it fails to reconstruct large articulations, such as the fast motion of the cat's head (2nd row), and in general the reconstruction is coarser than single-video BANMo. Furthermove, as shown in Fig. 5, Nerfies is not able to leverage more videos to improve the reconstruction quality, while the reconstruction of BANMo improves given more videos.

Comparison with ViSER. Another baseline, ViSER [55], reconstructs articulated objects given multiple videos. However, it assumes root body poses are given by running LASR, which produces less robust root pose estimations than our PoseNet. Therefore, we provide ViSER the same root poses from our initialization pipeline. As shown in Fig. 4, ViSER produces reasonable articulated shapes. However, detailed geometry, such as ears, eyes, nose and rear limbs of the cat are blurred out. Furthermore, detailed articulation, such as head rotation and leg switching are not recovered. In contrast, BANMo faithfully recovers these high-fidelity geometry and motion. We observed that implicit shape representation (specifically VolSDF [56] with a gradually reducing transparency value) is less prune to bad local optima, which helps recovering body parts and articulations. Furthermore, implicit shape representation allows for maintaining a continuous geometry (compared to finite number of vertices), enabling us to recover detailed shape.

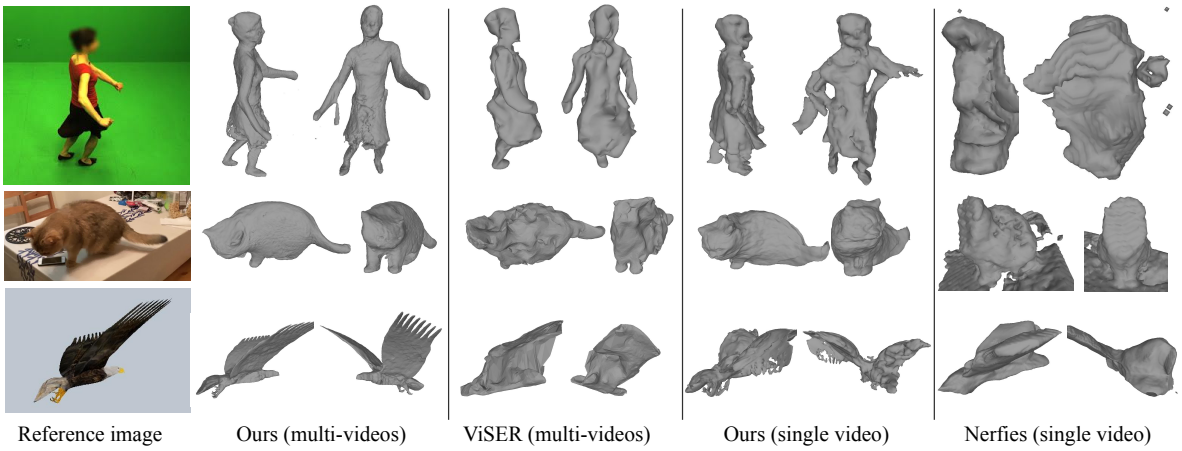

Figure 4. **Qualitative comparison of our method with prior art [30, 55].** From top to bottom: AMA's `samba`, `casual-cat`, `eagle`.

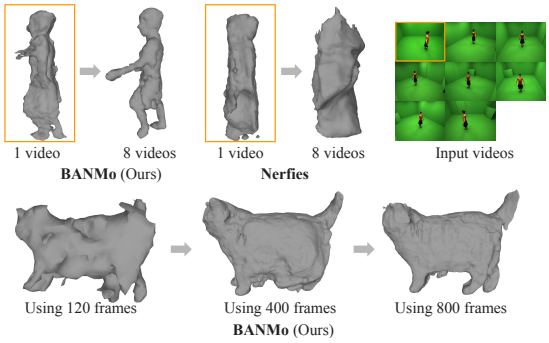

Figure 5. **Reconstruction completeness vs number of input videos and video frames.** BANMo is capable of registering more input videos if they are available, improving the reconstruction.

4.2. Quantitative Evaluation on Multi-view Videos

We use two sources of data with 3D ground truth for quantitative evaluation of our method.

