Building 4D Models of Objects and Scenes from Monocular Videos

Gengshan Yang

CMU-RI-TR-23-54

July, 2023

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Deva Ramanan, *Chair*
Shubham Tulsiani
Jessica Hodgins
Yaser Sheikh
Angjoo Kanazawa, *UC Berkeley*

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics.

To my parents Zhixun Yang, Yanqin Wei, my partner Yisha, and my cat Pikachu

Abstract

This thesis studies how to infer the time-varying 3D structures of *generic*, *deformable* objects, and *dynamic* scenes from monocular videos. A solution to this problem is essential for virtual reality and robotics applications. To reconstruct and track dynamic structures in 3D, prior work takes advantage of specialized sensors or parametric body models learned from registered 3D data. However, neither of them scales robustly to diverse sets of objects and events that one may see in the real world.

Inferring 4D structures given 2D observations is challenging due to its under-constrained nature: Given images captured at different time instances, there are an infinite number of interpretations of their underlying geometry, color, and motion. In a casual setup where there is neither sufficient sensor measurement nor rich 3D supervision, one needs to tackle three challenges: (1) Registration: how to find correspondence and track camera frames? (2) Depth ambiguity: how to lift 2D observations to 3D? (3) Limited views: how to infer the structures that are not observable?

We first study the 4D reconstruction problem in a single video setup and then extend it to multiple videos, different instances, and scenes. Inspired by analysis-by-synthesis, we set up an *inverse graphics* problem and solve it with *generic data-driven priors*. Inverse graphics models (e.g., differentiable rendering, differentiable physics simulation) approximate the true generation process of a video with differentiable operations, allowing one to inject prior knowledge about the physical world and compute gradients to update the model parameters. Generic data-driven priors (e.g., optical flow, pixel features, viewpoint) provide guidance to register pixels to a canonical 3D space, which allows us to fuse observations over time and across similar instances. Building upon these observations, we develop methods to capture 4D models of deformable objects and dynamic scenes from in-the-wild video footage.

Acknowledgments

I'm most grateful to my advisor Deva Ramanan for training me to become an independent researcher and for his guidance through my Masters's and PhD. I couldn't think of a better advisor than Deva who supports students in pursuing their interests and transforming research into an enjoyable endeavor. If it were not for the countless hours he spent in our discussions, I would not have delved deep into the core of computer vision research. His passion for knowledge and discovery always stimulates me to push beyond conventional boundaries.

I would like to thank my committee members – Shubham Tulsiani, Jessica Hodgins, Yaser Sheikh, and Angjoo Kanazawa for their time, and constructive feedback on my research and thesis. They always stand as my role models and inspire me to work on challenging problems and develop unique perspectives.

I also would like to express my gratitude to my mentors and collaborators during my internships – Deqing Sun, Varun Jampani, Ce Liu, Daniel Vlasic, Forrester Cole, Huiwen Chang, Bill Freeman, Hanbyul Joo, Minh Vo, Natalia Neverova, Andrea Vedaldi, Joshua Manela, and Michael Happold, who have guided me at various stages of my academic journey. Their assistance in overcoming research hurdles, enhancing my technical skills, and improving my presentation and speech has been invaluable.

Many thanks to faculty members of CMU – Srinivasa Narasimhan, Simon Lucey, Jun-Yan Zhu, Zachary Manchester, and Kris Kitani, who inspired me and helped me in various discussions and projects.

During my six years of study at CMU, I had the opportunity to collaborate and learn from great colleagues – Peiyun, Chaoyang, Aayush, Martin, Rohit, Ravi, Achal, Pavel, Donglai, Yufei, Jason, Dinesh, Ziyan, Xianyi, Shuyan, Wenxuan, Chao, Allan, Kangle, Jeff, Chonghyuk, Alex, Jono, Aljosa, Nate, Neehar, Tarasha, Zhiqiu, Haithem, Gautam, Anish, Siva, Ishan, Shuo, Ziyang, and Swami. They give me professional advice and make my life colorful.

Many thanks to my advisors and mentors in my undergrad – Na Lu, Chao Shen, Jinjun Wang, and Nuno Vasconcelos who provided opportunities to work on exciting projects and stimulated my interest in research.

I am also grateful to those who helped me outside of research, my voice

teacher Betsy Lawrence, friends at CMU Jazz Choir, Hadley Pratt, and Suzanne Muth who answered all my questions about graduate program.

My research has been supported by CMU Argo AI Center for Autonomous Vehicle Research, Qualcomm Innovation Fellowship, and Google Cloud Platform (GCP) awards.

Finally, to my parents, my partner, and my cat for their unconditional support during times of stress and uncertainty. They are the inspiration that keeps me moving forward.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

Chapter 1

Introduction

1.1 Background

Reconstructing the dynamic 3D world from imagery is a long-standing problem in computer vision. It can be used to capture digital avatars, assets, and environments that faithfully represent the real world, and therefore essential for virtual reality applications that require an immersive user experience. It also offers a scalable solution to obtain diverse and natural motion signals from real-world agents that can be used for motion and behavior imitation in robotics [110, 185].

In the past, capturing dynamic events often requires specialized multiview camera systems [1, 109, 181]. An optical motion capture (MoCap) system typically consists of multiple high-resolution cameras whose positions and orientations are carefully calibrated. Multiple cameras can observe and triangulate the positions of markers, which can be mounted on human or animal subjects. Although these marker-based systems provide a robust way to capture motion within labs or studios, they have inherently limited capture volume and are very sensitive to camera configuration changes, making outdoor usage extremely difficult and unreliable. MoCap systems also typically cost tens to hundreds of thousands of dollars. All of these factors limit the diversity of environments and targets that can be studied with a MoCap system.

Alternatively, markerless motion capture has become increasingly popular in recent years [12, 103, 106, 317]. For example, Joo *et al.* [103] built a multiview system to capture human social interaction. Although these systems capture whole-body movement without using markers, they still require an indoor studio with hundreds of

synchronized cameras, which makes it challenging to generalize to in-the-wild targets and behaviors. Some recent works [107, 117, 287] learn data-driven models to predict full body movements from a single monocular camera. However, they heavily rely on pre-built body models (e.g., SMPL [144]) and do not generalize well to in-the-wild clothing, events, and non-human categories.

Meanwhile, there are millions of hours of videos on the internet, which provide diverse and rich samples of the real world. However, such data are rarely used for reconstruction purposes. A natural question is, given casually captured monocular videos, can we build 4D models including geometry, motion, and even *physics*, that faithfully represent the dynamics 3D world?

1.2 Challenges

Inferring 4D structures given 2D observations is challenging due to its under-constrained nature: given RGB pixels captured at different time instances, there are an infinite number of interpretations of the underlying 3D elements whose geometry, color, and motion generate those observations. In a casual setup where one has neither complete sensor input nor rich 3D supervision, one needs to tackle three fundamental challenges: (1) Registration: how to find correspondence of pixels and track the camera frame over time? (2) Scale ambiguity: how to lift 2D observations to 3D accounting for the depth ambiguity? (3) Occlusion: how to infer the structures that are not observable due to self-occlusion or occlusion by the others?

Classic structure-from-motion (SfM) methods [54, 221, 223, 233] take advantage of *multiview* data recordings. Given N camera views, the reconstruction problem can be formulated as jointly solving (N-1) relative camera transformations and a canonical geometry from either 2D correspondences or photometric information. The problem is typically well-conditioned as there are more constraints than unknowns. However, such results are limited to rigid structures. When dealing with time-varying geometry, the number of unknowns easily explodes. Non-rigid structure-from-motion (NRSfM) methods [44, 176, 253] leverage priors such as spatial and temporal smoothness, which is a double edge sword that constrains the problem on one hand, but brings the risk of over-regularizing the solution due to the nature of the over-simplified shape and motion models. Moreover, NRSfM methods often assume accurate 2D point tracking

over the whole sequence, which is a challenging problem on its own.

A recent trend for reconstructing dynamic structures to bake 3D priors into parametric body models or neural network weights, assuming 3D data and/or registrations (e.g., viewpoint, correspondence) are available. These have proven quite successful for human reconstruction [144, 214, 334]. However, although one could learn highly-accurate models from registered 3D scans for certain categories, such as humans and vehicles, it is costly to acquire large-scale 3D data for the others, such as animals, plants, and clothes. Moreover, it is almost impossible to collect diverse enough training data on motion, physical interactions, and social behaviors that represent the real world, considering their richness and complexity.

1.3 Main Contributions

Recent advances in inverse graphics [140, 153, 165] and visual correspondence [31, 37, 76, 171, 178, 242, 249, 301] bring opportunities for solving the above-mentioned challenges in dynamic 3D reconstruction. Inverse graphics (e.g., differentiable rendering, differentiable physics simulation) provides a differentiable mechanism that approximates the true generation process of a video, through which one could not only back-propagate gradients to improve the model, but also inject prior knowledge about the physical world. Generic data-driven priors (e.g., optical flow, pixel features, viewpoint) provide guidance to register pixels to a canonical 3D space, which allows us to fuse observations over time and borrow common structures across similar instances.

Leveraging the above observation, this thesis aims to develop methods to build 4D models from monocular video observations that faithfully represent the physical world. An overview of the pipeline is shown in Fig. 1.1

1.3.1 Building 4D Models of Objects from Videos

We first look at the capture problem at an instance level: how to recover the 3D geometry and motion for a *generic deformable objects* given *monocular RGB videos*?

LASR [306] introduces a template-free method for articulated shape reconstruction given a single video. It leverages *data-driven motion correspondence* and *segmentation* priors in a differentiable mesh rendering setting. By differentiably-rendering object segmentation and motion correspondence, one can compute gradients to adjust

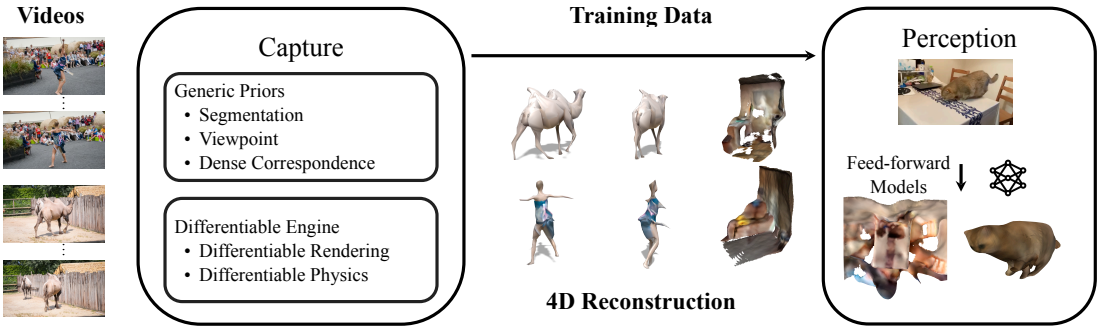

Figure 1.1: Given monocular RGB videos, our goal is to build 4D representations of deformable objects and dynamic scenes. We first capture the 4D model using inverse graphic optimization and then distill them into efficient feed-forward architectures.

the camera, shape and motion parameters by mininizing the difference between the rendered and observed signals produced by off-the-shelf networks [116, 301]. Relying on only low-level cues such as motion and segmentation, LASR successfully reconstructs humans, animals, and objects from short video clips.

Besides correspondence and segmentation, ViSER [307] further incorporates off-the-shelf *appearance features* to reconstruct a long video. We introduce a method to register pixels to a canonical 3D space and integrate their features over time to build a view-invariant feature embedding. Such 3D feature embedding establishes dense long-range correspondences that aid ViSER to track and reconstruct body parts in long videos with viewpoint changes and occlusions.

In addition to a single video, another interesting type of data is a *video collection* of a single instance, which is widely available for one's pets or family members. 3D-ifying those are challenging due to different viewpoints and backgrounds across videos, making viewpoint registration hard. BANMo [308] addresses this challenge by combining visual correspondence with *viewpoint priors*. For instance, one could pre-train a viewpoint estimator for categories of interest (e.g., quadrupeds and humans) using synthetic data or annotated images, that provides a rough initialization of camera pose with respect to the object. Using rough viewpoint significantly simplifies the problem and avoids shape-viewpoint ambiguity. As a result, BANMo builds high-quality animatible 3D models of cats, dogs, penguins, and humans from casual video collections.

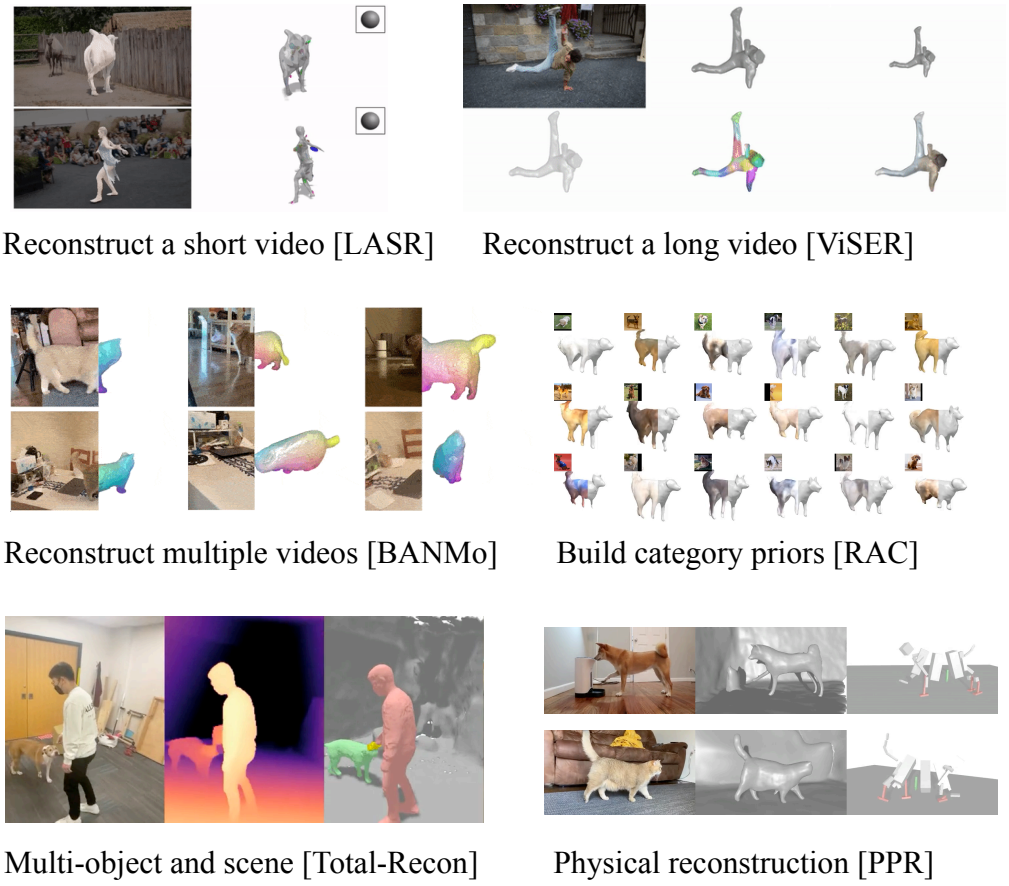

Figure 1.2: **Dynamic object and scene capture.** We capture deformable objects and dynamic scenes from casually-taken videos.

