Class-agnostic 4D Reconstruction from Monocular Videos

Anonymous Author(s)
Affiliation
Address
email

Abstract

We introduce **FREDO**, a class-agnostic **F**ramework to **RE**construct the shape and motion of non-rigid **D**ynamic **O**bjects from unconstrained monocular videos. Compared to prior works, our problem setting is more realistic yet more challenging for two reasons: 1) due to camera settings and object articulation an object of interest may never be entirely visible; 2) we aim to reconstruct different types of objects with one unified framework. Based on the recently proposed implicit function representation, we develop two novel modules which help to address the aforementioned challenges. First, we infer a canonical space which permits to aggregate visual cues across time so that the 3D prediction is closely aligned with the visual input. Second, we develop a 4D transformation module which captures object articulation in space and time to support temporal propagation and aggregation. We study the efficacy of FREDO in extensive experiments on the recently introduced SAIL-VOS 3D dataset. We find FREDO outperforms state-of-the-art dynamic reconstruction method by 12.3 mean volumetric Intersection-over-Union (IoU). In ablation studies we validate each of its components.

1 Introduction

4D (space-time) scene understanding is an important problem in machine learning and computer vision, and useful for applications across domains from robotics to augmented/virtual reality (AR/VR). However, accurately recovering the geometry and dynamics of non-rigid objects or even an entire scene in the wild remains a challenge for mainly two reasons: (1) partial visibility of objects because of camera settings or articulation; and (2) variability within and across different object categories.

Existing work addresses these two challenges by either assuming that an object is always entirely visible [30], or by assuming that objects are static [36, 37], or by building priors specific to a particular object category like humans [23] or animals [53]. However, any of these assumptions also limits applicability of those models to unconstrained video data recorded in the wild, where these challenges are only met when taking special care during a video capture.

In contrast, we aim to study the more challenging unconstrained 4D reconstruction setting where objects may never be entirely visible because of either a limited camera field of view or because of occlusion. We focus on non-rigid objects that are deformable and have various motion patterns. Moreover, we aim for a class-agnostic reconstruction framework which can capture the articulate shape of various object types.

To achieve this we develop FREDO. As illustrated in Fig. 1, FREDO predicts the 3D shape of different objects (*e.g.*, human and car) and models their dynamics (*e.g.*, hugging and door shutting) by leveraging a short RGBD video clip together with the instance masks of each observed object. This input seems realistic as depth-sensors are increasingly prevalent [11, 39] and as segmentation models [5, 50] are increasingly accurate and readily available, *e.g.*, in smartphones.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

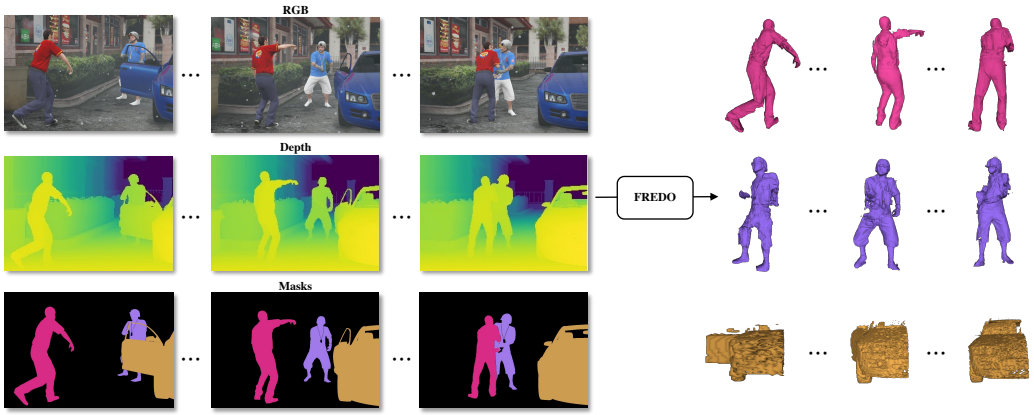

Figure 1: Our framework, FREDO, takes in RGBD video clips and the instance masks of cared objects, and predicts the 3D geometry of various deformable objects at each time step. Despite the objects in the video clip are either occluded (*e.g.*, person) or partially-observed (*e.g.*, car), FREDO renders relatively complete and temporally smooth results. For visualization, we discretize the predicted 3D geometry as mesh.

Specifically, to address the partial visibility challenge introduced by camera settings or by occlusion, FREDO predicts a temporally coherent appearance in a canonical space (§ 3.1). To ensure that the same model is able to reconstruct very different object types in a unified manner we introduce a pixel-aligned 4D implicit representation, which encourages the 3D prediction to closely aligned with the 2D visual inputs (§ 3.1). The visible parts from different frames of the video clip (*e.g.*, the arm of the person with the blue shirt in Fig. 1) are aggregated to reconstruct the same object (§ 3.2). During inference, the reconstructed 3D object in canonical space is propagated to other frames to ensure a temporally coherent prediction (§ 3.3).

We conduct experiments on the recently proposed SAIL-VOS 3D dataset [14], improving over prior work [30, 37] on 4D reconstruction by 12.3 mean IoU. We provide comprehensive evaluations and analysis to validate the effectiveness of each of the introduced components (§ 4).

In summary, our key contributions include: (1) a pixel-aligned neural representation for non-rigid dynamic objects; and (2) an end-to-end trainable framework (FREDO) for learning this representation from monocular video clips showing partially observed objects.

2 Related work

To recover the geometry of non-rigid objects across time, we first discuss possible 3D geometry representations. We then review reconstruction approaches for non-rigid objects. Lastly, we discuss prior works for modelling dynamics of objects, and related datasets.

3D geometry representations. Representations to describe 3D objects can be categorized into two groups: discrete and continuous. Common discrete representations are voxels [12, 25, 33, 38], point-clouds [1, 10] and meshes [13, 18, 28, 46]. Even though being widely used, these representations require to consider some important challenges. Voxels can be easily processed with deep learning frameworks, but are memory inefficient. Point-clouds are more memory efficient to process [34, 35], but do not contain any surface information and thus fail to capture fine details. Meshes are more expressive than the previous two, but their topology and discretization may cause concerns.