AMA human dataset. For human reconstruction, we use the Articulated Mesh Animation (AMA) dataset [47]. AMA contains 10 sets of multi-view videos captured by 8 synchronized cameras. We select 2 sets of videos of the same actor (`swing` and `samba`), totaling 2600 frames, and treat them as unsynchronized monocular videos as the input. Time synchronization and camera extrinsics are *not* used for optimization. We use the ground-truth object silhouettes and the optical flow estimated by VCN-robust [53].

Animated Objects dataset. To quantitatively evaluate on other object categories, we download free animated 3D models from TurboSquid, including an `eagle` model and a model for human `hands`. We render them from different camera trajectories with partially overlapping motions. Each animated object is rendered as 5 videos with 150 frames per video. We provide ground-truth root poses (or camera poses) and ground-truth silhouettes to BANMo and baselines. Optical flow is computed by VCN-robust [53].

Table 1. **Quantitative results on AMA and Animated Objects.** 3D Chamfer distances (cm) averaged over all frames (↓). The 3D models for `eagle` and `hands` are resized such that the largest edge of the axis-aligned object bounding box is 2m. ** with ground-truth root poses.* (S) refers to single-video results.

Method	`swing`	`samba`	mean	`eagle`	`hands`
Ours	**5.95**	**5.86**	**5.91**	**3.93***	**3.10***
ViSER	15.16	16.31	15.74	19.72*	7.42*
Ours (S)	6.30	6.31	6.31	6.29*	5.96*
Nerfies (S)	11.47	11.36	11.42	8.61*	10.82*

Metric. We quantify our method and baselines against the 3D ground-truth in terms of per-frame reconstruction errors estimated as Chamfer distances between the ground-truth mesh points $\mathbf{S}^*$ and the estimated mesh points $\hat{\mathbf{S}}$:

$$d_{\text{CD}}\left(\mathbf{S}^*, \hat{\mathbf{S}}\right) = \Big(\sum_{x \in \mathbf{S}^*} \min_{y \in \hat{\mathbf{S}}} \|x-y\|_2^2\Big) + \Big(\sum_{y \in \hat{\mathbf{S}}} \min_{x \in \mathbf{S}^*} \|x-y\|_2^2\Big).$$

Before computing d_{CD}, we align the estimated shape to the ground-truth via Iterative Closest Point (ICP) up to a 3D similarity transformation.

Results. We compare BANMo against ViSER and Nerfies in Tab. 1. To set up a fair comparison with Nerfies, we run both methods on a per-video basis, denoted as Ours (S) and Nerfies (S). We found Nerfies that uses RGB reconstruction loss only does not produce meaningful results due to the homogeneous background color. To improve Nerfies results, we provide it with ground-truth object silhouettes, and optimize a carefully balanced RGB+silhouette loss [57]. Still, our method produces lower errors than ViSER and Nerfies.

4.3. Diagnostics

We ablate the importance of each component, by using a subset of videos. To also ablate root pose initialization and registration, we test on AMA's `samba` and `swing` (325 frames in total). We include exhaustive ablations in supplement, and only highlight crucial aspects of BANMo below.

Root pose initialization. We show the effect of PoseNet for root pose initialization (Sec 3.4) in Fig. 6: without it, the root poses (or equivalently camera poses) collapsed to a degenerate solution after optimization.

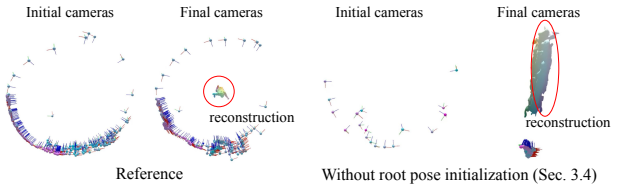

Figure 6. **Diagnostics of root pose initialization (Sec.3.4).** With randomly initialized root poses, the estimated poses (on the right) collapsed to a degenerate solution, causing reconstruction to fail.

Registration. In Fig. 7, we show the benefit of using canonical embeddings (Sec 3.3), and measured 2D flow (Eq. 13) to register observations across videos and within a video. Without the canonical embeddings and corresponding losses (Eq. 14-15), each video will be reconstructed separately. With no flow reconstruction loss, multiple ghosting structures are reconstructed due to failed registration.