1.3.2 Learning 3D Body Prior from Videos

In the past, building shape and pose priors for categories is challenging due to the need for 3D scans, laborious registration, and manual rigging. Prior methods [306, 308] provides a pathway to obtain high-quality 3D models from monocular videos but are limited to single instances. RAC [309], takes a step further to build category shape and motion priors from many videos of similar instances.

One challenge is how to capture the *morphological variation* of instances within a category. For example, huskys and corgis are both dogs, but have different body shapes, appearance, and skeleton dimensions. Another challenge arises from the impoverished nature of in-the-wild videos: objects are often *partially observable* in

a limited number of viewpoints, and object segmentation can be inaccurate. Those goals appear contradictory with each other. On one hand, the category model should be made as flexible as possible to capture instance-specific details. On the other hand, to deal with partial or impoverished video inputs, one would want to constrain the common structures to be shared across a category. For instance, dogs have two ears.

To address these challenges, RAC exploits the latent structure of a category: an instance code randomization scheme is introduced to regularize the unobserved body parts to be coherent across instances while remaining faithful to the input views. To further deal with impoverished segmentation, a 3D background representation is jointly optimized, which helps disentangle objects from background pixels. As a result, we learn animatable 3D models of cats, dogs, and humans which outperform prior art. Category prior allows one to generate instances and motions from a category and also enables object reconstruction with partial observations.

Optimization with inverse graphics produces models that are faithful to the constraints and input data, but are too costly to be deployed on edge devices and do not meet the latency requirements. In DASR [246], we further show category priors can be distilled into efficient neural architectures, mapping videos to deformable neural fields in real time.

1.3.3 Capturing Physically-plausible Motion from Videos

To build 4D models from videos, one is not only interested in reconstructing the geometry and pose of an instance in the camera space but also tracking its movements in the *world coordinates* [315]. This is challenging if prior knowledge about the object's shape and motion is not given, due to the fundamental depth ambiguity between 3D structures and 2D pixel observations. In TotalRecon [235], we show that a sparse depth sensor helps reduce the scale ambiguity and reconstruct metric-scale object shape and motion in the world space.

When depth sensor is not available, PPR [310] enforces physically-plausible scales through differentiable physics simulation. Just as differentiable renderers have lowered the barrier of entry for (neural) 3D modeling, differentiable simulators are also lowering the barrier for incorporating physical constraints. We find coupling differentiable rendering with differentiable physics-based simulation can jointly solve

for the object geometry, motion, background scene, and physics parameters including body mass distributions and control parameters. Compared to prior approaches that rely on strong human priors to estimate 3D pose and ground contact, PPR works for more unconstrained settings including both humans and animals in an unknown environment, enabled by end-to-end optimization from videos to physics, which potentially opens the door for more comprehensive analysis of interactions and behaviors of agents.

1.3.4 Perceive Dynamic Scenes

In the end, we look into the dynamic scene perception problem that I worked on in my early years of Ph.D., hoping this sheds light on how to learn efficient and generalizable perception models.

Figure 1.3: **Dynamic scene perception.**

As a widely observed lesson in supervised learning, scaling up training data is the key to generalization. However, it is often costly to collect ground-truth data for dynamic scene perception tasks (such as optical flow and scene flow estimation). As a result, existing real-world datasets contain dozens or hundreds of images with ground-truth annotations [65, 161, 219], making generalization challenging. To tackle this challenge, we analyze and design suitable inductive bias in neural architectures.

We first show that the use of coarse-to-fine, volumetric, and separable convolutional architectures leads to high efficiency and generalization ability in the context of depth estimation [304], stereo matching [305] and optical flow [301]. Because one may hallucinate multiple interpretations of depth and correspondence, we ask the models to also report back *multimodal* distributions over 3D structures and possible matches, allowing them to produce uncertainty that can be passed to downstream modules.

Inspired by biological vision, we use time-to-contact [302] to upgrade optical flow to 3D by reasoning about the local patch expansion. Compared to priors works that directly regress the 3D flow vectors, this simple modification leads to significant improvement in scene flow estimation. In RigidMask [303], we analyze and design rigidity cost maps given low-level cues including depth, motion, and optical expansion, which can be integrated as a network layer to segment rigid bodies. We show that enforcing rigidity constraints further improves depth and 3D motion estimation.

1.4 Excluded Research

To keep the draft clean, some of my research works that are also relevant to this thesis are excluded, including data-driven methods for depth estimation, stereo matching, optical flow, dynamic scene reconstruction from an RGBD video, distilling 4D neural fields, and robot motion imitation from videos:

1. Inferring Distributions Over Depth from a Single Image, Yang et al. [304],
2. Hierarchical Deep Stereo Matching on High-resolution Images, Yang et al. [305],
3. Volumetric Correspondence Networks for Optical Flow, Yang and Ramanan [301],
4. Deformable Scene Reconstruction for Embodied View Synthesis, Song et al. [235],
5. Distilling Neural Fields for Real-Time Articulated Shape Reconstruction, Tan et al. [246],
6. A General System for Legged Robot Motion Imitation from Casual Videos, Zhang et al. [324].

Part I

Building 4D Models of Objects

Chapter 2

Articulated Shape Reconstruction from a Monocular Video

2.1 Introduction

Perceiving and modeling the geometry and dynamics of 3D entities is an open research problem in computer vision and has numerous applications. One fundamental challenge is the under-constrained nature of the problem: from limited 2D image measurements, there exist multiple interpretations of the geometry and motion of the 3D world.

A recent and promising trend for addressing this challenge is exploiting data priors, which have proven quite successful for high-level vision tasks, such as image classification and object detection [47, 135]. However, in contrast to high-level vision tasks, it is often costly to obtain 3D annotations for real-world entities. For example, SMPL [144] is learned from thousands of registered 3D scans of human. SMAL [333] is learned from scans of animal toys and a manually rigged mesh model. It involves nontrivial efforts to collect such data for an arbitrary object category. Therefore, existing methods often fail to capture objects of novel or unknown classes, and hallucinate an average 3D structure based on the category shape prior, as shown in Fig. 3.1.

Interestingly, remarkable progress has been made in the field of SLAM and structure-from-motion without relying on strong shape priors by taking advantage of *multiview* data recordings. However, such results are limited to static scenes. We

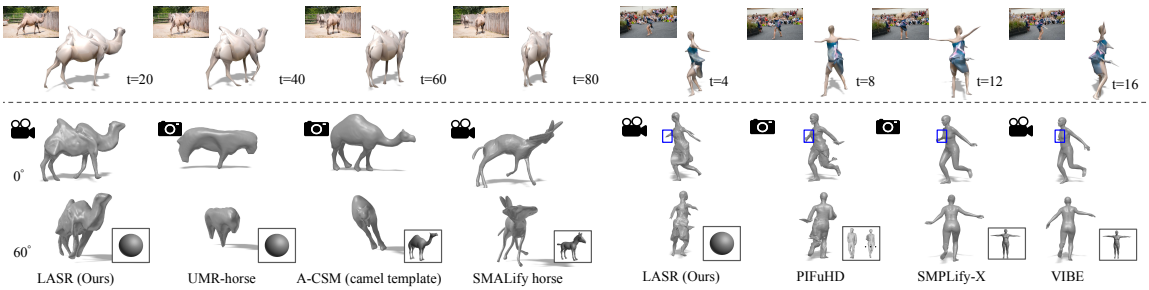

Figure 2.1: **Top**: Sample input video frames and articulated shapes recovered by our method (LASR). **Bottom**: Comparison with existing methods, where the input to each method (either video or image) is denoted at the top left, and the shape template being used is denoted at the bottom right of each result. Many existing approaches on nonrigid shape reconstruction heavily rely on category-specific 3D shape templates, such as SMPL for human [144, 182] and SMAL for quadrupeds [24, 333]. In contrast, LASR jointly recovers the object shape, articulation, and camera parameters from a monocular video without using category-specific shape templates. By relying on generic shape and motion priors, LASR applies to a wider range of nonrigid shapes and yields high-fidelity 3D reconstructions: It recovers both humps of the camel, which are missing from other methods. It also recovers the silk ribbon of the dancer (as denoted by the blue box), which confuses SMPLify-X and VIBE as the right arm. Please refer to video results on our project page.

explore an intermediate regime between these two extremes: *Can one reconstruct an articulated shape from video data without relying on template priors?*

Why videos? To reconstruct 3D object shape from images, prior work learns category-specific shape models either from 3D data [71, 214] or from 2D supervision, such as object silhouette and keypoints in a large image collection [38, 73, 108, 131]. However, 3D data are generally difficult to acquire at a large scale due to sensor design. Although it is easier to collect images of the same category, enforcing multiview constraints is often challenging, due to ambiguities of associating 2D observations across instances and under different viewpoints [43, 209]. Video serves as an alternative to depth scans and image collections – videos are easier to acquire, and provide well-defined multiview constraints on the 3D shape of the same instance.

Why optical flow? To solve the inverse problem, prior work discussed various forms of 2D constraints or supervision, such as object silhouette, texture, 2D keypoints, and semantic parts [10, 73, 108, 131]. Why should motion be treated as a first-class citizen? Besides that optical flow naturally encodes correspondences, it provides more

fine-grained information than keypoints as well as semantic parts. Different from long-range point tracks, which is the classic input for NRSfM [216], optical flow can be obtained more reliably [249, 301] over two consecutive frames.

Why not nonrigid SfM? NRSfM deals with a problem similar to ours: given a set of 2D point trajectories depicting a deformable object in a collection of images, the goal is to recover the 3D object shape and pose (i.e., relative camera position) in each view. Usually, trajectories of 2D points are factorized into low-rank shape and motion matrices [29, 74, 121] without using 3D shape templates. Although NRSfM is able to deal with generic shapes, it requires reliable long-term point tracks or keypoint annotations, which are challenging to acquire densely in practice [216, 230, 243].

Proposed approach: Instead of inferring 3D shapes from category-specific image collections or point trajectories, we build an articulated shape model from a monocular video of an object. Recent progress in differentiable rendering allows one to recast the problem as an analysis-by-synthesis task: *we solve the inverse graphics problem of recovering the 3D object shape (including spacetime deformations) and camera trajectories (including intrinsics) to fit video observations, such as object silhouette, raw pixels, and optical flow.* An overview of the pipeline is shown in Fig. 2.2.

Contributions: We propose a method for articulated shape reconstruction from a monocular video that does not require a prior template or category information. It takes advantage of dense two-frame optical flow to overcome the inherent ambiguity in the nonrigid structure and motion estimation problem. It automatically recovers a nonrigid shape under the constraints of rigid bones under linear-blend skinning. It combines coarse-to-fine re-meshing with soft-symmetric constraints to recover high-quality meshes. Our method demonstrates state-of-the-art reconstruction performance in the BADJA animal video dataset [24], strong performance against model-based methods on humans, and higher accuracy on two animated animals than A-CSM [122] and SMALify [24] that use shape templates.

2.2 Related Work

Below and in Tab. 2.1, we discuss related work of nonrigid shape recovery according to priors being used (shape template, category prior, or generic shape and motion priors).

Table 2.1: Related work in nonrigid shape reconstruction. (1)Model-based optimization and regression methods. (2)Category-specific mesh reconstruction. (3)Template-free approaches. S: single view. V: video or multi-view data. I: images. J2: 2D joints. J3: 3D joints. M: 2D masks. V3: 3D scans. C: camera matrices. F: optical flow. MF: multi-frame optical flow. quad: quadruped animals. †:Only representative categories are listed.

	Method	category	template	test-input	train
(1)	SMPLify [27]	human	SMPL	S:J2,M	None
	VIBE [117]	human	SMPL	V:I	J2,J3
	SMALify [24]	quadx5	SMAL	V:J2,M	None
	SMALR [334]	quadx12	SMAL	S:J2,M	None
	SMALST [335]	zebra	SMAL	S:I	J2,V3
	WLDO [25]	dog	SMAL	S:I	J2,M
(2)	CMR [108]	bird†	SfM-hull	S:I	J2,M,C
	UCMR [73]	bird†	cate-mesh	S:I	M
	UMR [131]	bird†	None	S:I	M
	IMR [259]	animals	cate-mesh	S:I	M
	A-CSM [122]	animals	cate-mesh	S:I	M
	VMR [130]	animals	cate-mesh	V:M	None
(3)	PIFuHD [214]	human	None	S:I	V3
	NRSfM [6, 44]	any	None	V:J2	None
	N-NRSfM [230]	any	None	V:MF,M	None
	WSD [38]	dolphin†	cate-mesh	V:J2,M	None
	A3DC [203]	any	None	V:stroke	None
	LASR (Ours)	any	None	V:F,M	None

Model-based reconstruction: Model-based reconstruction leverages a parametric shape model to solve the under-constrained 3D shape and pose estimation problem. A large body of work in 3D human and animal reconstruction uses such parametric shape models [144, 182, 287, 333, 334], which are learned from 3D scans of human or animal toys [144, 333], and allow one to recover the 3D shape given very few annotations at test time (2D keypoints and silhouettes). Recently, model-based regression methods are developed to predict model-specific shape and pose parameters from a single image or video [9, 25, 117, 335], usually trained with ground-truth 3D data generated by such parametric shape models. Although model-based methods achieve great success in reconstructing "closed-world" objects of known category and rich 3D data, it is challenging to apply to unknown object categories, or categories with limited 3D data.

Category mesh reconstruction: A recent trend is to reduce supervision from 3D or multi-view capturing to 2D annotations, such as keypoints and object silhouettes [73, 108, 131]. Such methods often take advantage of category priors, including a collection of images from the same category, and category-specific shape templates [122, 259]. Recent progress makes single-view reconstruction of birds and other common categories possible without 3D annotation. However, the single view reconstruction is usually coarse and lacks instance-specific details. Recent work adapts category-specific models to a test-time video [130], but still does not handle objects of unknown classes.