To overcome these challenges, continuous representations [7, 32, 26] have been introduced. Specifically, parametric implicit functions are used to describe the 3D geometry of objects and scenes. These methods are not constrained by discretization and can thus model arbitrary geometry at very high resolution [37]. In addition to shape modelling, implicit functions have also been widely used for tasks such as novel view synthesis [27].

Geometry reconstruction of non-rigid objects. Non-rigid objects are considered hard to reconstruct due to big intra-class variations. For popular object categories such as humans or certain animals, object templates [17, 23, 52, 53] have been utilized to ease reconstruction. These templates are

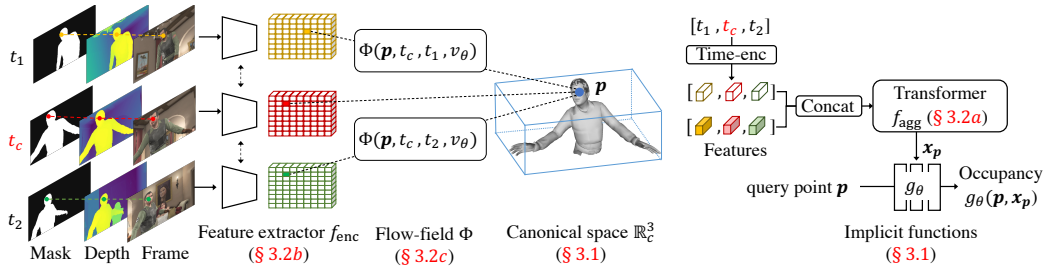

Figure 2: **Framework overview.** FREDO first computes features maps for different video frames using a shared network. For a query point $\mathbf{p}$ (red point) in canonical space $\mathbb{R}^3_c$, FREDO extracts the corresponding 2D features from the feature maps of different frames and aggregate these features using the temporal aggregator f_{agg}. The obtained feature is then used to compute either the occupancy score or velocity vector.

carefully designed parametric 3D models, which restrict the solution space to a small set of model parameters. However, two issues remain: (1) The expressiveness of templates is limited. For instance, the Skinned Multi-Person Linear Model (SMPL) [23] struggles to capture the clothing or hair styles. (2) Systems relying on templates are class-specific as different object categories need to be parameterized differently. It remains illusive to construct a template for every object category.

To overcome these issues, non-template based methods [21, 36, 37] recover 3D geometry from scratch. However, these methods often remain class-specific and focus on static reconstruction. Another line of related work is Non-Rigid Structure From Motion (NR-SFM) [3, 31, 51], which requires either additional shape priors [3, 51] or labeled correspondences [31]. In this paper, we develop a class-agnostic reconstruction framework which doesn't rely on any templates or priors.

Dynamics modelling. Modelling the pixel-level 2D motion of objects from input frames is a popular task known as optical flow estimation [9, 40, 42]. People have also studied dense 3D scene flows to model the motion of each pixel or point cloud in 3D space [16, 24, 43]. These methods compute the flow map for all the pixels or points explicitly, which is time-consuming and less efficient. When using object template representations, a human's 3D motion has also been modeled as the template's parameter change [19, 49]. Recently, a parametric implicit function [22, 30] has been applied to estimate 3D motion, which we adapt and further improve by conditioning on pixel-aligned temporally-aggregated visual features.

4D reconstruction dataset. For 3D static reconstruction, a number of datasets have been introduced over the years, *e.g.*, ModelNet [47], ShapeNet [4], Pix3D [41], and ObjectNet3D [48]. However, very few datasets, *e.g.*, SAIL-VOS 3D [14], Human3.6M [15], 3DPW [45], and D-FAUST [2], support the study of 4D reconstruction (spatial and temporal). We choose SAIL-VOS 3D [14] to train and evaluate our model as it contains dynamic objects of different classes (*e.g.*, human, car, motorcycle) and as it provides monocular RGBD videos. In contrast, Human3.6M [15], 3DPW [45] and D-FAUST [2] only contain meshes for humans.

3 FREDO

As illustrated in Fig. 1, we aim to recover the 3D geometry of a non-rigid object over space and time given a monocular RGBD video together with frame-aligned instance masks and camera matrices. To achieve this we develop FREDO which is illustrated in Fig. 2. Specifically, FREDO reconstructs an object shape and its dynamics in a canonical space, which we detailed in § 3.1. For this FREDO employs a temporal feature aggregator, a pixel-aligned visual feature extractor, and a 3D flow-field, all of which we discuss in § 3.2. These three modules help align articulation of the object across time and resolve occlusions to condense the most useful information into the canonical space. Lastly, we detail how to perform inference using FREDO (§ 3.3) as well as how to train it (§ 3.4).

Notation. We refer to the input video clip via the set $\{I_1, \ldots, I_N\}$. It consists of N RGBD frames $I_i \in [0,1]^{4\times W\times H}$ ($i \in \{1, \ldots, N\}$) of width W and height H, each recorded at time step t_i. For each frame I_i, we also assume the camera matrix and instance mask $m_{ij} \in \{0,1\}^{W\times H}$ are given, where m_{ij} indicates the set of pixels that correspond to object j. For readability, we define the

following operations: 1) `propagate`: transforms temporally in 3D space; 2) `project`: transforms from 3D to 2D (image space); and 3) `lift`: transforms from 2D (image space) to 3D.

3.1 Canonical 4D implicit function

Non-rigid objects deform across time and are often occluded or partially out-of-view. In addition, the depth-size ambiguity also challenges reconstruction algorithms. To condense information about the object across both space and time, we leverage a 3-dimensional canonical space $\mathbb{R}^3_c$, which aims to capture a holistic geometry representation centered around the object of interest. In our case $\mathbb{R}^3_c$ denotes a bounding-box-constrained 3-dimensional space. Temporally, the canonical space corresponds to the center frame I_c with $c = \lceil (1+N)/2 \rceil$. Consequently, its time step is t_c.