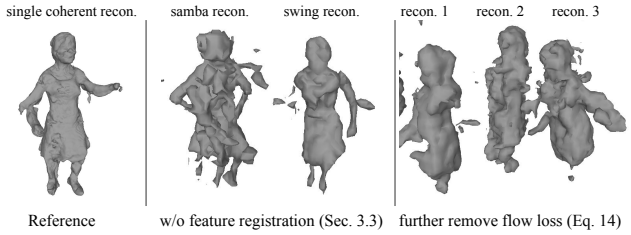

Figure 7. **Diagnostics of registration (Sec. 3.3).** Without canonical embeddings (middle) or flow loss (right), our method fails to register frames to a canonical model, creating ghosting effects.

Active sampling. We show the effect of active sampling (Sec 3.4) on a `casual-cat` video (Fig. 8): removing it results in slower convergence and inaccurate geometry.

Deformation modeling. We demonstrate the benefit of using our neural blend skinning model (Sec 3.2) on an `eagle` sequence, which is challenging due to its large wing articulations. If we swap neural blend skinning for MLP-SE(3) [30], the reconstruction is less regular. If we swap for MLP-translation [19, 35], we observe ghosting wings due to wrong geometric registration (caused by large motion). Our method can model large articulations thanks to the regularization from the Gaussian component, and also handle complex deformation such as close contact of hands.

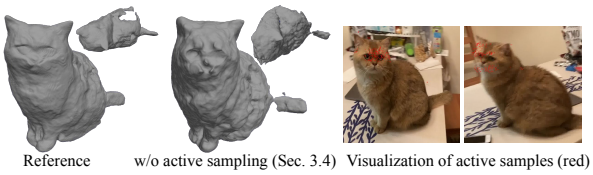

Figure 8. **Diagnostics of active sampling over (x, y) (Sec. 3.4).** With no active sampling, our method converges slower and misses details (such as ears and eyes). Active samples focus on face and boundaries pixels where the color reconstruction errors are higher.

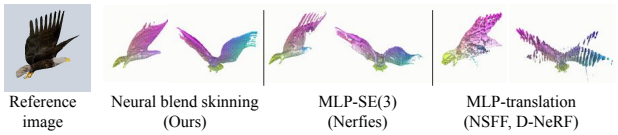

Figure 9. **Diagnostics of deformation modeling (Sec.3.2).** Replacing our neural blend skinning with MLP-SE(3) results in less regular deformation in the non-visible region. Replacing our neural blend skinning with MLP-translation as in NSFF and D-Nerf results in reconstructing ghosting wings due to significant motion.

Ability to leverage more videos. We compare BANMo to Nerfies in terms of the ability to leverage more available video observations. To demonstrate this, we compare the reconstruction quality of optimizing 1 video vs. 8 videos from the AMA `samba` sequences. Results are shown in Fig. 5. Given more videos, our method can register them to the same canonical space, improving the reconstruction completeness and reducing shape ambiguities. In contrast, Nerfies does not produce better results given more video observations.

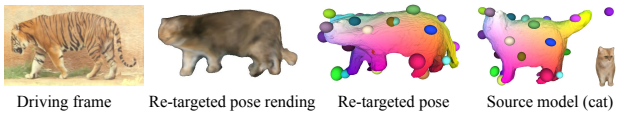

Figure 10. **Motion re-targeting from a pre-optimized cat model to a tiger.** Color coded by point locations in the canonical space.

Motion re-targeting. As a distinctive application, we demonstrate BANMo's ability of to re-target the articulations of a driving video to our 3D model by optimizing the frame-specific root and body pose codes $\boldsymbol{\omega}_r^t$, $\boldsymbol{\omega}_b^t$, as shown in Fig. 10. To do so, we first optimize all parameters over a set of *training* videos from our `casual-cat` dataset. Given a driving video of a tiger, we freeze the shared model parameters (including shape, skinning, and canonical features) of the cat model, and only optimize the video-specific

and frame-specific codes, i.e. root and body pose codes $\boldsymbol{\omega}_r^t$, $\boldsymbol{\omega}_b^t$, as well as the environment lighting code $\boldsymbol{\omega}_e^t$.