Template-free reconstruction: Among the template-free methods, PIFu [213, 214] learns to predict an implicit shape representation for clothed human reconstruction, but requires ground-truth 3D shapes to train. A3DC [203] reconstructs articulated animals from videos, but requires involved user stroke interactions. Without requiring 3D data or user interactions, NRSfM factorizes a set of 2D keypoints or point trajectories into the 3D object shape and pose in each view assuming "low-rank" shape or deformation [6, 44, 74]. Recently, deep networks have been applied to learn such factorization of specific categories from 2D annotations [121, 176, 270]. Close to our approach, Neural Dense NRSfM (N-NRSfM) [230] learns a video-specific shape and deformation model from dense 2D point tracks. However, such methods are limited by the accuracy of 2D trajectory inputs, which is challenging to estimate in real-world sequences when large motion occurs [216, 230, 243].

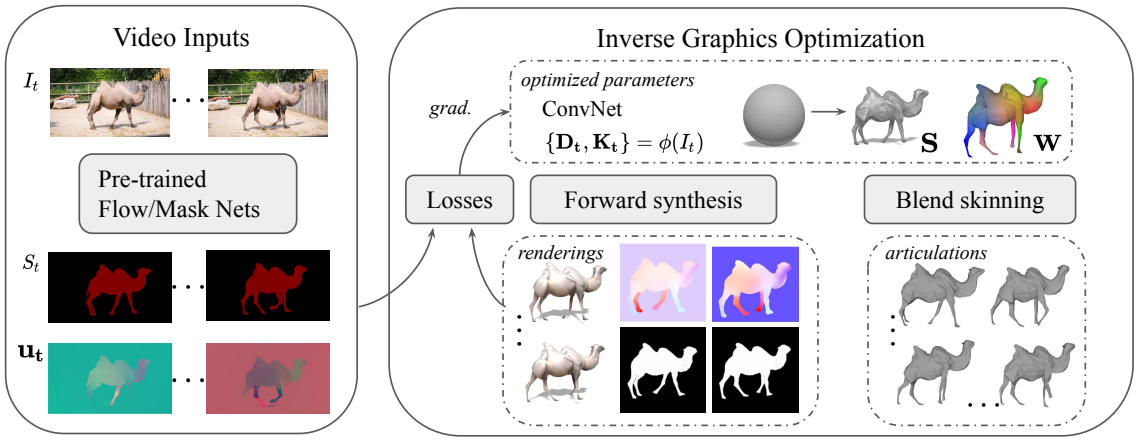

Figure 2.2: Method overview. Given a monocular video of an object, we jointly recover the object's rest shape $\mathbf{S}$, skinning weights $\mathbf{W}$, articulation $\mathbf{D_t}$, and camera parameters $\mathbf{K_t}$ by solving an inverse graphics problem through gradient-based optimization. The object rest shape is represented by a mesh (initialized from a sphere) and articulated at each frame under linear blend skinning (Sec. 2.3.2). The time-varying parameters, including focal length, object root transformation and articulation, are parameterized by a neural basis, i.e., convolutional network given image inputs (Sec. 2.3.4). At each iteration, we randomly sample pairs of consecutive frames and forward-render object silhouette, texture, and two-frame optical flow using a differentiable renderer (Sec. 2.3.1). The renderings are compared against raw pixels, segmentation and optical flow estimated by off-the-shelf networks to generate gradients signals and update the model (Sec. 2.3.3).

2.3 Approach

Problem: Given a monocular video $\{I_t\}$ with an object of interest (indicated by a segmentation mask $\{S_t\}$), we tackle the nonrigid 3D shape and motion estimation problem, which includes estimating (1) $\mathbf{S}$: the rest shape of the object, (2) $\mathbf{D_t}$: the time-varying articulations as well as the object root body transformations, and (3) $\mathbf{K_t}$: the camera intrinsics.

Overview: Figure 2.2 illustrates the overview of our method. Motivated by recent progress in differentiable rendering and self-supervised shape learning [108, 140], we cast the nonrigid 3D shape and motion estimation problem as an analysis-by-synthesis task. Despite the under-constrained nature of this problem, we hypothesize that, by giving appropriate video measurements, a "low-rank" shape and motion can be

solved up to an unknown scale. Model parameters $\mathbf{X} = \{\mathbf{S}, \mathbf{D_t}, \mathbf{K_t}\}$ are updated (via gradient descent) to minimize the difference between the rendered output $\mathbf{Y} = \mathbf{f}(\mathbf{X})$ and ground-truth video measurements $\mathbf{Y}^*$ at *test* time (Sec. 2.3.1). To deal with the fundamental ambiguities in object shape, deformation and camera motion, we seek (1) a "low-rank" but expressive parameterization of deformation (Sec. 2.3.2), (2) rich constraints provided by optical flow and raw pixels, and (3) appropriate regularization of object shape deformation and camera motion (Sec. 2.3.3).

2.3.1 Forward-synthesis model

We first introduce the forward synthesis model. Given a frame index t and model parameters $\mathbf{X}$, we synthesize the measurements of the corresponding frame pair $\{t, t+1\}$, including color images renderings $\{\hat{I}_t, \hat{I}_{t+1}\}$, object silhouettes renderings $\{\hat{S}_t, \hat{S}_{t+1}\}$ and forward-backward optical flow renderings $\{\hat{\mathbf{u}}_t^+, \hat{\mathbf{u}}_{t+1}^-\}$.

Rendering pipeline: We represent object shape as a triangular mesh $\mathbf{S} = \{\bar{\mathbf{V}}, \bar{\mathbf{C}}, \mathbf{F}\}$ with vertices $\bar{\mathbf{V}} \in \mathbb{R}^{N\times 3}$, vertex colors $\bar{\mathbf{C}} \in \mathbb{R}^{N\times 3}$ and a fixed topology $\mathbf{F} \in \mathbb{R}^{M\times 3}$. To model time-varying articulations $\mathbf{D_t}$, we have

$$\mathbf{V}_t = \mathbf{G_{0,t}}(\bar{\mathbf{V}} + \Delta\mathbf{V_t}) \tag{2.1}$$

where $\Delta\mathbf{V_t}$ is a per-vertex motion field applied to the rest vertices $\bar{\mathbf{V}}$, and $\mathbf{G_{0,t}} = \begin{pmatrix}[c|c]\mathbf{R_0} & \mathbf{T_0}\end{pmatrix}_\mathbf{t}$ is an object root body transformation matrix (index 0 is used to differentiate from bone transformations indexed from 1 in Sec. 2.3.2). Finally, we apply a perspective projection $\mathbf{K_t}$ before rasterization, where principal point (p_x, p_y) is assumed to be constant and focal length f_t varies over time to deal with zoom-in/out.

Shaders: We render object silhouette and color images with a differentiable renderer [140]. Color images are rendered given per-vertex appearance $\bar{\mathbf{C}}$ and constant ambient light. To synthesize the forward flow $\mathbf{u}_t^+$, we take surface positions $\mathbf{V}_t$ corresponding to each pixel in frame t, compute their locations $\mathbf{V}_{t+1}$ in the next frame, then take the difference of their projections:

$$\begin{pmatrix} u_{x,t}^+ \\ u_{y,t}^+ \end{pmatrix} = \begin{pmatrix} \mathbf{P}_t^{(1)}\mathbf{V}_t / \mathbf{P}_t^{(3)}\mathbf{V}_t \\ \mathbf{P}_t^{(2)}\mathbf{V}_t / \mathbf{P}_t^{(3)}\mathbf{V}_t \end{pmatrix} - \begin{pmatrix} \mathbf{P}_{t+1}^{(1)}\mathbf{V}_{t+1} / \mathbf{P}_{t+1}^{(3)}\mathbf{V}_{t+1} \\ \mathbf{P}_{t+1}^{(2)}\mathbf{V}_{t+1} / \mathbf{P}_{t+1}^{(3)}\mathbf{V}_{t+1} \end{pmatrix}, \tag{2.2}$$

where $\mathbf{P}^{(i)}$ denotes the ith row of the projection matrix $\mathbf{P}$.

2.3.2 Articulation Modeling

Unknowns vs constraints: Similar to NRSfM, we analyse the number of unknowns and constraints to solve the inverse problem. Given T frames of a video, we have

$$\begin{aligned} \#\text{ Unknowns} = & 3N + 3NT + 6T + (T+2), \\ & (\bar{\mathbf{V}}) \quad (\Delta\mathbf{V}) \quad (\mathbf{R_0}, \mathbf{T_0}) \quad (\mathbf{K}) \end{aligned} \tag{2.3}$$

which grows linearly with the number of vertices. Motivated by NRSfM [44] that uses low-rank shape and motion basis to deal with the exploding solution space, we seek an expressive but low-rank representation of shape and motion.

Linear-blend skinning: Instead of modeling deformation as per-vertex motion $\Delta\mathbf{V_t}$ [73, 108, 131], we adopt a linear-blend skinning model (LBS) [122, 127] to constrain vertex motion by blending B rigid "bone" transformations $\{\mathbf{G_1}, \cdots, \mathbf{G_B}\}$, which reduces the number of parameters and makes optimization easier. Besides bone transformations, the LBS model defines a skinning weight matrix $\mathbf{W} \in \mathbb{R}^{B\times N}$ that attaches the vertices of rest shape vertices $\bar{\mathbf{V}}$ to the set of bones. Each vertex is transformed by linearly combining the weighted bone transformations in the object coordinate frame and then transformed to the camera coordinate frame,

$$\mathbf{V}_{i,t} = \mathbf{G_{0,t}}(\sum\nolimits_b \mathbf{W}_{b,i}\mathbf{G}_{b,t})\bar{\mathbf{V}}_i \tag{2.4}$$

where i is the vertex index, b is the bone index. Unlike A-CSM [122] that only learns articulation, we learn skinning weights and time-varying bone transformations jointly.

Parametric skinning model: Inspired by the work on surface editing and local 3D shape learning [67, 240], we model the skinning weights as a mixture of Gaussians with B components. The probability of assigning vertex i to component b is given as

$$W_{b,i} = Ce^{-\frac{1}{2}(\mathbf{v}_i - \mathbf{J}_b)^T\mathbf{Q}_b(\mathbf{v}_i - \mathbf{J}_b)}, \tag{2.5}$$

where $\mathbf{J}_b \in \mathbb{R}^3$ is the center of b-th Gaussian, $\mathbf{Q}_b$ is the corresponding precision matrix that determines the orientation and radius of a Gaussian, and C is a normalization

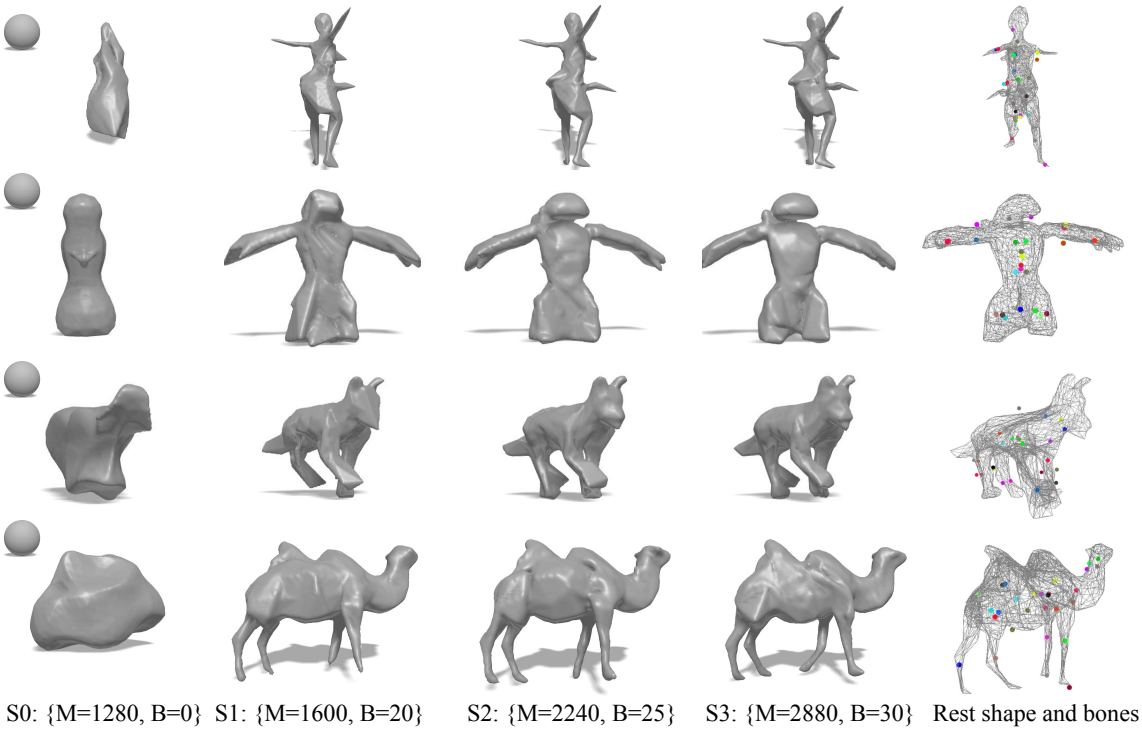

Figure 2.3: Coarse-to-fine reconstruction from S0 to S3. The learned centers of 3D Gaussians are shown as colored dots.

factor that ensures the probabilities of assigning a vertex to different Gaussians sum up to one. $\mathbf{W} \rightarrow \{\mathbf{Q}, \mathbf{J}\}$ is optimized. Note that the mixture of Gaussian models not only reduces the number of parameters for skinning weights from NB to $9B$, but also guarantees smoothness, the benefits of which are empirically validated in our experiments (Tab. 2.5). The number of shape and motion parameters now becomes

$$\begin{aligned} \#\text{ Unknowns} = & 3N + 6BT \quad + \quad 9B \quad + \quad 6T \quad + \quad (T+2), \\ & (\bar{\mathbf{V}}) \quad (\mathbf{G_{1\ldots B}}) \qquad (\mathbf{J}, \mathbf{Q}) \quad (\mathbf{R_0}, \mathbf{T_0}) \quad (\mathbf{K}) \end{aligned} \tag{2.6}$$

which grows linearly w.r.t.the number of frames and bones.

2.3.3 Self-supervised Learning from a Video

We exploit rich supervision signals from dense optical flow and raw pixels, as well as shape and motion regularizers to further constrain the problem.

Reconstruction Losses: The supervision for our analysis-by-synthesis pipeline includes silhouette loss, optical flow loss, texture loss, and perceptual loss. Given a pair of rendered outputs $(\hat{S}_t, \hat{I}_t, \hat{\mathbf{u}}_t)$ and measurements $(S_t, I_t, \mathbf{u}_t)$, the inverse graphics loss is computed as,

$$L_{\text{IG}} = \beta_1 ||\hat{S}_t - S_t||_2^2 + \beta_2 \sigma_t ||\hat{\mathbf{u}}_t - \mathbf{u}_t||_2 + \beta_3 ||\hat{I}_t - I_t||_1 + \beta_4 pdist(\hat{I}_t, I_t) \tag{2.7}$$

where $\{\beta_1, \cdots, \beta_4\}$ are weights empirically chosen, σ_t is the normalized confidence map for flow measurement, and $pdist(\cdot, \cdot)$ is the perceptual distance [325] measured by an AlexNet pretrained on ImageNet. Applying L2 norm loss to optical flow is empirically better than squared L2 loss, and we hypothesize the reason being that the former is more tolerant to outliers in the observed flow fields.