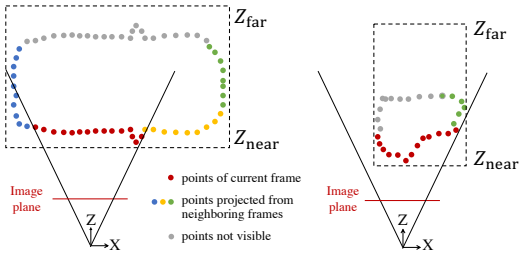

Figure 3: **Canonical Space.** We show examples of canonical space in top-down view. On the left, points from different frames are aggregated to form an enlarged space for capturing the complete shape. On the right, the space is constructed to align with the instance mask.

To construct the canonical space, we infer a 3D bounding box around the non-rigid object j. For all frames I_i, $i \in \{1, \dots, N\}$, we first `lift` the pixels of instance mask m_{ij} of object j to the world coordinate system using the depth map and the camera matrices. These 3D points are then aggregated into canonical space using the camera matrix of the canonical frame. From the aggregated point clouds we infer the horizontal and vertical bounding box coordinates as well as the object's closest distance from the camera, *i.e.*, Z_{near}. However, since the visible pixels only represent the front surface of the observed object, the largest distance of the object from the camera, *i.e.*, Z_{far}, is unknown. To estimate this value, we simply set Z_{far} to stay a fixed distance from Z_{near}. Note that this procedure ignores object articulation. We illustrate two examples of this canonical space in Fig. 3, where red points belong to the canonical frame, blue/yellow/green ones are points aggregated from other frames in the clip, and gray points are the invisible silhouette of the set of observed points. The inferred canonical space is relatively tight and captures the complete shape of the observed object.

To more precisely capture an object inside the bounding-box-constrained canonical space, we use a 4D implicit function. For each 3D query point in canonical space, this function conditions on temporal information to indicate whether that point is inside or outside the object. Specifically, for a query point inside the canonical space, *i.e.*, for $\mathbf{p} \in \mathbb{R}^3_c$, the 4D implicit function is defined as:

$$g_\theta(\mathbf{p}, \mathbf{x}_\mathbf{p}) : \mathbb{R}^3_c \times \mathbb{R}^K \rightarrow [0, 1]. \tag{1}$$

Intuitively, the higher the score $g_\theta(\cdot, \cdot)$ is, the more likely the point $\mathbf{p}$ is part of the object. Here, θ subsumes all trainable parameters in the framework and $\mathbf{x}_\mathbf{p} \in \mathbb{R}^K$ is a K-dimension temporally-aggregated visual feature which summarizes information from all RGBD frames I_i. Concretely,

$$\mathbf{x}_\mathbf{p} = f_{\text{agg}} \left(\{ f_{\text{enc}}(\psi\left(\Phi(\mathbf{p}, t_c, t_i, v_\theta)\right), I_i) \}_{i=1}^N \right), \tag{2}$$

where $\psi(\cdot) : \mathbb{R}^3_c \rightarrow \mathbb{R}^3$ transforms a point in canonical space to 3D world coordinate system. Meanwhile, Φ is the flow-field which `propagates` the point $\mathbf{p}$ from t_c to t_i in canonical space, leveraging a learned 3-dimensional velocity-field v_θ. Note, t_c refers to the time step of the canonical frame I_c while t_i denotes the time step of frame I_i. Given the transformed point $\psi(\Phi(\mathbf{p}, t_c, t_i, v_\theta)) \in \mathbb{R}^3$, we use a pixel-aligned feature encoder f_{enc} to extract a set of visual representations, each of which comes from one of the N frames I_i, *i.e.*, $\forall i \in \{1, \dots, N\}$. The set function f_{agg} is a temporal feature aggregator which merges the temporal information of different time steps. We'll discuss details of temporal aggregator f_{agg}, encoder f_{enc}, flow-field Φ and velocity-field v_θ in § 3.2.

This canonical 4D implicit function `propagates` points across time and extracts the pixel-aligned visual representations from different frames. This differs from: 1) prior works that only consider static objects. Actually, setting $N = 1$ in Eq. (2) will simplify Eq. (1) to $g(\mathbf{p}, f_{\text{enc}}(\psi(\mathbf{p}), I))$, recovering a static Pixel-aligned Implicit Function (PIFu) used in [36, 37]. 2) [30] which encodes the whole video clip $\{I_1, \dots, I_N\}$ into one single feature vector, thus loosing spatial information.

3.2 FREDO model design

In this section, we introduce the temporal aggregator f_{agg}, the feature extractor f_{enc}, and the flow field Φ used in Eq. (2). These modules help FREDO aggregate information from different time steps (frames) and model the dynamics of non-rigid objects.

a) Temporal aggregator f_{agg}. The temporal aggregator f_{agg} is a set function which computes a K-dimensional point representation $\mathbf{x_p} \in \mathbb{R}^K$. The aggregator is developed to deal with partial observations and is implemented as a transformer model as shown in Fig. 2. Assume the flow-field Φ is given and object j is of interest. We first `propagate` a query point within the canonical space, *i.e.*, the point $\mathbf{p} \in \mathbb{R}^3_c$, to every time step t_i, obtaining locations $\psi(\Phi(\mathbf{p}, t_c, t_i, v_\theta)) \in \mathbb{R}^3 \, \forall i \in \{1, \ldots, N\}$. We then `project` each 3D point $\psi(\Phi(\mathbf{p}, t_c, t_i, v_\theta))$ back to the corresponding 2D image frame I_i using the associated camera matrix. For all points that are projected into the mask area m_{ij}, we extract a visual representation using the pixel-aligned feature extractor f_{enc}, which we detail below. Note that due to occlusions or limited camera-view, we may not be able to extract features from every frame in the clip. To cope with this, the aggregator f_{agg} is designed as a set function. Specifically, we implement f_{agg} using a transformer [44] which consists of alternating layers of multi-headed self-attention and MLPs. Please also check the implementation details in § 4.1 amd Appendix A.1.