5. Discussion

We have presented BANMo, a method to reconstruct high-fidelity animatable 3D models from a collection of casual videos, without requiring a pre-defined shape template or pre-registered cameras. BANMo registers thousands of *unsynchronized* video frames to the same canonical space by fine-tuning a generic DensePose prior to specific instances. We obtain fine-grained registration and reconstruction by leveraging neural implicit representation for shape, appearance, canonical features, and skinning weights. On real and synthetic datasets, BANMo shows strong empirical performance for reconstructing clothed human and quadruped animals, and demonstrates the ability to recover large articulations, reconstruct fine-geometry, and render realistic images from novel viewpoints and poses.

Limitations. BANMo uses a pre-trained DensePose-CSE (with 2D keypoint annotaionts [28]) to provide rough root body pose registration, and therefore not currently applicable to categories beyond humans and quadruped animals. Further investigation is needed to extend our method to arbitrary object categories. Similar to other works in differentiable rendering, BANMo requires a lot of compute, which increases linearly with the number of input images. We leave speeding up the optimization as future work.

A. Notations

We refer readers to a list of notations in Tab. 4 and a list of learnable parameters in Tab. 5.

B. Method details

B.1. Root Pose Initialization

As discussed in Sec. 3.4, to make optimization robust, we train a image CNN (denoted as PoseNet) to initialize root body transforms $\mathbf{G}^t$ that aligns the camera space of time t to the canonical space of CSE, as shown in Fig. 11.

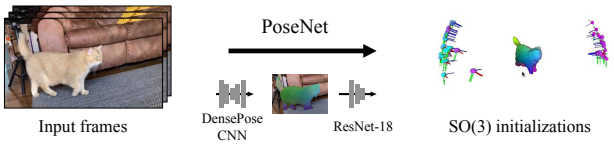

Figure 11. **Inference pipeline of PoseNet.** To initialize the optimization, we train a CNN PoseNet to predict root poses given a single image. PoseNet uses a DensePose-CNN to extract pixel features and decodes the pixel features into root pose predictions with a ResNet-18. We visualize the initial root poses on the right. Cyan color represents earlier time stamps and magenta color represent later timestamps.

Preliminary DensePose CSE [27, 28] trains pixel embeddings ψ_I and surface feature embeddings ψ for humans and quadruped animals using 2D keypoing annotations. It represents surface embeddings by a canonical surface with N vertices and vertex features $\psi \in \mathbb{R}^{N\times 16}$. A SMPL mesh is used for humans, and a sheep mesh is used for quadraped animals. The embeddings are trained such that given an pixel feature, a 3D point on the canonical surface can be uniquely located via feature matching.

Naive PnP solution Given 2D-3D correspondences provided by CSE, one way to solve for $\mathbf{G}^t$ is to use perspective-n-points (PnP) algorithm assuming objects are rigid. However, the PnP solution suffers from catastrophic failures due to the non-rigidity of the object, which motivates our PoseNet solution. By training a feed-forward network with data augmentations, our PoseNet solution produces fewer gross errors than the naive PnP solution.

Synthetic dataset genetarion. We train separate PoseNet, one for human, and one for quadruped animals. The training pipeline is shown in Fig. 12. Specifically, we render surface features as feature images $\psi_{\text{rnd}} \in \mathbb{R}^{112\times 112\times 16}$ given viewpoints $\mathbf{G}^* = (\mathbf{R}^*, \mathbf{T}^*)$ randomly generated on a unit sphere facing the origin. We apply occlusion augmentations [39] that randomly mask out a rectangular region in the rendered feature image and replace with mean values of the corresponding feature channels. The random occlusion augmentation forces the network to be robust to outlier inputs, and empirically helps network to make robust predictions in presence of occlusions and in case of out-of-distribution appearance.

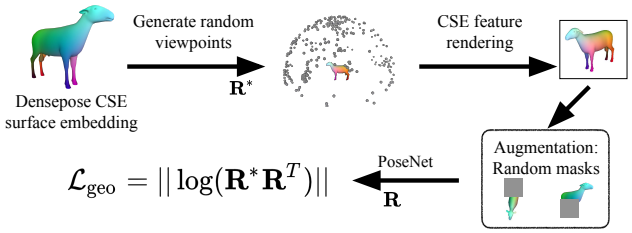

Figure 12. **Training pipeline of PoseNet.** To train PoseNet, we use DesePose CSE surface embeddings, which is pertained on 2D annotations of human and quadruped animals. We first generate random viewpoints on a sphere that faces the origin. Then we render surface embeddings as 16-channel images. We further augment the feature images with random adversarial masks to improve the robustness to occlusions. Finally, the rotations predicted by PoseNet are compared against the ground-truth rotations with geodesic distance.