Shape and motion regularization: We exploit generic shape and temporal regularizers to constrain the problem. A Laplacian operator is applied to the rest mesh to enforce smooth surfaces,

$$L_{\text{shape}} = \sum_{i=1}^{V} ||\bar{\mathbf{V}}_i - \frac{1}{|N_i|} \sum_{j \in N_i} \bar{\mathbf{V}}_j||^2, \tag{2.8}$$

where N_i denotes the set of adjacent vertices of vertex i. Motion regularization includes an ARAP (as-rigid-as-possible) deformation term and a least deformation term. The ARAP term encourages natural deformation [240, 259],

$$L_{\text{ARAP}} = \sum_{i=1}^{V} \sum_{j \in N_i} | \; ||\mathbf{V}_{i,t} - \mathbf{V}_{j,t}||_2 - ||\mathbf{V}_{i,t+1} - \mathbf{V}_{j,t+1}||_2 \; |. \tag{2.9}$$

The least deformation term encourages the deformation from the rest shape to be small [108],

$$L_{\text{least-motion}} = \sum_{i=1}^{V} ||\mathbf{V}_{i,t} - \bar{\mathbf{V}}_i||_2, \tag{2.10}$$

which discourages arbitrarily large deformations and reduces ambiguities in joint object root body pose and articulation recovery.

Soft-symmetry constraints: To exploit the reflectional symmetry structure exhibited in common objects, we pose a soft-symmetry constraint along the symmetry

plane $(\mathbf{n}^*, 0)$ at an arbitrary frame t^*. The symmetry plane is initialized from visual inspection and jointly optimized. We encourage the rest shape to be similar to its reflection,

$$L_{\text{symm-shape}} = L_{\text{cham}}(\{\bar{\mathbf{V}}, \mathbf{F}\}, \{\mathbf{H}\bar{\mathbf{V}}, \mathbf{F}\}) \tag{2.11}$$

where $\mathbf{H} = \mathbf{I} - 2\mathbf{n}_*\mathbf{n}_*^T$ is the Householder reflection matrix, and the Chamfer distance (L_{cham}) is computed as bidirectional pixel-to-face distances. For the centers of Gaussian control points $\mathbf{J}$, we also have

$$L_{\text{symm-bone}} = L_{\text{cham}}(\bar{\mathbf{J}}, \mathbf{H}\bar{\mathbf{J}}). \tag{2.12}$$

The total loss is a weighted sum of all losses with the weights empirically chosen and fixed for all experiments.

2.3.4 Implementation Details

Neural basis for time-varying parameters: Instead of optimizing explicit time-varying parameters $\{\mathbf{D_t}, \mathbf{K_t}\}$, we parameterize those as predictions from a convolutional network (ResNet-18 [80]) given an input image $\mathbf{I}_t$,

$$\psi_w(\mathbf{I}_t) = (\mathbf{K}, \mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2, \cdots, \mathbf{G}_B)_t, \tag{2.13}$$

where one parameter is predicted for focal length, four parameters are predicted for each bone rotation parameterized by quaternion, and three numbers are predicted for each translation, adding to $1 + 7(B + 1)$ numbers in total at each frame. The weights are initialized with ImageNet [47] pre-training and then optimized by LASR for each test video. Intuitively, the network learns a joint basis for cameras and poses that is empirically much easier to optimize than the raw parameters (Tab. 2.5).

Silhouette and flow measurements: We assume reliable segmentation of the foreground object is given, which can be manually annotated [187], or estimated using instance segmentation and tracking methods [116, 329]. Our method requires reasonable optical flow estimation, which can be provided by state-of-the-art flow estimators [249, 301]. Notably, LASR recovers from some bad flow initialization and obtains better long-term correspondences (Tab. 2.3).

Table 2.2: Choices of hyper-parameters for training.

Name	Value
Optimization parameters	
Network architecture of ϕ_w	ResNet-18 (ImageNet-pretrained) [80]
Optimizer	Adam [114]
Learning rate for ϕ_w	1×10^{-4}
Learning rate for other params.	5×10^{-3}
Batch size	8 image pairs
Loss weight $\{\beta_1, \ldots, \beta_4\}$	$\{0.5, 0.5, 2, 5 \times 10^{-3}\}$
Measurement pre-processing	
Crop center	Center of object bounding box
Crop size	$1.2 \times$ longest edge
Resized to	256×256

Coarse-to-fine reconstruction: We adopt a coarse-to-fine strategy to reconstruct high-quality meshes inspired by Point2Mesh [78]. **S0**: We first assume a rigid object and optimize the rest shape and cameras $\{\mathbf{S}, \mathbf{G}_{\mathbf{0,t}}, \mathbf{K}_\mathbf{t}\}$ for 20 epochs. The rest shape is initialized from a subdivided icosahedron projected onto a sphere. **S1-S3:** We perform iso-surface extraction and re-meshing [86] to fix mesh self-intersections and long edges. After remeshing, the number of vertices and the number of bones increase, as shown in Fig. 2.3. The centers of Gaussian control points are initialized by running K-means on the vertices coordinates. We then jointly optimize all parameters $\{\mathbf{S}, \mathbf{D}_\mathbf{t}, \mathbf{K}_\mathbf{t}\}$ for 10 epochs. The above procedure is repeated three times (S1-S3).

Training details: We include details of the hyper-parameters used for training in Tab. 2.2.

2.4 Experiments

Setup: Due to the difficulty of obtaining 3D ground truth for nonrigid objects in the real world, we evaluate 2D keypoint transfer accuracy as a proxy of 3D reconstruction quality on real videos. We additionally evaluate 3D reconstruction accuracy on objects with ground-truth meshes.

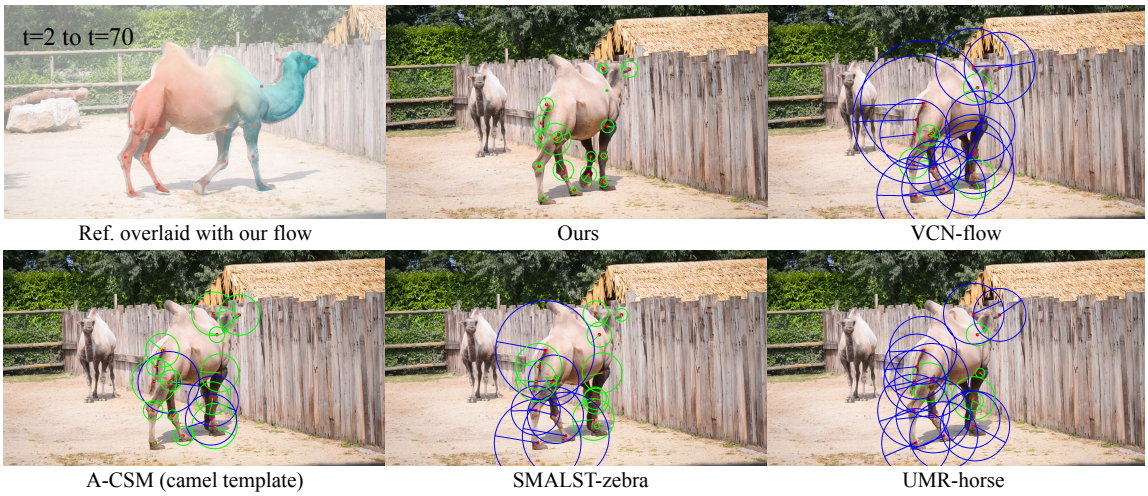

Figure 2.4: Keypoint transfer between frame 2 and frame 70 of the camel video. The distance between tranferred keypoint and target annotation is represented by the radius of circles. A correct transfer is marked with green and a wrong transfer is marked with blue. Our method transfers keypoint more accurately than baselines.

2.4.1 2D Keypoint Transfer on Animal Videos

Dataset: We test our method on an animal video dataset, **BADJA** [24], which provides nine real animal videos with 2D keypoint and mask annotations, derived from the DAVIS video segmentation dataset [187] and online stock footage. It includes three videos of dogs, two videos of horsejump, and one video of camel, cow, bear as well as impala. We report quantitative results on one video per-category and show the reconstruction of the rest in the sup. mat.

Metric: To approximate the accuracy of 3D shape and articulation recovery, we adopt **percentage of correct keypoint transfer** (PCK-T) [108, 122, 311] metric. Given a reference and target image pair with 2D keypoint annotations, the reference keypoint is transferred to the target image, and labeled as "correct" if the transferred keypoint is within some threshold distance $d_{th} = 0.2\sqrt{|S|}$ from the target keypoint, where $|S|$ is the area of the ground-truth silhouette [24]. In practice, we transfer points by re-projection from the reference frame to the target frame given the articulated shape and camera pose estimations. If the back-projected keypoint lies outside the reconstructed mesh, we re-project its nearest neighbor that intersects the mesh. The accuracy is averaged over all T(T-1) pairs of frames.

Baselines: We compare with state-of-the-art methods for animal reconstruction and refer to Tab. 2.1 for a taxonomy. **SMALST** [335] is a model-based regressor trained for zebras. It takes an image as input and predicts shape, pose and texture for the SMAL [333] model. **UMR** [131] is a category-specific shape estimator trained for several categories, including birds, horses and other categories that have a large collection of annotated images. We report the performance of the horse model since the models of other animal categories are not available. **A-CSM** [122] learns a category-specific canonical surface mapping and articulations from an image collection. At test time, it takes an image as input and predicts the articulation parameters of a rigged template mesh. It provides 3D templates for 27 animal categories and an articulation model for horses, which is used throughout the experiments. **SMALify** [24] is a model-based optimization approach that fits one of five categories (including cat, dog, horse, cow and hippo) of SMAL models to a video or a single image. We provide all the video frames with ground-truth keypoint and mask annotations. Close to our setup, **N-NRSfM** [230] trains a video-specific model for object shape, deformation and camera parameters from multi-frame optical flow estimations [61]. Finally, we include a detection-based method, **OJA** [24], which trains an hourglass network to detect animal keypoints (indicated by Detector), and post-process the joint cost maps with a proposed optimal assignment algorithm. The results of PCK are taken from the paper [24] without recomputing PCK-T.

Results: Qualitative results of 3D shape reconstruction are shown in Fig. 3.1 and Fig. 2.5, where we compare with UMR, A-CSM and SMALify on the camel, bear and dog video. Quantitative results of keypoint transfer are shown in Tab. 2.3. Given that all 3D reconstruction baselines are category-specific and might not provide the exact model for some categories (such as camel), we pick up the best model or template for each animal video. Compared with 3D reconstruction baselines, LASR is better for all categories, even on the categories the baselines are trained for (e.g., LASR: 49.3 vs UMR: 32.4 on horsejump-high). Replacing the GT masks with an object segmentor, PointRend [116], the performance of LASR ('+Auto-mask' in Tab. 2.3) drops, but is still better than all the reconstruction baselines. Compared to detection-based methods, our accuracy is higher on the horsejump video, and close to the baseline on other videos. LASR also shows a large improvement compared to the initial optical flow (81.9% vs 47.9% for camel), especially between long-range frames as in Fig. 2.4.

Table 2.3: 2D Keypoint transfer accuracy on BADJA. (1)Model-based regression. (2)Category-specific reconstruction. (3)Free-form reconstruction. Methods with† do not reconstruct 3D shape. Results with* indicates the method is not designed for such category. Best results are underlined, and bolded if reconstruct a 3D shape.

Method	camel	dog	cows	horse	bear
(1)SMALST [335]	49.7*	12.8*	59.7*	10.4*	67.2*
(2)A-CSM [122]	60.2*	24.5*	65.7*	21.5	39.7*
(2)UMR [131]	35.1*	38.5*	68.1*	32.4	56.9*
(3)N-NRSfM [230]	67.8	17.9	70.0	8.7	60.2
(3)LASR (Ours)	**81.9**	**65.8**	**83.7**	**49.3**	**85.1**
(3) +Auto-mask	78.9	59.5	82.7	42.2	82.6
†Static	51.9	13.0	55.5	8.8	58.6
†Detector [24]	73.3	66.9	89.2	26.5	83.1
†OJA [24]	87.1	66.9	94.7	24.4	88.9
†Flow-VCN [301]	47.9	25.7	60.7	14.4	63.8

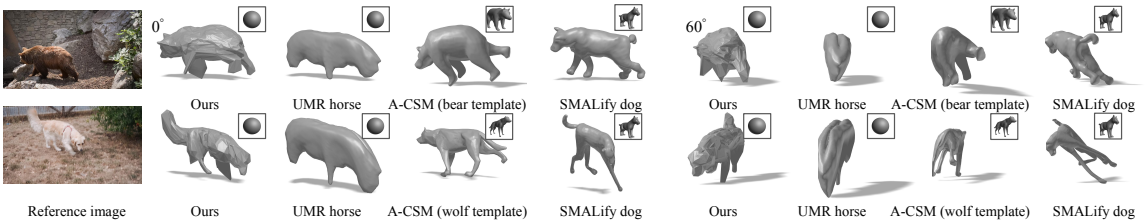

Figure 2.5: Comparison on BADJA bear and dog videos. The reconstruction of the first frame of the video is shown from two viewpoints. Compared to UMR that also does not use a shape template, LASR reconstructs more fine-grained geometry. Compared to A-CSM and SMALify that use a shape template, LASR recovers instance-specific details, such as the fluffy tail of the dog, and a more natural pose.

2.4.2 Mesh Reconstruction on Articulated Objects

Dataset: To evaluate mesh reconstruction accuracy, we collect a video dataset of five articulated objects with ground-truth mesh and articulation, including one dancer video from AMA (Articulated Mesh Animation dataset) [266], one German shepherd video, one horse video, one eagle video and one stone golem video from TurboSquid. We also include a rigid object, Keenan's spot to evaluate performance on rigid object reconstruction and ablation for the S0 stage.

Table 2.4: Mesh reconstruction error in terms of Chamfer distance on our animated object dataset. To ensure comparable results over objects, ground-truth shapes are rescaled such that the maximum distance between vertices is 10. Best results are bolded. "-" means a method does not apply to a particular sequence.