b) Pixel-aligned feature extractor f_{enc}. Pixel-aligned features help FREDO make 3D predictions that are aligned with the visual 2D input. To achieve this, we develop $f_{\text{enc}}(\mathbf{q}, I_i)$ where the first argument is a point $\mathbf{q}$ in 3D world coordinate system and the second argument is a frame I_i. We first `project` the point $\mathbf{q}$ in world coordinate system to the frame I_i. We then use a pre-trained convolutional neural net [29] to extract the 2D feature map of a video frame $I_i, \forall i \in \{1, \ldots, N\}$. For points that fall within the instance mask of a frame, we extract its visual representation using bi-linear interpolation at the `projected` location. We then append a positional encoding [44] of the frame time step t_i to this feature which helps to retain temporal information. We illustrate this process in Fig. 2 and the resulting feature is combined with visual cues from other frames to serve as input to the temporal aggregator f_{agg}.

c) Flow-field Φ. The flow field models how the articulation of the considered object moves 3D points in space and time. For this let $\Phi(\mathbf{p}, t_1, t_2, v_\theta) \in \mathbb{R}^3_c$ denote a flow-field function in canonical space. It computes the position at time step t_2 of a query 3D point, whose location is $\mathbf{p} \in \mathbb{R}^3_c$ at time step t_1. To compute the displacement, we define a velocity field $v_\theta(\cdot)$ which represents the 3D velocity vectors in space and time via

$$v_\theta(\mathbf{p}, \mathbf{z_p}, t) : \mathbb{R}^3_c \times \mathbb{R}^K \times \mathbb{R} \rightarrow \mathbb{R}^3. \tag{3}$$

Here, $\mathbf{p} \in \mathbb{R}^3_c$ is a point in canonical space with corresponding visual feature $\mathbf{z_p} \in \mathbb{R}^K$ computed as

$$\mathbf{z_p} = f_{\text{agg}}\left(\{f_{\text{enc}}(\psi(\mathbf{p}), I_i)\}_{i=1}^N\right). \tag{4}$$

$f_{\text{enc}}(\psi(\mathbf{p}), I_i)$ is the feature of world coordinate point $\psi(\mathbf{p})$ extracted from frame I_i. Note, $\mathbf{z_p}$ differs from $\mathbf{x_p}$ defined in Eq. (2). The feature $\mathbf{z_p}$ summarizes information of a *static* location from all frames I_i. This feature is beneficial as it helps to capture whether a point remains static or whether it moves. The velocity network then leverages this feature $\mathbf{z_p}$ to predict the object's self-motion.

Using the velocity field, we compute the target location at time step t_2 of a point originating at location $\mathbf{p}$ at time step t_1 by integrating the velocity field over the interval $[t_1, t_2]$ via

$$\Phi(\mathbf{p}, t_1, t_2, v_\theta) = \mathbf{p} + \int_{t_1}^{t_2} v_\theta(\Phi(\mathbf{p}, t_1, t, v_\theta), \mathbf{z_p}, t) dt. \tag{5}$$

Note that $\Phi(\mathbf{p}, t_1, t_2, v_\theta)$ can represent both forward ($t_2 > t_1$) or backward motion ($t_2 < t_1$) given the initial location and velocity-field. To solve the flow-field for discrete video time steps, we approximate the above continuous integral equation using a neural-ODE solver [6], specifically a Runge-Kutta solver. Note, since we integrate over the velocity field, the range of $v_\theta(\mathbf{p}, \mathbf{z_p}, t)$ does not need to be constrained to the canonical space.

3.3 Inference

To reconstruct objects in a relatively long video sequence during inference time, we apply a sliding window-based strategy. Given a temporal window of size N and stride s for input clip $\{I_1, \ldots, I_N\}$ whose corresponding time steps are $\{t_1, \ldots, t_N\}$. The inference procedure is summarized as:

Step 1: We construct the canonical space at the center frame I_c and sample a query point set $\mathcal{P} \subseteq \mathbb{R}_c^3$ uniformly from the inferred space. This step requires no network inference and is thus efficient.

Step 2: We extract static features $\mathbf{z_p}$ for all $\mathbf{p} \in \mathcal{P}$ using Eq. (4). Next, we compute the trajectory of point $\mathbf{p}$, *i.e.*, $\Phi(\mathbf{p}, t_c, t_i, v_\theta)$ for all time steps t_i associated with frames I_i, $\forall i \in \{1, \ldots, N\}$, using a neural ODE solver which solves Eq. (5).

Step 3: We then compute dynamic features $\mathbf{x_p}$ for all $\mathbf{p} \in \mathcal{P}$ using Eq. (2).

Step 4: Using Eq. (1) we finally compute the occupancy scores for all $\mathbf{p} \in \mathcal{P}$, which are then transformed into a triangle mesh via Multi-resolution Iso-Surface Extraction (MISE) [26]. Note, the mesh is constructed in canonical space.

Step 5: To obtain the mesh associated with frame I_i, following Step 2 using flow-field Φ, we `propagate` all vertices of the extracted mesh from the time step t_c of the canonical space to time step t_i which corresponds to non-canonical frame I_i. After using function $\psi(\cdot)$ defined in Eq. (2), we obtain the mesh in 3D world coordinate system. While various discrete representations could be obtained from our implicit representation, we use meshes for evaluation and visualization purposes.

Note, when $s = 1$, we treat each single frame as a canonical frame and reconstruct the object densely for each time step using information from a temporal window of size N. This usually leads to the best result but it is also less efficient. When $s > 1$, we reconstruct the shape in the canonical space and `propagate` it to time steps of neighboring frames using the learned flow-field (Step 5).