Loss and inference. We use the geodesic distance between the ground-truth and predicted rotations as a loss to update PoseNet,

$$\mathcal{L}_{\text{geo}} = ||\log(\mathbf{R}^*\mathbf{R}^T)||, \quad \mathbf{R} = \text{PoseNet}(\psi_{\text{rnd}}), \tag{18}$$

where we find learning to predict rotation is sufficient for initializing the root body pose. In practice, we set the initial object-to-camera translation to be a constant $\mathbf{T} = (0, 0, 3)^T$. We run pose CNN on each test video frame to obtain the initial root poses $\mathbf{G}_0^t = (\mathbf{R}, \mathbf{T})$, and compute a delta root pose with the root pose MLP:

$$\mathbf{G}^t = \mathbf{MLP}_{\mathbf{G}}(\omega_r^t)\mathbf{G}_0^t. \tag{19}$$

B.2. Active Sampling Over Pixels

As discussed in Sec. 3.4, we optimize a compact 5-layer MLP function to represent the uncertainty over the image coordinate and frame index: $\hat{\mathbf{U}}(\mathbf{x}^t) = \mathbf{MLP}_{\mathbf{U}}(\mathbf{x}^t)$. The uncertainty MLP is optimized by comparing against the color reconstruction errors in the current forward step:

$$\mathcal{L}_{\mathbf{U}} = \sum_{\mathbf{x},t} \left\| \mathcal{L}_{\text{rgb}}(\mathbf{x}^t) - \hat{\mathbf{U}}(\mathbf{x}^t) \right\|. \tag{20}$$

Note that the gradient from $\mathcal{L}_{\mathbf{U}}$ to $\mathcal{L}_{\text{rgb}}(\mathbf{x}^t)$ is stopped such that $\mathcal{L}_{\mathbf{U}}$ does not generate gradients to parameters besides $\mathbf{MLP}_{\mathbf{U}}$. After a half of the optimization steps, we select $N^a = 2048$ additional *active* samples from pixels with high uncertainties. To do so, we randomly sample $N^p = 16384$ pixels, and evaluate their uncertainties by passing their coordinates and frame indices to $\mathbf{MLP}_{\mathbf{U}}$. The pixels with high uncertainties are selected as active samples and combined with the random samples.

Table 2. Table of hyper-parameters.

Name	Value	Description
B	64	Number of bones
N	128	Sampled points per ray
N^p	16384	Sampled rays per batch
N^a	2048	Active ray samples per batch
N^b	32	Image pairs per batch
(H, W)	(512,512)	Resolution of observed images

B.3. Optimization details

Canonical 3D grid. As mentioned in Sec 3.3, we define a canonical 3D grid $\mathbf{V}^* \in \mathbb{R}^{20\times20\times20}$ to compute the matching costs between pixels and canonical space locations. The canonical grid is centered at the origin and axis-aligned with bounds $[x_{\text{min}}, x_{\text{max}}]$, $[y_{\text{min}}, y_{\text{max}}]$, and $[z_{\text{min}}, z_{\text{max}}]$. The bounds are initialized as loose bounds and are refined during optimization. For every 200 iterations, we update the bounds of the canonical volume as an approximate bound of the object surface. To do so, we run marching cubes on a 64^3 grid to extract a surface mesh and then set L as the axis-aligned (x, y, z) bounds of the extracted surface.

Near-far planes. To generate samples for volume rendering, we dynamically compute the depth of near-far planes (d_n^t, d_f^t) of frame t at each iterations of the optimization. To do so, we compute the projected depth of the canonical surface points $d_i^t = (\Pi^t \mathbf{G}^t \mathbf{X}_i^*)_2$. The near plane is set as $d_n^t = \min(d_i) - \epsilon_L$ and the far plane is set as $d_f^t = \max(d_i) + \epsilon_L$, where $\epsilon_L = 0.2\big(\max(d_i) - \min(d_i)\big)$. To avoid the compute overhead, we approximate the surface with an axis-aligned bounding box with 8 points.