Method	dancer ↓	dog ↓	horse ↓	golem ↓
SMPLify-X [182]	0.26	-	-	-
VIBE [117]	**0.22**	-	-	-
A-CSM [122]	-	0.38	0.26	-
SMALify [24]	-	0.51	0.41	-
PIFuHD [214]	0.28	-	-	-
UMR [131]	-	0.44	0.42	-
LASR (Ours)	0.35	**0.28**	**0.23**	0.16

Metric: Most prior work on mesh reconstruction assumes given camera parameters. However, both the camera and the geometry are unknown in our case, which leads to ambiguities in evaluation, including scale ambiguity (exists for all monocular reconstruction) as well as the depth ambiguity (exists for weak perspective cameras as used in UMR, A-CSM, VIBE, etc.). To factorize out the unknown camera matrices, we align two meshes with a 3D similarity transformation solved by ICP. Then, the bidirectional Chamfer distance is adopted as the evaluation metric. We follow prior work [68, 202] to randomly sample 10k points uniformly from the surface of predicted and ground-truth meshes, and compute the average distance between the nearest neighbor for each point in the corresponding point cloud.

Baselines: Besides A-CSM, SMALify, and UMR for animal reconstruction, we compare with SMPLify-X, VIBE, and PiFUHD for human reconstruction. **SMPLify-X** [182] is a model-based optimization method for expressive human body capture. We use the female SMPL model for the dancer sequence, and provide the keypoint inputs estimated from OpenPose [36]. **VIBE** [117] is a state-of-the-art model-based video regressor for human pose and shape inference. **PIFuHD** is a state-of-the-art free-form 3D shape estimator for clothed humans. It takes a single image as input and predicts an implicit shape representation, which is converted to a mesh by the marching cube algorithm. To compare with SMALify on dog and horse, we manually annotate 18 keypoints per-frame, and initialize with the corresponding shape template.

Table 2.5: Ablation study with mesh reconstruction error.

S0	ref.	(1)w/o flow	(2)w/o L_{can}	(3)w/o CNN
spot	0.05	0.55	0.61	0.63
S0-S3	ref.	(4)w/o LBS	(5)w/o C2F	(6)w/o GMM
dog	0.28	0.68	0.59	0.34

Results The visual comparison on human and animals are shown in Fig. 3.1 and Fig. 2.6 respectively. We report the quantitative results in Tab. 2.4. On the dog video, our method is better than all the baselines (0.28 vs A-CSM: 0.38), possibly because A-CSM and UMR are not trained specifically for dogs (although A-CSM uses a wolf template), and SMALify cannot reconstruct a natural 3D shape from limited keypoint and silhouette annotations. For the horse video, our method is slightly better than A-CSM, which uses a horse shape template, and outperforms other baselines. For the dancer sequence, our method is not as accurate as baseline methods (0.35 vs VIBE: 0.22), which is expected given that all baselines either use a well-designed human model, or have been trained with 3D human mesh data, while LASR does not have access to 3D human data. For the stone golem video, our method is the only one that reconstructs a meaningful shape. Although the stone golem has a similar shape to a human's, OpenPose does not detect joints correctly, leading to the failure of SMALify-X, VIBE and PiFUHD.

Qualitative results on DAVIS videos: To examine the performance on arbitrary real-world objects, we use five DAVIS videos, including dance-twirl, scooter-board, soapbox, car-turn, mallard-fly, and a cat video captured by us and segmented by PointRend. The comparison with COLMAP [222], a template-free SfM-MVS pipeline, is shown in Fig. 2.8. More results are available in the sup. mat.

Ablation study: We investigate the effect of different design choices on the rigid "spot" and animated dog sequences. The videos are rendered into T=15 frames given ambient light and a camera rotating around the object by 90 degrees at zero elevation. Besides color images, we render silhouette and optical flow as the supervision. Results are shown in Fig. 2.7 and quantitative results are reported in Tab. 2.5. In terms of camera parameter optimization and rigid shape reconstruction (**S0**), we

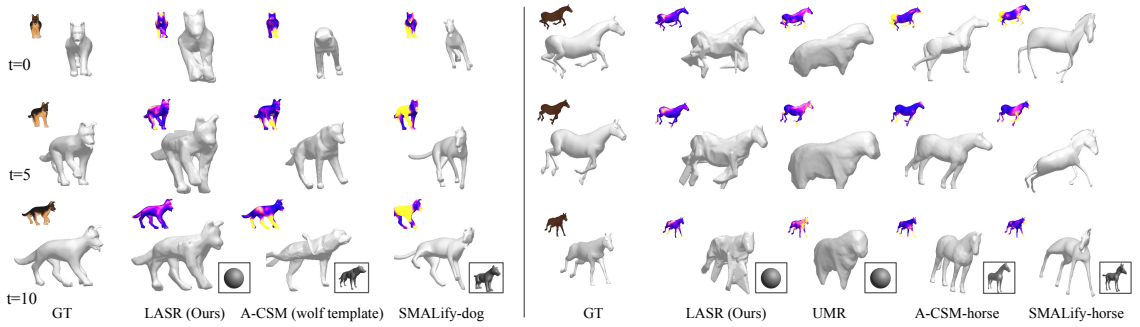

Figure 2.6: Shape and articulation reconstruction results on synthetic dog and horse sequences. We also visualize Chamfer distances measured from the ground truth to the reconstructed mesh on top of each result, and yellow indicates high error. Compared to UMR, LASR successfully reconstructs the four legs of the horse. Compared to template-based methods (A-CSM and SMALify), LASR successfully reconstructs the instance-specific details (ears and tails of the dog) and recovers a more natural articulation. The reference is shown at the left corner and the template mesh used is shown in the bottom right boxes.

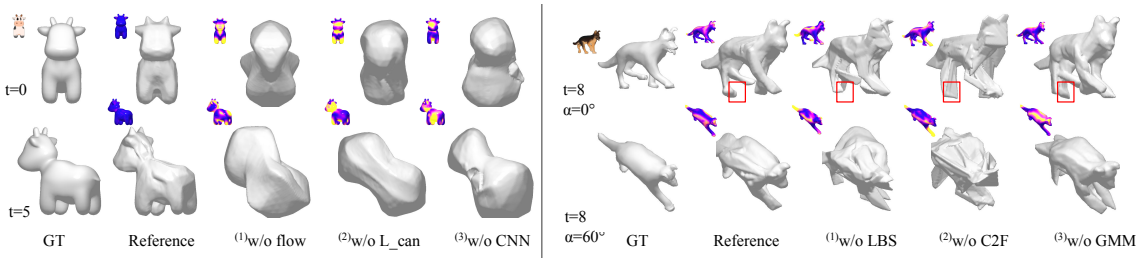

Figure 2.7: Left: Ablation study on camera and rigid shape optimization (**S0**). Removing the optical flow loss introduces large errors in camera pose estimation and therefore the overall geometry is not recovered. Removing the canonicalization loss leads to worse camera pose estimation, and therefore the symmetric shape constraint is not correctly enforced. Finally, if we directly optimize the camera poses without using a convolutional network, it converges much slower and does not yield an ideal shape within the same iterations. Right: Ablation study on articulated shape optimization (**S1-S3**). We show the reconstructed articulated shape at the middle frame (t=8) from two viewpoints. Without LBS model, although the reconstruction looks plausible from the visible view, it does not recover the full geometry due to the redundant deformation parameters and lack of constraints. Without coarse-to-fine re-meshing, fine-grained details are not recovered. Replacing GMM skinning weights (9xB parameters) with an NxB matrix leads to extra limbs and tails on the reconstruction.

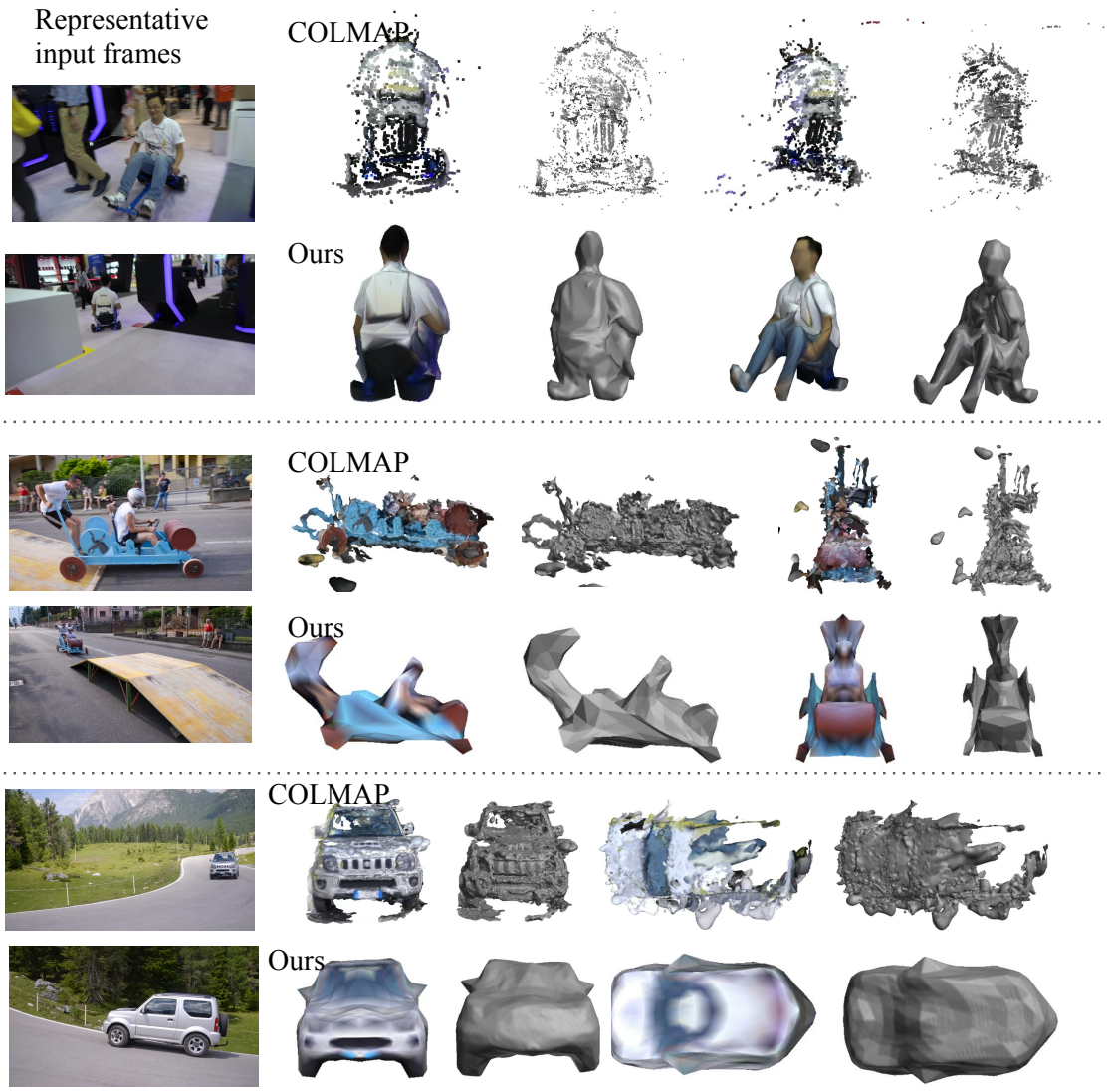

Figure 2.8: Visual comparison on near-rigid DAVIS sequences: scooter, soapbox, and car. COLMAP reconstructs only the visible rigid part, while our method faithfully reconstructs both the rigid object and near-rigid person.

find it beneficial to use (1)optical flow as supervision signals, (2)canonicalization of symmetry plane, and (3)CNN as an implicit representation for camera parameters. For articulated shape reconstruction (**S1-S3**), it is critical to use (4)linear blend skinning, (5)coarse-to-fine re-meshing, and (6)parametric skinning model.

Chapter 3

Canonical Embeddings for Video Pixel Registration

3.1 Introduction

Reconstructing the world from a sequence of monocular frames is a long-standing task in computer vision. While there has been tremendous progress in reconstructing rigid scenes (via SfM and SLAM [54, 233, 253], or recent techniques based on neural rendering [165]), reconstructing *dynamic* scenes with articulated objects remains elusive. For example, given a monocular video, it is still challenging to reconstruct an everyday scene of a moving person with loose clothing. In this work, we tackle the problem of estimating the deforming mesh of articulated objects given a segmented monocular video of that object. Our method avoids the use of any mesh templates or category-specific priors and generalizes to unknown deformable articulated objects in the wild.

Nonrigid shape recovery is highly under-constrained due to fundamental ambiguities between shape, appearance, and time-varying deformation. Current approaches for addressing these challenges fall into two camps: better data "likelihoods" or better "priors". The first camp extracts richer sensor data, via multi-camera studio setups [103] or depth sensors [172], but requires substantial efforts to work in the wild. The second camp makes use of category-level priors over object shapes [117, 122] and is particularly effective for human reconstruction. However, building such models requires considerable offline efforts in the form of registered 3D scans [144] or manual

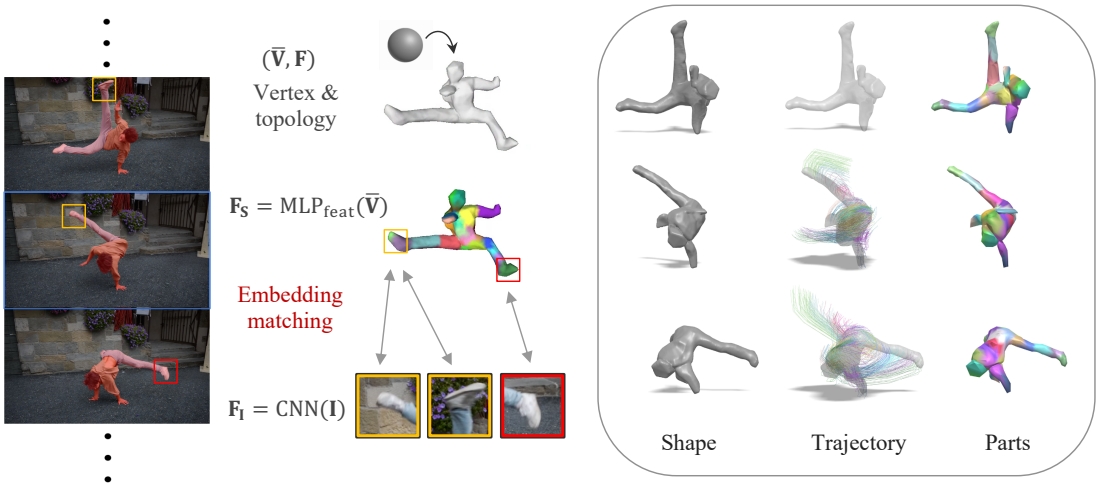

Figure 3.1: Given a long video (or multiple short videos), ViSER jointly learns articulated 3D shapes (represented as a mesh with vertices $\bar{\mathbf{V}}$ and faces $\mathbf{F}$) and joint pixel-surface embeddings (including a surface embedding $\mathbf{F_S}$ and a pixel embedding $\mathbf{F_I}$) that establishes dense long-range pixel correspondences over time. As a result, ViSER produces accurate shapes, long term trajectories and meaningful part segmentation.

keypoint annotations [76], both of which are difficult to scale to arbitrary object categories.