3.4 Training

Even though FREDO contains multiple parametric components, *i.e.*, the temporal aggregator f_{agg}, the feature extractor f_{enc}, the velocity-field network v_θ, and the reconstruction network g_θ, it is fully differentiable as we utilize a neural ODE solver [6] to integrate the predictions of the velocity-field network v_θ. For simplicity, we use θ to subsume all trainable parameters of these modules. To better extract shape and motion from given video clips, we jointly optimize these modules end-to-end by addressing the following program:

$$\min_\theta \mathcal{L}_{\text{shape}}(\mathcal{D}, \theta) + \mathcal{L}_{\text{temp}}(\mathcal{D}, \theta), \tag{6}$$

where $\mathcal{D}$ is the training set. $\mathcal{L}_{\text{shape}}(\mathcal{D}, \theta)$ is the shape reconstruction loss in canonical space which encourages FREDO to recover the accurate 3D geometry. $\mathcal{L}_{\text{temp}}(\mathcal{D}, \theta)$ is a temporal coherence loss defined on temporal point correspondences. This loss encourages the flow-field Φ to capture the precise dynamics of different objects. We detail the training set, the sampling procedure, and both loss functions next.

Training set. We train FREDO on a dataset $\mathcal{D}$ that consists of entries from different videos and of various object instances. Formally, $\mathcal{D} = \left\{\left(\{(I_i, t_i)\}_{i=1}^N, \mathcal{V}, \mathcal{Y}\right)\right\}$. Specifically, a data entry for an N-frame video clip includes three components: 1) a set of RGBD frames $\{I_i\}$, associated time steps $\{t_i\}$, as well as corresponding camera matrices $\forall i \in \{1, \ldots, N\}$; 2) an instance ground-truth mesh $\mathcal{V}$ that provides-temporally aligned supervision. Such supervision is available in the recently-released SAIL-VOS 3D dataset [14]. For every vertex $\mathbf{v} \in \mathcal{V}$, we denote its positions at time step t_i via $\mathbf{v}_i$; 3) the ground-truth occupancy $\mathcal{Y}$ in the canonical space. Concretely, for a point $\mathbf{p} \in \mathbb{R}_c^3$ in canonical space, occupancy label $y(\mathbf{p}) \in \{0, 1\}$ indicates whether $\mathbf{p}$ is inside the object ($y = 1$) or outside the object ($y = 0$). As mentioned in § 3.1, the canonical space is chosen so that it corresponds to time step t_c, where $c = \lceil (1 + N)/2 \rceil$.

Sampling procedure. To optimize for the parameters θ during training we randomly sample a set of points $\mathbf{p} \in \mathcal{P}(\mathcal{V})$ within the canonical space. $\mathcal{P}(\mathcal{V})$ contains a mixture of uniform sampling and importance sampling around the ground-truth mesh's surface $\mathcal{V}$ at time step t_c. We apply Gaussian perturbation on sampled surface points, which we empirically found to lead to better results. Similar strategies are also used in prior works [36, 37].

Shape reconstruction loss. To encourage that the canonical 4D implicit function g_θ accurately captures the shape of objects we use the following shape reconstruction loss:

$$\mathcal{L}_{\text{shape}}(\mathcal{D}, \theta) = \sum_{\left(\{(I_i, t_i)\}_{i=1}^N, \mathcal{V}, \mathcal{Y}\right) \in \mathcal{D}} \frac{1}{|\mathcal{D}||\mathcal{P}(\mathcal{V})|} \sum_{\mathbf{p} \in \mathcal{P}(\mathcal{V})} \text{BCE}\left(g_\theta(\mathbf{p}, \mathbf{x_p}), y(\mathbf{p})\right), \tag{7}$$

where $\text{BCE}(\cdot, \cdot)$ represents the standard binary cross-entropy loss.

Temporal coherence loss. FREDO models dynamics of objects explicitly through the flow-field Φ, which leverages the velocity-field network v_θ. As the ground-truth correspondences across time are available in $\mathcal{V}$, we define the temporal correspondence loss via the squared error

$$\mathcal{L}_{\text{temp}}(\mathcal{D}, \theta) = \sum_{\left(\{(I_i, t_i)\}_{i=1}^N, \mathcal{V}, \mathcal{Y}\right) \in \mathcal{D}} \frac{1}{|\mathcal{D}| \cdot N \cdot |\mathcal{V}|} \sum_{\mathbf{v} \in \mathcal{V}} \sum_{i=1}^{N} \|\Phi(\mathbf{v}_c, t_c, t_i, v_\theta) - \mathbf{v}_i\|_2^2. \tag{8}$$

4 Experiments

In this section, we first introduce the key implementation details (§ 4.1) and the experimental setup (§ 4.2). We then present the quantitative results (§ 4.3), ablation studies and analysis (§ 4.4), and qualitative results (§ 4.5).

4.1 Implementation details

Reconstruction network: Following [36], the reconstruction network g_θ is implemented as a 6-layer MLP of dimensions (259, 1024, 512, 256, 128, 1) with skip-connections. The first layer's dimension of 259 is due to the concatenation of visual features (256-dimensional) and query point locations (3-dimensional).

Temporal aggregator: f_{agg} uses a transformer model with 3 multi-headed self-attention blocks and a 2-layer MLP. Layer normalization and skip-connections are applied for each block, and we set the hidden dimension to be 256. To compute the time-encoding, we set the time step at the canonical frame to be 0 and use positional-encoding [44] with 6 exponentially increasing frequencies.

Feature extractor: f_{enc} is implemented as a 4-stack hourglass network [29] following PIFu [36]. Given the instance mask, we take out the object of interest in the picture and resize it to 512×512 before providing it as input to f_{enc}. The output feature map has dimensions of $128 \times 128 \times 256$ of spatial resolution 128×128 and feature dimension $K = 256$.