Hyper-parameters. We refer readers to a complete list of hyper-parameters in Tab. 2.

C. Additional results

C.1. Quantitative evaluation

In Sec. 4.3, we presented qualitative results of diagnostics experiments. In Tab. 3, we show quantitative evaluations corresponding to each experiment. As a result, removing canonical features, root pose initialization or flow loss leads to much higher 3D Chamfer distances.

Table 3. **Diagnostics.** Averaged 3D Chamfer distances in cm (↓).

Method	`swing`	`samba`	mean
Reference	**7.05**	**6.59**	**6.82**
w/o canonical feature, Sec 3.3	53.60	56.69	55.15
w/o root pose init., Sec 3.4	55.07	56.76	55.92
w/o flow loss, Eq. 13	47.41	48.55	47.85

C.2. Qualitative results

We refer readers to our supplementary webpage for complete qualitative results.

Table 4. Table of notations.

Symbol	Description
	Index
t	Frame index, $t \in \{1, \ldots, T\}$
b	Bone index $b \in \{1, \ldots, B\}$ in neural blend skinning
i	Point index $b \in \{1, \ldots, N\}$ in volume rendering
	Points
$\mathbf{x}$	Pixel coordinate **x** = (x,y)
$\mathbf{X}^t$	3D point locations in the frame t camera coordinate
$\mathbf{X}^*$	3D point locations in the canonical coordinate
$\hat{\mathbf{X}}^*$	Matched canonical 3D point locations via canonical embedding
	Property of 3D points
$\mathbf{c} \in \mathbb{R}^3$	Color of a 3D point
$\sigma \in \mathbb{R}$	Density of a 3D point
$\boldsymbol{\psi} \in \mathbb{R}^{16}$	Canonical embedding of a 3D point
$\mathbf{W} \in \mathbb{R}^B$	Skinning weights of assigning a 3D point to B bones
	Functions on 3D points
$\mathcal{W}^{t,\leftarrow}(\mathbf{X}^t)$	Backward warping function from $\mathbf{X}^t$ to $\mathbf{X}^*$
$\mathcal{W}^{t,\rightarrow}(\mathbf{X}^*)$	Forward warping function from $\mathbf{X}^*$ to $\mathbf{X}^t$
$\mathcal{S}(\mathbf{X}, \omega_b)$	Skinning function that computes skinning weights of $\mathbf{X}$ under body pose ω_b
	Rendered and Observed Images
$\mathbf{c}/\hat{\mathbf{c}}$	Rendered/observed RGB image
$\mathbf{o}/\hat{\mathbf{s}}$	Rendered/observed object silhouette image
$\mathcal{F}/\hat{\mathcal{F}}$	Rendered/observed optical flow image

Table 5. Table of learnable parameters.

Symbol	Description
	Canonical Model Parameters
$\mathbf{MLP_c}$	Color MLP
$\mathbf{MLP}_{\text{SDF}}$	Shape MLP
$\mathbf{MLP}_{\boldsymbol{\psi}}$	Canonical embedding MLP
	Deformation Model Parameters
$\mathbf{C}^* \in \mathbb{R}^3$	Center of the bones in the rest pose.
$\mathbf{MLP}_{\Delta}$	Delta skinning weight MLP
$\mathbf{MLP_G}$	Root pose MLP
$\mathbf{MLP_J}$	Body pose MLP
	Learnable Codes
$\boldsymbol{\omega}_b^* \in \mathbb{R}^{128}$	Body pose code for the rest pose
$\boldsymbol{\omega}_b^t \in \mathbb{R}^{128}$	Body pose code for frame t
$\boldsymbol{\omega}_r^t \in \mathbb{R}^{128}$	Root pose code for frame t
$\boldsymbol{\omega}_e^t \in \mathbb{R}^{64}$	Environment lighting code for frame t, shared across frames of the same video
	Other Learnable Parameters
$\boldsymbol{\psi}_I$	CNN pixel embedding initialized from DensePose CSE
α_w	Temperature scalar for skinning weights
α_s	Temperature scalar for canonical feature matching
β	Scale parameter that controls the solidness of the object surface
$\Pi \in \mathbb{R}^{3\times3}$	Intrinsic matrix of the pinhole camera model