In this work, we use a practical but less explored variant of the data-likelihood camp: we use *multiple frames* of a video rather than multiple cameras or depth sensors. This considerably complicates analysis for dynamic, non-rigid scenes. Nonrigid structure-from-motion (NRSfM) [29, 230] attempts to constrain the problem by relying on motion correspondences such as 2D point tracks. While 2D correspondences over short time scales (i.e., optical flow) are relatively robust to extract, correspondences over long time scales are notoriously difficult to estimate because of appearance variations arising from viewpoint changes, occlusion and fast motion. In practice, this limits the applicability of NRSfM methods to controlled lab sequences.

We propose ViSER (Video-Specific Surface Embeddings for Reconstruction), which establishes long-range correspondence and reconstructs articulated 3D shapes from a monocular video. Fig. 3.1 shows a sample outdoor video and the corresponding ViSER results. The key insight behind ViSER is to force long-range video pixel correspondences to be consistent with an underlying canonical 3D mesh through the

use of video-specific embeddings that capture the pixel appearance of each surface point. These embeddings behave as a continuous set of keypoint descriptors defined over the surface mesh, learned with coordinate-based MLPs that are fit to each video via self-supervised losses. ViSER simultaneously optimizes the image CNN, surface MLP, and 3D shape so as to fit the observed video frames. It reconstructs state-of-the-art articulated 3D shape and 3D trajectories without using category-specific priors, making it easily scalable to diverse videos including humans with challenging clothing and poses as well as animals. Lastly, we demonstrate that ViSER recovers meaningful part segmentation and blend skinning weights from videos, which typically require considerable manual effort from 3D artists.

3.2 Related Work

Low-level correspondence. Optical flow is a well-studied representation for short-term correspondence between adjacent frames of a video. After decades of research, recent CNN models [242, 249, 301] for optical flow have achieved an impressive level of accuracy as evidenced by the Sintel and KITTI benchmarks [32, 65]. However, it is challenging to concatenate optical flow for reliable long-range correspondence due to occlusions and strong appearance changes [207, 216, 243]. ViSER does not concatenate optical flow but use it as a constraint to establishes long-range correspondence.

The layered approach [45, 100, 272] segments a video into different moving objects with coherent motion, thereby establishing long-range correspondence for every frames through the shared layers. Early layered methods assume parameter motion for each layer and can only handle limited scenes. Unwrap Mosiacs [201] uses a dense 2D-to-2D mapping from a texture map to every input frame, and editing operations on the texture map naturally transfers to each individual frame. However, the 2D representation cannot flexibly model complex 3D phenomena, such as occlusions.

Dense pose and surface mappings. DensePose [76] directly maps pixels to the 3D surface of a human body model. It requires large amounts of training data with annotated image-to-surface correspondence and is hard to generalize to other categories. Articulation-aware Canonical Surface Mapping (A-CSM) [122] uses geometric cycle consistency for learning to map pixels to corresponding points on a template shape without using keypoint annotations. However, it requires a pre-

defined template shape for each category. Continuous Surface Embeddings (CSE) [170] establishes dense correspondences between image pixels and 3D object geometry by predicting an embedding vector of the corresponding vertex in the object mesh for each pixel in a 2D image. While applicable to multiple categories, CSE requires annotations and only applies to categories in the training set. ViSER requires neither a template shape nor annotations to work on categories in the wild.

Nonrigid shape reconstruction. One way to accurately reconstruct articulated shapes is to rely on rich sensor data, *e.g.*, multi-view [103] or depth sensors [172], which requires substantial efforts to setup and reconstruct objects in the wild. For monocular videos/images, one popular approach is to adopt strong 3D shape and pose priors [117, 144, 213, 214, 333, 334] but it works well only on limited categories, whose 3D data are easy to collect. To deal with more nonrigid object categories, a recent trend is to learn a category-level 3D shape model from a collection of images or videos with 2D annotations, such as keypoints and object silhouettes [73, 108, 122, 130, 131, 259, 282, 316]. Although they are able to reconstruct more object categories, such as birds and quadruped animals, the reconstruction usually lacks details, and the level of deformation recovered tends to be low.

Category-agnostic methods, such as nonrigid structure from motion (NRSfM) methods [29, 74, 121, 230] reconstruct nonrigid 3D shapes from a set of 2D point trajectories. However, due to the difficulty in obtaining accurate long-range correspondences [216, 243] they do not work well for videos in the wild. A recent work, LASR [306], uses two-frame optical flow to reconstruct articulate shapes from a monocular video with differentiable rendering. Despite the promising results, LASR does not reason about long-range correspondences and can only reliably reconstruct what is visible in a short video. ViSER establishes reliable long-range correspondence that are robust to moderate shape variations and appearance changes. Thus, ViSER can obtain much higher-quality reconstruction by using either a long video or several videos of a category.

3.3 Approach

Fig. 3.2 provides an overview of our approach, which follows a typical framework of differentiable rendering [108, 140]. Borrowing the notation from LASR [306], we

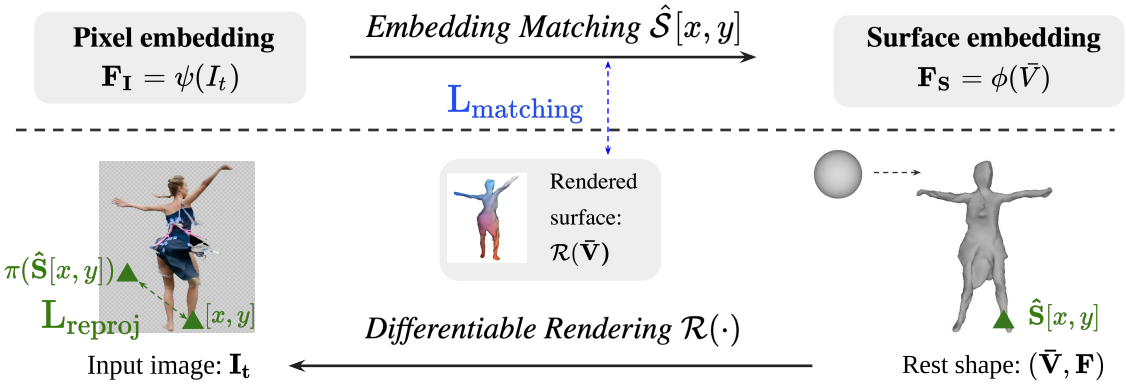

Figure 3.2: We learn a joint pixel-surface embedding space for dense correspondence between pixels in video frames I_t and points on a canonical 3D surface $(\bar{\mathbf{V}}, \mathbf{F})$. Such embedding space is optimized through "top-down" differentiable rendering $\mathcal{R}(\cdot)$ and "bottom-up" correspondence matching $\hat{\mathbf{S}}[x, y]$ (Sec 3.2). We introduce a 3D matching loss to optimize the embeddings, where the matched surface locations are encouraged to be close to the rendered surface locations. The embedding further enables articulated shape optimization through a 2D-3D-2D cycle reprojection: pixel $[x, y] \rightarrow$ matched surface $\hat{\mathbf{S}}[x, y] \rightarrow$ re-projected pixel $\pi(\hat{\mathbf{S}}[x, y])$ (Sec. 3.3).

formalize our task as follows. Given a set of video observations including RGB pixel color, segmentation masks, and optical flow estimates $\{I_t, S_t, u_t\}_{t=\{0,\ldots,T\}}$, our goal is to recover a set of shape and motion parameters $\{\mathbf{S}, \mathbf{D}_t\}$ that produce reconstructions $\{\hat{I}_t, \hat{S}_t, \hat{u}_t\}_{t=\{0,\ldots,T\}}$ that match the video observations. We refer to supplementary material for a complete list of notations defined in the paper.

3.3.1 Preliminaries

We represent an object's shape as a triangular mesh $\mathbf{S} = \{\bar{\mathbf{V}}, \mathbf{F}\}$ with canonical vertices $\bar{\mathbf{V}} \in \mathbb{R}^{3\times N}$ and a fixed topology (edge connectivity) $\mathbf{F} \in \mathbb{R}^{3\times M}$. To render an object, we displace mesh vertices with motion parameters $\mathbf{D}_t$, apply a perspective projection with camera intrinsics $\mathbf{K}_t$, and rasterize.

We model vertex motion with root body transformations $\mathbf{G}_0$ and object articulations $\{\mathbf{G_1}, \cdots, \mathbf{G_B}\}$ using linear blend skinning (LBS) [122, 127]. LBS constrains vertex motion by linearly blending B rigid "bone" transformations with a skinning

weight matrix $\mathbf{W} \in \mathbb{R}^{B \times N}$, transforming the canonical shape into frame t as

$$\mathbf{V}_{i,t} = \mathbf{G}_{\mathbf{0},\mathbf{t}} \left(\sum_b \mathbf{W}_{b,i} \mathbf{G}_{b,t} \right) \bar{\mathbf{V}}_i \tag{3.1}$$

where i is the vertex index, b is the bone index. Similar to LASR, the root body and bone transformations are represented as the outputs of a pose CNN given an input image, $(\mathbf{G_0}, \cdots, \mathbf{G_B}) = \psi_p(I_t)$.

We define a set of surface properties for rendering, including vertex 3D coordinates, textures and features, and rasterize them in a differentiable manner [140]. We denote the differentiable rendering function that renders the property $\mathbf{C}$ defined on a canonical surface to an image as $\mathcal{R}(\mathbf{C}; \mathbf{V}, \mathbf{W}, \mathbf{G})$, which executes the blending skinning function in Eq. (3.1) and softly blends the surface property based on their depth and barycentric coordinates [140]. For simplicity, we omit the shape, skinning, and motion parameters parameters and write the differentiable rendering function as $\mathcal{R}(\mathbf{C})$. To render optical flow, we rasterize and project vertex coordinates in two consecutive frames and compute their 2D displacements [306]. Such renderings are compared against video observations to compute gradients for updating model parameters.

3.3.2 Video-specific Surface Embedding

Pixel-surface embeddings. We learn pixel and surface embeddings that map corresponding pixels in different frames to the same point on a canonical 3D surface. Intuitively, consider a particular region on the canonical surface mesh that is the "nose" of an articulated human. The surface embedding captures a descriptor for the nose, which can then be matched to pixel-level descriptors at each frame.

Given an input image I_t, the pixel-wise descriptor embedding is computed by a U-Net [210] encoder:

$$\mathbf{F_I}[x, y, t] = \psi_e(I_t)[x, y] \in \mathbb{R}^{16}, \tag{3.2}$$

where $[x, y, t]$ are pixel locations at frame t. The surface embedding is computed by a position-encoded MLP:

$$\mathbf{F_S}(X, Y, Z) = \phi_e(X, Y, Z) \in \mathbb{R}^{16}, \tag{3.3}$$

where $\phi_e(\cdot)$ is an MLP defined over 3D points (X, Y, Z) in the canonical space, augmented with Fourier positional encoding [165]. The two embeddings are optimized on test videos such that pixels representing the same surface location in different frames are mapped to the same canonical surface point [122].

Correspondence via soft-argmax regression. Given the pixel and surface embeddings, we construct a per-frame cost volume $D(\mathbf{F_I}, \mathbf{F_S})$ of size $H \times W \times N_s$ over pixels and surface points (we randomly sample $N_s = 200$ surface points at each step) by considering their cosine feature distances,

$$D(\mathbf{F_I}, \mathbf{F_S})[x, y, i] = 1 - \cos\big(\mathbf{F_I}[x, y], \mathbf{F_S}(X_i, Y_i, Z_i)\big). \tag{3.4}$$

Normalizing the cost volume over the surface point dimension yields a softmax "heatmap" over surface points that potentially match to pixel (x, y), as shown in Fig. 3.3 (Left):

$$\sigma_{(x,y)}[i] = \frac{e^{-D[x,y,i]/\tau}}{\sum_j e^{-D[x,y,j]/\tau}}, \tag{3.5}$$

where τ is a temperature scaling parameter that is jointly optimized with the feature embeddings. To output a single surface point for pixel (x, y), we can compute a "soft" argmax [112, 301] by taking the expectation of the softmax distribution over the 3D locations of the points samples,

$$\hat{\mathbf{S}}[x, y] = \sum_i \sigma_{(x,y)}[i](X_i, Y_i, Z_i), \tag{3.6}$$

where (X_i, Y_i, Z_i) is the i-th sampled surface point and $\sigma_(x, y)[i]$ is the matching probability of pixel (x, y) over the sampled points $i \in \{1, 2, \ldots, N_s\}$.

We can also normalize the $H \times W \times N_s$ cost volume over spatial positions to capture a distribution of pixel locations that match to each surface point (X_i, Y_i, Z_i):

$$\sigma_{(X_i,Y_i,Z_i)}[x, y] = \frac{e^{-D[x,y,i]/\tau}}{\sum_{[x,y]} e^{-D[x,y,j]/\tau}}. \tag{3.7}$$

and compute a similar soft argmax mapping of surface points to pixels, as shown in Fig. 3.3 (Right).

Relation to keypoints. The output of classic keypoint detectors are often repre-

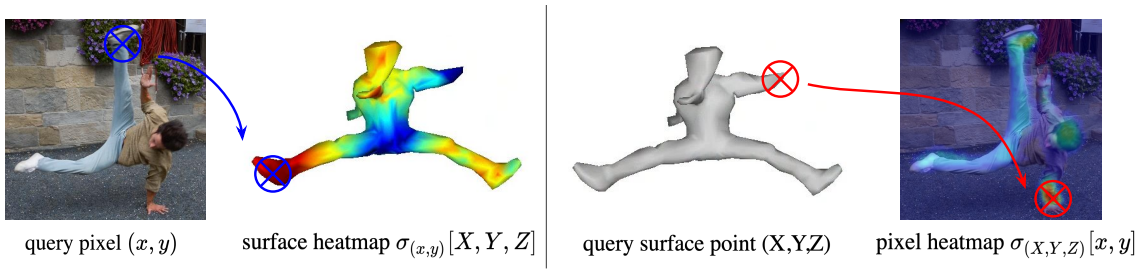

Figure 3.3: Pixel-surface embeddings establish a continuous mapping between pixels and points on a canonical surface. **Left:** Given a query pixel at (x,y), we match it to a set of canonical surface points, where the matching distribution is used to regress a continuous mapping to the canonical surface. **Right:** Given a query surface point (X,Y,Z), a matching distribution over pixels can be computed. Warm color indicates high matching probability.

sented as K-channel heatmaps over the pixel grid, where K is the number of keypoints. To define dense keypoints, one may increase the number of channels, which is computationally heavy. Similar to CSE [170], we represent dense keypoints as low-dimensional pixel-surface embeddings, which establishes a mapping between pixels and a canonical 3D surface, but far more efficiently. DensePose [76] and CSM [122] use an alternative pixel-to-surface mapping that regresses a surface coordinate at every pixel. In contrast, our pixel-surface embedding captures multimodal uncertainties over keypoints; for example, $\sigma_{(x,y)}[i]$ can capture the fact that a particular pixel matches well to both the left and right ankle, as visualized in Fig. 3.3, while a regressor may "regress" to the mean of the two surface coordinates.