Velocity-field network: v_θ uses a 5-layer MLP with skip-connections following [30], where the internal dimension is fixed to 512. It takes query points as input and adds the visual features to the activations after the 1st block. For ODE solvers, we use the Dormand–Prince method (dopri5) [8].

In each training iteration, we sample 1024 query points for shape reconstruction and 512 vertices for learning of temporal coherence. We set the video clip length $N = 11$ and the canonical frame is the 6th one, *i.e.*, $c = 6$. We train FREDO end-to-end using the Adam optimizer [20] for 60 epochs with a batch size of 16. The learning rate is initialized to 0.0001 and decayed by $10\times$ at 40th and 55th epochs. Please see Appendix A.1 for more details.

4.2 Experimental setup

Dataset. We use the recently proposed SAIL-VOS 3D dataset [14], which is a photo-realistic synthetic dataset extracted from the game GTA-V. It consists of RGBD video frames at a resolution of 1280×800 together with the ground-truth instance masks and camera matrices. The original dataset contains 178 categories many of which are rigid classes. Here we focus on the 5 most common non-rigid objects: person, car, truck, motorcycle, and airplane (more details in Appendix A.2).

Metrics. We report the volumetric mean Intersection over Union (mIoU) (higher is better) and the Chamfer Distance (Chamfer) (lower is better) to evaluate the shape reconstruction. To evaluate the motion prediction, we compute the ℓ_2 distance between corresponding vertices (ℓ_2-corr) (lower is better). Please see [26, 30, 37] for more on these metrics.

Baselines. We consider several baselines: 1) **Static 3D reconstruction**: we adopt state-of-the-art implicit-function-based methods ONet [26] and PIFuHD [37], which are trained for per-frame static reconstruction. 2) **Static 4D reconstruction**: the static 3D reconstruction baselines can be extended to use 4D data by adding time to the implicit function input. We refer to these baselines as ONet-4D and PIFuHD-4D. 3) **Dynamic 4D reconstruction**: we aim to explicitly model both the shape and motion simultaneously. Oflow [30] falls into this category but it doesn't handle the partial observation

Methods	IoU ↑						Chamfer ↓						ℓ_2-corr ↓					
Classes	person	car	truck	motor	airplane	mean	person	car	truck	motor	airplane	mean	person	car	truck	motor	airplane	mean
Static 3D reconstruction																		
ONet [26]	38.1	15.3	7.9	17.1	5.8	16.8	2.16	2.59	3.74	3.39	4.37	3.25	-	-	-	-	-	-
PIFuHD [37]	41.2	18.9	10.1	16.8	7.0	18.8	1.63	2.03	2.84	2.76	3.68	2.59	-	-	-	-	-	-
Static 4D reconstruction																		
ONet-4D	37.6	16.5	6.7	18.9	8.4	17.6	2.37	2.20	3.92	3.05	3.99	3.11	-	-	-	-	-	-
PIFuHD-4D	43.4	19.7	9.2	17.6	7.2	19.4	1.49	1.84	2.43	2.61	3.45	2.36	-	-	-	-	-	-
Dynamic 4D reconstruction																		
Oflow [30]	38.3	17.1	8.9	23.4	13.2	20.2	1.81	1.97	2.74	1.63	2.39	2.11	1.68	1.56	1.43	1.59	2.42	1.74
FREDO	**50.5**	**35.2**	**23.5**	**32.9**	**20.4**	**32.5**	**0.87**	**1.19**	**1.56**	**1.04**	**1.72**	**1.27**	**1.19**	**1.10**	**0.97**	**0.92**	**1.76**	**1.19**

Table 1: **Quantitative results on SAIL-VOS 3D.** FREDO outperforms both static reconstruction and existing dynamic reconstruction methods by a great margin. For both shape reconstruction (IoU and Chamfer) and dynamics modeling (ℓ_2-corr), FREDO demonstrates significant improvements.

or occlusion explicitly. Note that PHD [49] and HMMR [19] don't apply in our problem setting as we pursue a class-agnostic framework, and 4 out of 5 categories of interest don't have templates.

4.3 Quantitative results

We present quantitative results in Tab. 1, where we show both the per-class and mean results. For a fair comparison, we evaluate the static methods densely on each frame, and evaluate dynamic methods (Oflow and FREDO) with stride $s = 1$. We observe that: 1) FREDO improves upon all the static methods by a large margin for shape reconstruction, *i.e.*, IoU and Chamfer distance. This is because the static methods cannot capture the visual information from other frames in the video clip and thus fail to reconstruct the complete shape. 2) FREDO also improves upon Oflow for shape reconstruction (+12.3 IoU/-0.84 Chamfer) because of the pixel-aligned neural representation, whereas Oflow encodes the whole image as a single feature vector. 3) FREDO also improves upon Oflow in terms of capturing object dynamics (-0.55 ℓ_2-corr) as our canonical space minimizes error propagation due to the choice of c, whereas in Oflow the canonical space is always set as the model space of the 1st frame in the clip. 4) In terms of per-class results, our model performs best for person and worst for airplane. This is because the former class has the most training examples and the later one is very rare while the 3D shape is complex.

4.4 Analysis

Ablation study. In Tab. 2, we provide an ablation study of different components in FREDO.

1) We first replace the temporal aggregator f_{agg} with an average pooling layer where features of different frames are simply averaged and fed into the shape reconstruction network and velocity field network. The results are shown in the 1st row of Tab. 2 (avg. pooling) where we find that the performance dropped by 2.8 IoU, 0.27 Chamfer distance, and 0.11 ℓ_2-corr.

	IoU ↑	Chamfer ↓	ℓ_2-corr ↓
avg. pooling	29.7	1.54	1.30
w/o alignment	26.4	1.92	1.43
implicit-flow	22.5	2.40	1.76
w/o $\mathcal{L}_{\text{temp}}$	25.1	2.13	1.87
FREDO	**32.5**	**1.27**	**1.19**

Table 2: **Ablation studies.** We validate the effectiveness of each single component of FREDO. The evaluation follows the settings in Tab. 1 and mean results over all classes are reported.