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 2011. 1

[2] Marc Badger, Yufu Wang, Adarsh Modh, Ammon Perkes, Nikos Kolotouros, Bernd Pfrommer, Marc Schmidt, and Kostas Daniilidis. 3D bird reconstruction: a dataset, model, and shape recovery from a single view. In *ECCV*, 2020. 2

[3] Benjamin Biggs, Ollie Boyne, James Charles, Andrew Fitzgibbon, and Roberto Cipolla. Who left the dogs out: 3D animal reconstruction with expectation maximization in the loop. In *ECCV*, 2020. 2

[4] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. In *CVPR*, 2000. 2

[5] Xu Chen, Yufeng Zheng, Michael J Black, Otmar Hilliges, and Andreas Geiger. Snarf: Differentiable forward skinning for animating non-rigid neural implicit shapes. 2021. 2, 4

[6] Shubham Goel, Angjoo Kanazawa, and Jitendra Malik. Shape and viewpoints without keypoints. In *ECCV*, 2020. 2

[7] Paulo FU Gotardo and Aleix M Martinez. Non-rigid structure from motion with complementary rank-3 spaces. In *CVPR*, 2011. 2

[8] Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-calibrating neural radiance fields. In *ICCV*, 2021. 2

[9] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 2

[10] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *ICCV*, 2017. 5

[11] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *CVPR*, 2020. 6

[12] Muhammed Kocabas, Nikos Athanasiou, and Michael J. Black. Vibe: Video inference for human body pose and shape estimation. In *CVPR*, June 2020. 2

[13] Filippos Kokkinos and Iasonas Kokkinos. To the point: Correspondence-driven monocular 3d category reconstruction. In *NeurIPS*, 2021. 2

[14] Chen Kong and Simon Lucey. Deep non-rigid structure from motion. In *ICCV*, 2019. 2

[15] Nilesh Kulkarni, Abhinav Gupta, David F Fouhey, and Shubham Tulsiani. Articulation-aware canonical surface mapping. In *CVPR*, pages 452–461, 2020. 2, 3, 5

[16] Suryansh Kumar. Non-rigid structure from motion: Prior-free factorization method revisited. In *WACV*, 2020. 2

[17] Xueting Li, Sifei Liu, Shalini De Mello, Kihwan Kim, Xiaolong Wang, Ming-Hsuan Yang, and Jan Kautz. Online adaptation for consistent mesh reconstruction in the wild. In *NeurIPS*, 2020. 2

[18] Xueting Li, Sifei Liu, Kihwan Kim, Shalini De Mello, Varun Jampani, Ming-Hsuan Yang, and Jan Kautz. Self-supervised single-view 3d reconstruction via semantic consistency. *ECCV*, 2020. 2

[19] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *CVPR*, 2021. 2, 3, 5, 6, 8

[20] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *ICCV*, 2021. 2

[21] Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. Neural actor: Neural free-view synthesis of human actors with pose control. *SIGGRAPH Asia*, 2021. 2

[22] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *SIGGRAPH Asia*, 2015. 2

[23] Diogo C Luvizon, Hedi Tabia, and David Picard. Human pose regression by combining indirect part detection and contextual information. *Computers & Graphics*, 2019. 5

[24] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *CVPR*, 2021. 2, 3

[25] Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. Gnerf: Gan-based neural radiance field without posed camera. In *ICCV*, 2021. 2

[26] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2, 3, 4, 5, 6

[27] Natalia Neverova, David Novotny, Vasil Khalidov, Marc Szafraniec, Patrick Labatut, and Andrea Vedaldi. Continuous surface embeddings. In *NeurIPS*, 2020. 3, 5, 6, 10

[28] Natalia Neverova, Artsiom Sanakoyeu, Patrick Labatut, David Novotny, and Andrea Vedaldi. Discovering relationships between object categories via universal canonical maps. In *CVPR*, 2021. 5, 9, 10

[29] Atsuhiro Noguchi, Xiao Sun, Stephen Lin, and Tatsuya Harada. Neural articulated radiance field. In *ICCV*, 2021. 2