3.3.3 Learning Embeddings and Articulated Shapes

Next we will introduce the loss functions that enable learning both embeddings and articulated shapes from monocular videos without a pre-defined shape template or annotated correspondence. To learn non-degenerate embeddings and overcome the local optima issue in differertiable renderers, we carefully construct a 3D matching loss and a 2D cycle loss.

3D match loss. Arguably, the simplest loss to learn embeddings is to minimize the

difference between the rendered surface features and the observed pixel features:

$$L_{\text{feature-consistency}} = \sum_{x,y} \Big(1 - \cos\big(\mathcal{R}(\mathbf{F_S})[x,y], \mathbf{F_I}[x,y]\big)\Big), \tag{3.8}$$

where $\cos(\cdot)$ denotes the inner product between two normalized vectors, and $\mathcal{R}(\mathbf{F_S})$ is the differentiably rendered surface descriptors. However, the feature consistency loss admits a trivial solution, where all pixel and surface features are the same constant (yielding zero error). To address this, we introduce a 3D matching loss that ensures pixel embeddings only match to surface embeddings rendered at the pixel location:

$$L_{\text{matching}} = \sum_{x,y} \Big\| \mathcal{R}(\bar{\mathbf{V}})[x,y] - \hat{\mathbf{S}}[x,y] \Big\|_2 \tag{3.9}$$

where $\mathcal{R}(\bar{\mathbf{V}})$ is the rendered 3D surface location and $\hat{\mathbf{S}}[x,y]$ is the estimated pixel-to-surface mapping from Eq. (3.6), computed through sampling and computing the softmax distributions $\sigma[i]$ over surface points [112]. To minimize the loss, the embeddings of surface points that do not project to (x,y) will be pulled away from the pixel embedding of (x,y) in a contrastive way [77].

2D cycle loss. The match loss aims to learn pixel-surface embeddings that are consistent over video frames and discriminative over difference surface locations. However for articulation optimization, the match loss suffers from bad local optima issue similar to other losses based on differentiable rendering [140]. For instance, when the rendering of a body part is outside the ground-truth object silhouette, a gradient update of articulation parameters would likely not incur a lower loss.

To guide articulated 3D shape learning using the learned pixel-surface embeddings, we further define a cycle-based re-projection loss, inspired by prior approaches in 3D model fitting with keypoints [27] and canonical surface mappings [122]. Given an input image, we establish a 2D-3D mapping by extracting a pixel embedding and matching it to surface embedding. Then, we compute the expected surface coordinate $\hat{\mathbf{S}}[x,y]$ at every pixel using Eq. (3.6), and ensure the differentiably rendered canonical surface coordinate lands back on the original pixel coordinate (x,y),

$$L_{\text{reproj}} = \sum_{x,y} \Big\| \mathcal{R}(\hat{\mathbf{S}}[x,y]) - (x,y) \Big\|_2 . \tag{3.10}$$

Reconstruction loss. Finally, we make use of reconstruction losses to ensure that generated images, masks, and flows match their estimated counterparts:

$$L_{\text{recon}} = \beta_1 ||\hat{S}_t^i - S_t||_2^2 + \beta_2 ||\hat{I}_t^i - I_t||_2^2 + \beta_3 \sigma_t ||\hat{u}_t^i - u_t||_2 + \beta_4 \text{pdist}(\hat{I}_t, I_t) \tag{3.11}$$

where $\{\beta_1, \cdots, \beta_4\}$ are weights empirically chosen, σ_t is the normalized confidence map for flow measurement, and $\text{pdist}(\cdot, \cdot)$ is the perceptual distance [325] measured by an ImageNet-pretrained AlexNet. The reconstruction losses ensure the match between rendered and observed optical flow, texture and silhouette images.

Regularization. To avoid degenerate shapes, we use mesh Laplacian regularization [108, 306] to enforce the recovered shape to be smooth, and as-rigid-as-possible (ARAP) regularization to enforce the deformation to be locally rigid [259]. Different from prior work that only preserves the length of edges after articulation, we encourage both the area and length of faces to be the same after articulation. The area preserving term is defined as

$$L_{\text{ARAP-area}} = \sum_{i=1}^{|E|} \sum_{j \in N_i} \left| \left| \mathbf{E_i^t} \times \mathbf{E_j^t} \right| - \left| \mathbf{E_i^{t+1}} \times \mathbf{E_j^{t+1}} \right| \right|, \tag{3.12}$$

where $|E|$ is the number of edges and N_i the indices of neighbouring edges.

3.3.4 Representing Surface Properties with MLPs

By extending surface embedding MLPs with additional dimensions, we can model other surface properties including textures and even surface-based geometric deformations. Compared to explicitly defined textures, such continuous implicit representations have the capacity to encode arbitrary amount of details and are empirically easier to optimize.

Surface appearance. The appearance of the object is represented by a coordinate-MLP queried at points on the canonical mesh surface. To handle view-dependent appearance (such as shadow and lighting), we further concatenate the Fourier features of the (X, Y, Z) coordinates with a frame appearance code, as the input to the texture MLP,

$$\mathbf{C_{i,t}} = \phi_{tex}(\mathcal{F}(\bar{\mathbf{V}}_i), \omega_t) \in \mathbb{R}^3, \tag{3.13}$$

where $\bar{\mathbf{V}}_i$ is the i-th canonical mesh vertex, which is passed through a Fourier encoder $\mathcal{F}(\cdot)$ as used in NeRF [165], and concatenated with ω_t, a 64-dimensional frame appearance code associated each image frame t, predicted from a ResNet-18, as $\omega_t = \psi_{tex}(I_t)$.

Instance shape deformation fields. To deal with videos of multiple instances of the same category, as experiments in Sec. 3.4.3, we model shape variations across instances by a continuous surface deformation field defined on the canonical surface. Similar to the surface texture, we represent the surface deformation field by a shape MLP,

$$\mathbf{V_{i,k}} = \bar{\mathbf{V}}_\mathbf{i} + \phi_{shape}(\mathcal{F}(\bar{\mathbf{V}}_i), \alpha_k) \in \mathbb{R}^3, \tag{3.14}$$

where $\mathbf{V_k}$ is the rest shape of instance k and α_k is a video-specific 64-dimensional shape code that is randomly initialized and optimized together with the shape MLP.

3.4 Experiments

We evaluate ViSER in three different scenarios where objects are highly articulating, making it challenging to reconstruct and estimate long-range correspondences. First, we consider long human videos with loose clothing and unusual poses. Next, we evaluate on videos of articulated animals for which accurate shape templates are missing. Finally, we analyze a multi-video variant of ViSER that learns a single model from multiple videos of the same category. All scenarios require jointly establishing long-range correspondences and reconstructing articulated 3D shapes at the same time.

Optimization details We use the AdamW [146] optimizer with a batch of 4 consecutive image pairs. We reconstruct a long video sequence in an incremental manner similar to classic SfM. First, we use an initial set of around 20 consecutive frames to initialize the shape and pixel surface embeddings. The initial set is selected such that the viewpoint coverage is large enough. Then we gradually add in new frames. When a new frame is added, we first apply the 2D cycle loss L_{reproj} to optimize its articulations, and then jointly optimize all frames with all losses. Empirically, simultaneously optimizing all the video frames produces unstable results of root body poses (or equivalently camera poses).

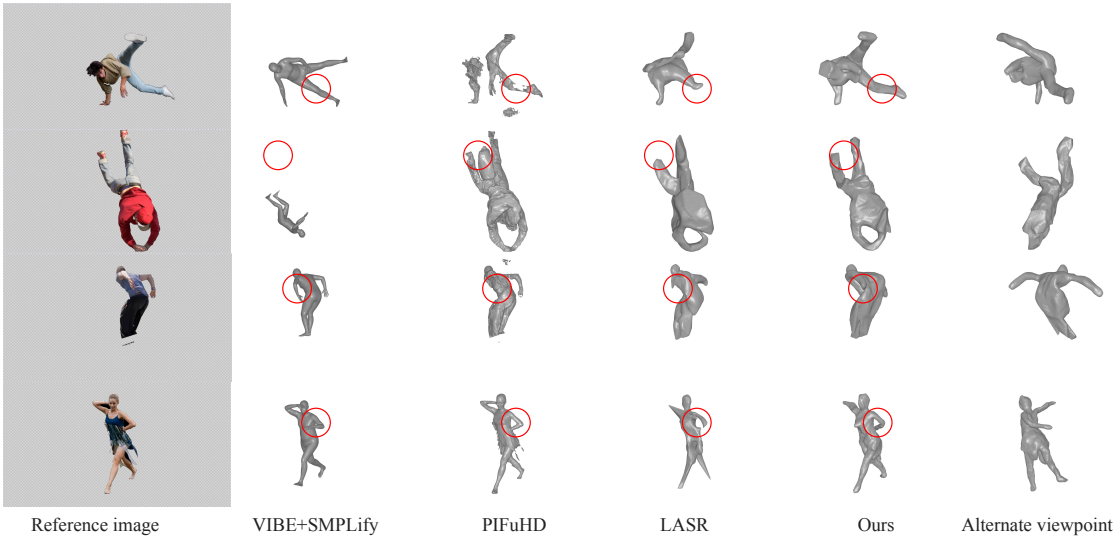

Figure 3.4: Qualitative comparisons for athletic video articulated shape reconstruction. Compared to methods that uses shape and pose priors (VIBE+SMPLify and PiFUHD), our method achieves comparable performance for common appearance and poses, and does much better on unusual poses such as break-dancers.

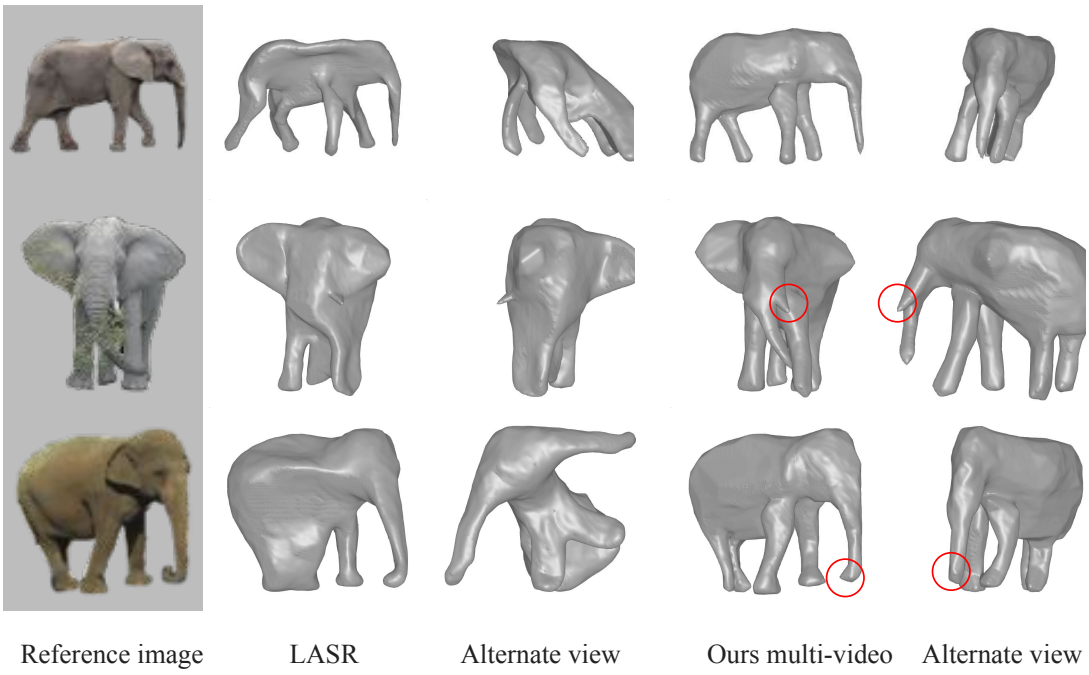

Figure 3.5: Qualitative comparisons for elephant shape reconstruction from multiple videos. Notice that ViSER is able to take advantage of multiple videos to improve the category-level shape reconstruction but also reconstruct instance-specific details (as shown in red circles).

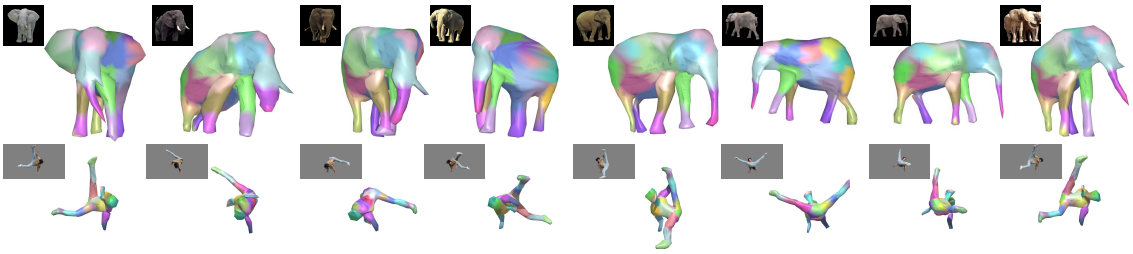

Figure 3.6: Part segmentation results. Colors are determined by hard-assigning vertices to the closest rigid bones.

Table 3.1: 2D Keypoint transfer accuracy on athletic videos. Methods with * use keypoint annotations to train. Best results are in bold.

Method	break-1	break-2	dance	parkour	ballet-1	ballet-2	ballet-3	Ave.
*DensePose CSE [170]	56.0	13.2	77.2	**85.9**	45.6	49.0	64.5	55.9
*VIBE+SMPLify [117]	37.1	8.2	70.4	83.8	**55.4**	53.0	**78.8**	55.2
LASR [306]	29.1	18.1	56.6	49.8	44.5	47.4	48.6	42.0
ViSER (Ours)	**70.5**	**22.5**	**80.7**	62.9	52.7	**56.1**	59.9	**57.9**

Table 3.2: 2D Keypoint transfer accuracy on elephant videos.