2) We then study the pixel-aligned feature representation $\mathbf{x_p}, \mathbf{z_p}$. We replace these two features with the feature map of the entire input frame following Oflow [30], but we still keep the transformer to aggregate these feature maps. The results of this variant are reported in Tab. 2 (w/o alignment). Compared to complete FREDO, this setting greatly hurts the results as the network can no longer handle partial observations and the 3D predictions don't well align with the visual input.

3) We validate the effectiveness of flow-field Φ through two experiments. First, we implement Φ as an implicit network to *directly* predict the location of point $\mathbf{p}$ at time step t_2. It takes input of a query point $\mathbf{p}$, the time interval t_1, t_2 and the visual feature $\mathbf{z_p}$, which is the same as Eq. (5)'s input. We term this variant implicit-flow and report its results in Tab. 2 (3rd row). Even though simpler, this model falls far short of FREDO, as the network needs to predict the entire trajectory across varying time intervals rather than using the local velocity vector. Second, we remove the temporal coherence loss $\mathcal{L}_{\text{temp}}$ during training. This setting aims to mimic a setup where ground-truth correspondences aren't available. As shown in Tab. 2 (last row), the model still trains. However, the ℓ_2-corr error drops significantly (0.78).

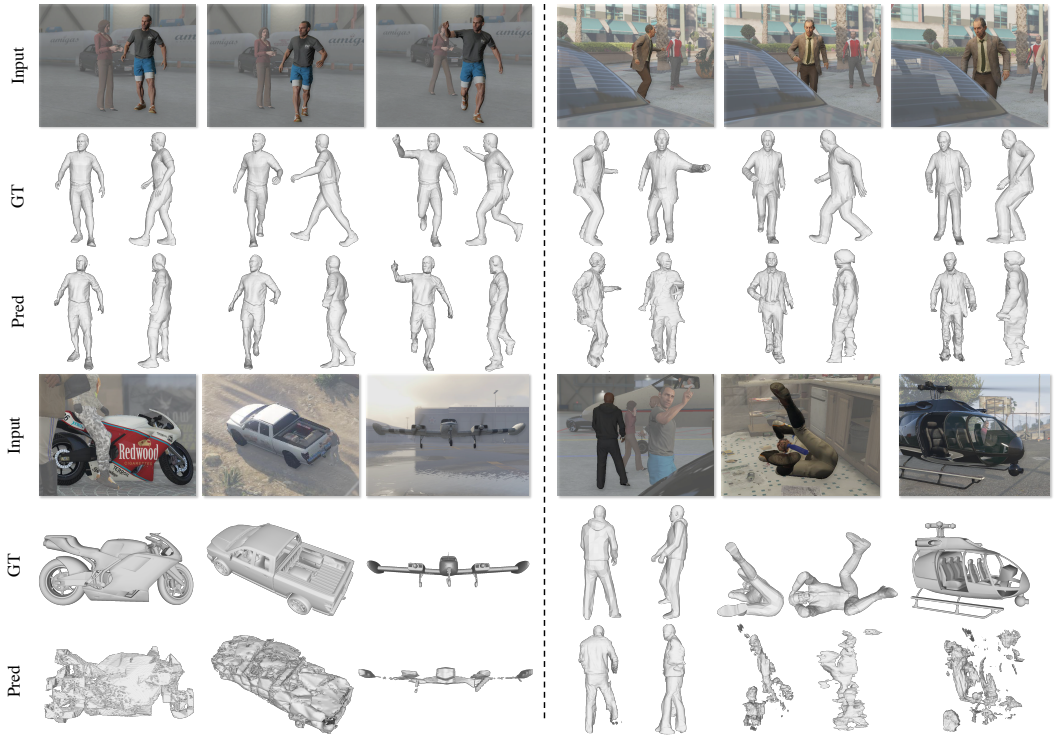

Figure 5: **Qualitative results.** We illustrate the input frames with the object of interest highlighted, the ground-truth mesh, and the reconstructed mesh (front and side views).

Oracle experiment to study flow-field Φ. To better understand the flow-field, we conduct an oracle experiment to test the flow-field given the perfect 3D shape. Concretely, for **Step 5** during inference (§ 3.3), we apply our flow-field function Φ on ground-truth meshes instead of our reconstructed one. Since SAIL-VOS 3D [14] provides every vertex's correspondence across time for ground-truth meshes, we can obtain the ground-truth flow-field. Therefore, the final performance's difference is only due to the accuracy difference between ground-truth and predicted flow-field. We obtain 72.6 (+40.1) IoU and 0.59 (-0.68) Chamfer distance when using the ground-truth flow-field. This indicates that the gap between learned flow-field and ground-truth is still big. We visualize the predictions and the ground-truth motion vectors in Fig. 4.

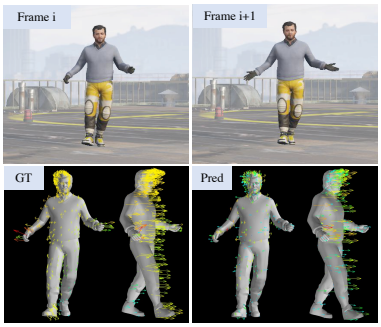

Figure 4: **Flow-field prediction.** FREDO successfully models the forward motion of the entire human body, *e.g.*, head, leg, chest *etc*. However, it fails to capture the more fine-grained motion, *e.g.*, the flow-field on the hand, where the ground-truth clearly indicates the hand will open while the prediction doesn't.

4.5 Qualitative results

Fig. 5 shows a few representative examples of our model's predictions on SAIL-VOS 3D [14]. FREDO is able to recover accurate geometry and dynamics of non-rigid objects from monocular frames. It completes the occluded parts (*e.g.*, legs of human on top right) or invisible parts (*e.g.*, back of the human) by aggregating temporal information. We also show several typical failure cases in Fig. 5 bottom right, where FREDO makes ambiguous predictions for the invisible parts (*e.g.*, man with 3 legs), fails to reconstruct objects in strange poses (*e.g.*, man on the floor), or doesn't understand rare and complex object instances (*e.g.*, helicopter). Please see Appendix B.1 for more results and analysis.