[30] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *ICCV*, 2021. 2, 3, 4, 6, 7, 8

[31] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv*, 2021. 2

[32] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *CVPR*, 2019. 2

[33] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *ICCV*, 2021. 2

[34] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *CVPR*, 2021. 2

[35] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *CVPR*, 2020. 2, 8

[36] Shunsuke Saito, Jinlong Yang, Qianli Ma, and Michael J. Black. SCANimate: Weakly supervised learning of skinned clothed avatar networks. In *CVPR*, 2021. 4

[37] Peter Sand and Seth Teller. Particle video: Long-range motion estimation using point trajectories. In *IJCV*, 2008. 2

[38] Vikramjit Sidhu, Edgar Tretschk, Vladislav Golyanik, Antonio Agudo, and Christian Theobalt. Neural dense non-rigid structure from motion with latent space constraints. In *ECCV*, 2020. 2

[39] Krishna Kumar Singh and Yong Jae Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *ICCV*, 2017. 10

[40] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *SIGGRAPH*. 2006. 1

[41] Noah Snavely, Steven M Seitz, and Richard Szeliski. Modeling the world from internet photo collections. *IJCV*, 2008. 1

[42] Shih-Yang Su, Frank Yu, Michael Zollhöfer, and Helge Rhodin. A-nerf: Articulated neural radiance fields for learning human shape, appearance, and pose. In *NeurIPS*, 2021. 2

[43] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew Davison. iMAP: Implicit mapping and positioning in real-time. In *ICCV*, 2021. 6

[44] Narayanan Sundaram, Thomas Brox, and Kurt Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *ECCV*, 2010. 2

[45] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *ICCV*, 2021. 2

[46] Shubham Tulsiani, Nilesh Kulkarni, and Abhinav Gupta. Implicit mesh reconstruction from unannotated image collections. In *arXiv*, 2020. 2

[47] Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. Articulated mesh animation from multi-view silhouettes. *TOG*, 2008. 7

[48] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *NeurIPS*, 2021. 3

[49] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf–: Neural radiance fields without known camera parameters. In *arXiv preprint arXiv:2102.07064*, 2021. 2

[50] Shangzhe Wu, Tomas Jakab, Christian Rupprecht, and Andrea Vedaldi. Dove: Learning deformable 3d objects by watching videos. *arXiv preprint arXiv:2107.10844*, 2021. 2

[51] Donglai Xiang, Hanbyul Joo, and Yaser Sheikh. Monocular total capture: Posing face, body, and hands in the wild. In *CVPR*, 2019. 2

[52] Yuanlu Xu, Song-Chun Zhu, and Tony Tung. Denserac: Joint 3d pose and shape estimation by dense render-and-compare. In *ICCV*, 2019. 6

[53] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. In *NeurIPS*, 2019. 6, 7

[54] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Huiwen Chang, Deva Ramanan, William T Freeman, and Ce Liu. LASR: Learning articulated shape reconstruction from a monocular video. In *CVPR*, 2021. 2, 3, 4

[55] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Ce Liu, and Deva Ramanan. Viser: Video-specific surface embeddings for articulated 3d shape reconstruction. In *NeurIPS*, 2021. 2, 5, 6, 7

[56] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *arXiv preprint arXiv:2106.12052*, 2021. 3, 6

[57] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. In *NeurIPS*, 2020. 5, 7

[58] Yufei Ye, Shubham Tulsiani, and Abhinav Gupta. Shelf-supervised mesh prediction in the wild. In *CVPR*, 2021. 2

[59] Jason Y. Zhang, Gengshan Yang, Shubham Tulsiani, and Deva Ramanan. NeRS: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. In *NeurIPS*, 2021. 6

[60] Silvia Zuffi, Angjoo Kanazawa, Tanya Berger-Wolf, and Michael Black. Three-d safari: Learning to estimate zebra pose, shape, and texture from images "in the wild". In *ICCV*, 2019. 2

[61] Silvia Zuffi, Angjoo Kanazawa, and Michael J. Black. Lions and tigers and bears: Capturing non-rigid, 3D, articulated shape from images. In *CVPR*, 2018. 2

[62] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *CVPR*, 2017. 2