5 Conclusion

In this work, we present FREDO, a novel class-agnostic framework to reconstruct the dynamic objects from unconstrained monocular videos. FREDO is implemented as a canonical 4D implicit function network to capture the precise shape and articulation and deal with partial visibility. We validate the effectiveness of FREDO on the SAIL-VOS 3D dataset.

References

[1] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas. Learning representations and generative models for 3d point clouds. In *ICML*, 2018.

[2] F. Bogo, J. Romero, G. Pons-Moll, and M. J. Black. Dynamic FAUST: Registering human bodies in motion. In *CVPR*, 2017.

[3] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3D shape from image streams. In *CVPR*, 2000.

[4] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv*, 2015.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2017.

[6] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations. In *NeurIPS*, 2018.

[7] Z. Chen and H. Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019.

[8] J. R. Dormand and P. J. Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 1980.

[9] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015.

[10] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, 2017.

[11] C. Franklin. Apple unveils new iPad Pro with breakthrough LiDAR Scanner and brings trackpad support to iPadOS. https://www.apple.com/, 2020.

[12] G. Gkioxari, J. Malik, and J. Johnson. Mesh r-cnn. In *ICCV*, 2019.

[13] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. A papier-mâché approach to learning 3d surface generation. In *CVPR*, 2018.

[14] Y.-T. Hu, J. Wang, R. A. Yeh, and A. G. Schwing. SAIL-VOS 3D: A Synthetic Dataset and Baselines for Object Detection and 3D Mesh Reconstruction from Video Data. In *CVPR*, 2021.

[15] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *TPAMI*, 2013.

[16] M. Jaimez, M. Souiai, J. González, and D. Cremers. A primal-dual framework for real-time dense rgb-d scene flow. *ICRA*, 2015.

[17] H. Joo, T. Simon, and Y. Sheikh. Total Capture: A 3D Deformation Model for Tracking Faces, Hands, and Bodies. In *CVPR*, 2018.

[18] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. Learning Category-Specific Mesh Reconstruction from Image Collections. In *ECCV*, 2018.

[19] A. Kanazawa, J. Y. Zhang, P. Felsen, and J. Malik. Learning 3d human dynamics from video. In *CVPR*, 2019.

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[21] X. Li, S. Liu, K. Kim, S. De Mello, V. Jampani, M.-H. Yang, and J. Kautz. Self-supervised single-view 3D reconstruction via semantic consistency. In *ECCV*, 2020.

[22] Z. Li, S. Niklaus, N. Snavely, and O. Wang. Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. In *CVPR*, 2021.

[23] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. SMPL: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 2015.

[24] W.-C. Ma, S. Wang, R. Hu, Y. Xiong, and R. Urtasun. Deep rigid instance scene flow. In *CVPR*, 2019.

[25] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, 2015.

[26] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019.

[27] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[28] C. Nash, Y. Ganin, S. A. Eslami, and P. Battaglia. Polygen: An autoregressive generative model of 3d meshes. In *ICML*, 2020.

[29] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.

[30] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *ICCV*, 2019.

[31] D. Novotny, N. Ravi, B. Graham, N. Neverova, and A. Vedaldi. C3dpo: Canonical 3d pose networks for non-rigid structure from motion. In *ICCV*, 2019.

[32] J. J. Park, P. R. Florence, J. Straub, R. A. Newcombe, and S. Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. *CVPR*, 2019.

[33] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, 2016.

[34] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017.

[35] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017.

[36] S. Saito, Z. Huang, R. Natsume, S. Morishima, A. Kanazawa, and H. Li. PIFu: Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization. In *ICCV*, 2019.

[37] S. Saito, T. Simon, J. Saragih, and H. Joo. PIFuHD: Multi-Level Pixel-Aligned Implicit Function for High-Resolution 3D Human Digitization. In *CVPR*, 2020.

[38] S. Song and J. Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *CVPR*, 2016.

[39] S. Stein. Lidar on the iPhone 12 Pro. https://www.cnet.com/, 2020.

[40] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018.

[41] X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *CVPR*, 2018.

[42] Z. Teed and J. Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020.

[43] Z. Teed and J. Deng. RAFT-3D: Scene Flow using Rigid-Motion Embeddings. In *CVPR*, 2021.

[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[45] T. von Marcard, R. Henschel, M. J. Black, B. Rosenhahn, and G. Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *ECCV*, 2018.

[46] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018.

[47] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015.

[48] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, and S. Savarese. Objectnet3d: A large scale database for 3d object recognition. In *ECCV*, 2016.

[49] J. Y. Zhang, P. Felsen, A. Kanazawa, and J. Malik. Predicting 3d human dynamics from video. In *ICCV*, 2019.

[50] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.

[51] Y. Zhu, D. Huang, F. De La Torre, and S. Lucey. Complex non-rigid motion 3d reconstruction by union of subspaces. In *CVPR*, 2014.

[52] S. Zuffi, A. Kanazawa, and M. J. Black. Lions and tigers and bears: Capturing non-rigid 3d articulated shape from images. In *CVPR*, 2018.

[53] S. Zuffi, A. Kanazawa, T. Berger-Wolf, and M. J. Black. Three-D Safari: Learning to Estimate Zebra Pose, Shape, and Texture from Images" In the Wild". In *ICCV*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See § 4.4 and § 4.5.
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See Appendix A.1 and attached files.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See § 4.1 and Appendix A.1.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Experiments are conducted on validation set and best numbers are reported. Ours significantly out-performs prior works.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix A.1
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See § 4.2.
   (b) Did you mention the license of the assets? [Yes] See Appendix A.1 and A.2.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See Appendix A.2.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See Appendix A.2.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]