Method	inner	across
CSE [170]	55.7	52.2
Flow-VCN [301]	51.1	41.2
LASR [306]	57.8	-
ViSER (Ours)	**80.4**	**68.9**

Table 3.3: 2D Keypoint transfer accuracy on BADJA dataset. Best results are in bold.

Method	camel	dog	cows	horse	bear	Ave.
CSE [170]	48.8	38.6	63.8	60.2	76.6	57.6
Flow-VCN [301]	47.9	25.7	60.7	14.4	63.8	42.5
N-NRSfM [230]	67.8	17.9	70.0	8.7	60.2	44.9
LASR [306]	**81.9**	65.8	**83.7**	49.3	85.1	73.2
ViSER (Ours)	80.1	**73.8**	82.9	**76.3**	**87.3**	**80.1**

3.4.1 Athletic Video Reconstruction

Dataset. To evaluate ViSER on long-videos, we construct an athletic video dataset that is challenging due to loose clothing and unusual body poses. It consists of four videos from DAVIS [187] and three ballet videos. All videos are segmented and manually annotated with keypoints following the MSCOCO format [135]. We only use keypoint annotations for evaluation purposes.

Metrics. Due to the lack of ground-truth 3D data for challenging athletic human

videos, we use 2D keypoint transfer as a proxy metric [24, 330]. Given any two frames from a video, the goal is to transfer an annotated 2D keypoint from one frame to another. The accuracy is measured by percentage of correctly transferred keypoints over all T(T-1) pairs of frames in a T-frame video. A transferred keypoint is marked as correct when its distance to the ground-truth annotation is lower than $d_{th} = 0.2\sqrt{|S|}$, where $|S|$ is the area of the ground-truth silhouette [24]. In general, a more accurate reconstruction leads to a higher transfer accuracy.

Baselines. To compare with template-based approaches for video human reconstruction, we use VIBE with SMPLify temporal smoothing [117]. To compare with template-free methods, we use LASR [306], which also reconstructs articulated shapes using the same input setting as ours. To transfer keypoints from a reference frame to a target frame, we back-project the annotated keypoint in the reference frame to the canonical surface, and then project the intersected 3D point to the target frame. We also compare against Densepose CSE [170], which produces dense pixel-to-surface correspondences for a given category, but does not produce 3D reconstructions. To transfer keypoints for Densepose CSE, we compute pixelwise surface mappings for both frames and find the best matching w.r.t.geodesic distance on the surface. We further qualitatively compare against a state-of-the-art human reconstruction method, PiFUHD [214] in Fig. 3.4, which only produces reconstruction, but not correspondence.

Results. Fig. 3.4 shows visual reconstruction results on sample videos and for different techniques. ViSER estimates reconstructions that are more faithful to the input than the baselines, especially when the humans have unusual poses like in the first two rows. The accurate long-range correspondence enables ViSER to reconstruct finer details than LASR that does not explicitly try to estimate long-range correspondences. We summarize quantitative comparisons in Tab. 3.1. There is a moderate performance gap between ViSER and template-based methods when the input fits the latter, such as parkour with tight clothing and usual pose. Note that the supervised Densepose CSE and OpenPose methods fail on breakdance videos due to the novel pose, and also do not work well on ballet dancers due to loose clothing. As a result, template-based approaches that rely on accurate pose recognition, such as VIBE [117] fails. In contrast. our method does not suffer from such poor out-of-distribution generalization. By establishing long-range correspondences, ViSER

achieves higher keypoint transfer accuracy and better 3D reconstruction than LASR.

3.4.2 Reconstructing Animals from a Video

We use BADJA [24] to evaluate ViSER on animal videos including camel, cow, dog, bear and horse. Similar to the athletic human video dataset, we compare against template-free methods such as LASR and neural-dense-NRSfM (N-NRSfM) [230]. Similar to LASR and our setup, N-NRSfM learns a video-specific model for object shape, deformation and camera parameters from multi-frame optical flow estimations [61]. We further report performance comparison with dense correspondence methods such as CSE and an optical flow method, VCN. We use the CSE model trained on corresponding animal categories (except that we use the horse model for camel), and the "robust" model of VCN [2], which is the input to our method. As shown in Tab. 3.3, ViSER achieves better or similar accuracy on all five animal videos compared to LASR and N-NRSfM. While the input optical flow is not robust at estimating long-range correspondences, our method integrates local optical flow to a dense long-range correspondences via a canonical shape, and achieves much better keypoint transfer accuracy. Note that CSE performs well for categories it has been trained on, such as cow, horse and bear, but performs poorly on novel animal categories, such as camel and a novel breed of dog.

3.4.3 Multi-video Shape and Correspondence

We curate a set of seven videos of different elephants from YTVOS [296] for multi-video shape and correspondence recovery. The annotations will be released for further research. We treat multiple videos as a single long video with strong appearance changes and shape variations. In the multi-video setup, We evaluate keypoint transfer accuracy on both the same instance (with video frames) and over different instances (across video frames), as denoted by "inner" and "across". Quantitative results in Tab. 3.2 shows that ViSER is more accurate than the baseline methods in both cross-video keypoint transfer and inner-video keypoint transfer by a large margin, without using any keypoint annotations or pre-defined shape templates. Fig. 3.5 show visual result comparisons. While LASR recovers the visible surfaces in a video, it cannot infer the invisible parts. In contrast, our method is able to take advantage of multiple

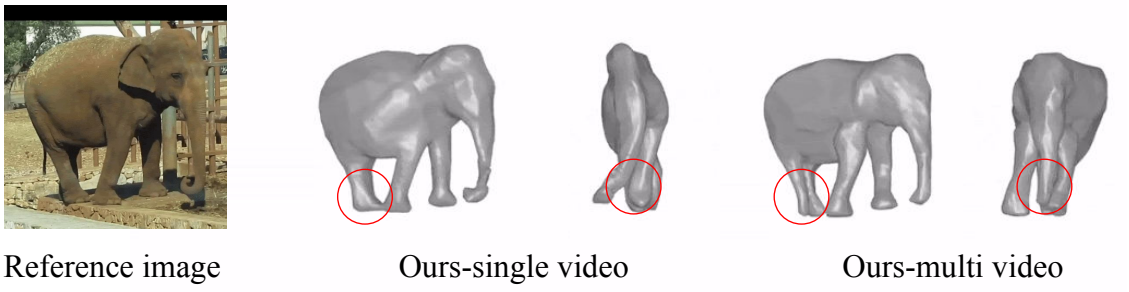

Figure 3.7: Comparison between single video ViSER and multi-video ViSER in terms of reconstructing YTVOS elephants. We find using multiple videos helps reconstructing the body parts that may be occluded in a single video. While single-video ViSER reconstructs a flattened shape and misses the hidden rear leg of the elephant, multiple video ViSER reconstructs a more plausible shape and recovers both the two rear limbs.

videos from the same category and produce a much better shape reconstruction. Note that LASR cannot handle multiple videos as it requires optical flow computed between every adjacent frame pairs. ViSER, on the other hand, also uses correspondences via estimated 3D shape, thereby allowing the use of multiple videos even when the optical flow is missing across videos.

Benefit of Using Multiple Videos. To examine the benefits of using multiple videos, we further compare multi-video ViSER with single-video ViSER, as shown in Fig. 3.7. We find using multiple videos helps reconstructing the body parts that may be occluded in a single video.

3.4.4 Part Discovery and Ablations

Part discovery. ViSER can discover detailed 3D part segmentation without any manual annotation, as shown in Fig. 3.6. After training either on a collection of videos or a long video, ViSER can segment the 3D shape into meaningful parts, such as the trunk of the elephants and the feet of the dancer.

Ablation study. We perform an ablation study on break-1 and elephants, as shown in Tab. 3.4. Without the contrastive matching loss, the pixel-surface embedding converges to a trivial solution with a significant decrease of accuracy. Removing the re-projection loss leads to much lower keypoint transfer (KPT) accuracy on the

Table 3.4: Ablation study on keypoint transfer. Best results are in bold.

Method	break-1	elephants-inner	elephants-cross
Full	**70.5**	**80.4**	**68.9**
w/o matching loss $L_{matching}$, Eq. (3.9)	36.2	51.3	42.6
w/o reprojection loss L_{reproj}, Eq. (3.10)	38.3	80.1	62.5
CSM regression [122]	47.1	77.4	63.3

breakdance-1 sequence and cross-video KPT accuracy on the elephant videos. Likely the surface reprojection loss plays an important role in learning correct articulation that follows the bottom-up dense keypoint predictions. This may effectively avoid the local minimum issue for the differentiable rendering optimization. Finally, replacing the pixel-surface embedding with direct CSM regression [122] does not reason about distribution of possible matches and results in worse performance.

Chapter 4

Building Animatable Models from Many Casual Videos

4.1 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 [4, 232, 233]. 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 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 or a pet over the span of several months or years. In this case, we wish our system to gather 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

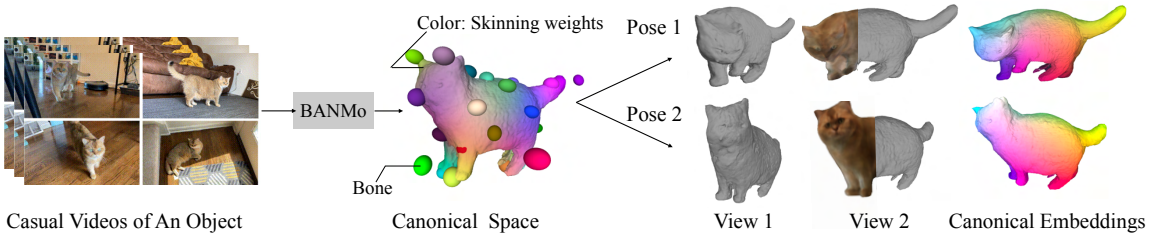

Figure 4.1: Given multiple casual videos capturing a deformable object, BANMo reconstructs an animatable 3D model, including an implicit canonical 3D shape, appearance, skinning weights, and time-varying articulations, without pre-defined shape templates or registered cameras. **Left**: Input videos; **Middle**: 3D shape, bones, and skinning weights (visualized as surface colors) in the canonical space; **Right**: Posed reconstruction at each time instance with color and canonical embeddings (correspondences are shown as the same colors).

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 model, similar to [42, 215, 306, 307], making the output *animatable* by manipulating bone transformations.

As shown in NRSfM [29], reconstructing a freely moving non-rigid object from monocular video is challenging, where epipolar constraints are not directly applicable. We address three core challenges in BANMo: (1) how to represent 3D geometry and appearance of the target in a canonical space; (2) how to deform 3D points between canonical space and individual time instances; (3) how to find pixel or part correspondences over videos given different viewpoint, lighting, background, and object deformations.

Concretely, we utilize neural implicit functions [165] 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 [306, 307]. 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 variations and deformations with unknown camera parameters, compared to dynamic NeRF approaches [42, 132, 179, 192]. To find correspondences, we present a module that performs dense matching between pixels and an implicit feature volume. Finally, for

robust and efficient optimization over a large number of video frames, we pre-train a pose network for human and quadruped animals to provide initial camera orientations. In a nutshell, BANMo presents a way to merge the recent non-rigid object reconstruction approaches [306, 307] in a dynamic NeRF framework [42, 132, 179, 192], 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 [307], by taking better advantage of the large number of frames in multiple videos.

4.2 Related work

Human and animal body models. A large body of work in 3D human and animal reconstruction uses parametric shape models [144, 182, 267, 287, 333, 334], which are built from registered 3D scans of human or animals, and serve to recover 3D shapes given a single image or video at test time [9, 25, 117, 117, 335]. Although parametric body models achieve great success in reconstructing human with large amounts of ground-truth 3D data, 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 images or videos. 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 [73, 108, 118, 130, 131, 282, 316]. Such methods often rely on a coarse shape template [122, 259, 327], and are not able to recover fine-grained details or large deformations. Recently, HDNet [94] uses social media videos to learn depth estimators for clothed human.

Category-agnostic video shape reconstruction. Non-rigid structure from motion (NRSfM) methods [29, 74, 121, 123, 230] 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 [216, 243], they do not work well for videos in the wild. Recent efforts such as LASR and ViSER [306, 307] 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 [99, 133, 157, 160, 165, 276]. To extend NeRF to dynamic scenes, recent works introduce additional functions to deform observed points to a canonical space or over time [132, 179, 180, 192, 257, 271]. 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 [138, 175, 183, 184, 238] produce pose-controllable NeRFs, but they rely on a human body model, or synchronized multi-view video inputs.

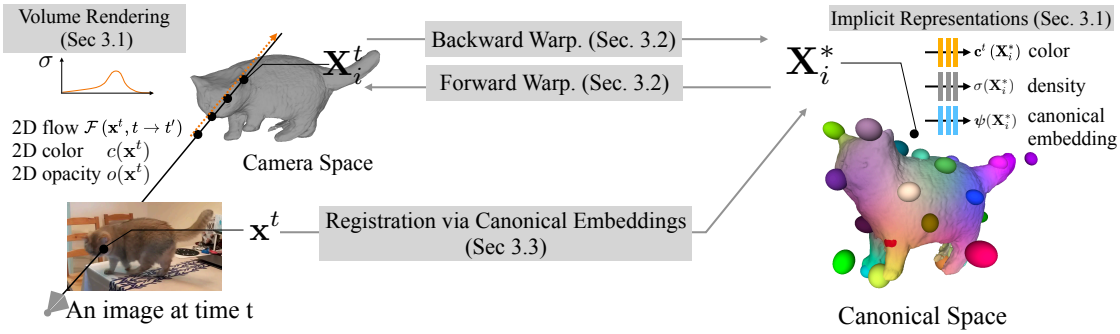

Figure 4.2: **Method overview.** BANMo optimizes a set of shape and deformation parameters (Sec. 4.3.1) that describe the video observations in pixel colors, silhouettes, optical flow, and higher-order features descriptors, based on a differentiable volume rendering framework. BANMo uses a neural blend skinning model (Sec. 4.3.2) 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. 4.3.3).

4.3 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 pixel feature embeddings. We refer readers to the overview in Fig. 4.2 and the list of notations in supplement.

4.3.1 Shape, Appearance, and Warping Model

We first represent shape and appearance of deformable objects in a canonical time-invariant *rest pose* space.

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) [165]. A 3D point $\mathbf{X}^* \in \mathbb{R}^3$ in the canonical space is associated with three properties: color $\mathbf{c} \in \mathbb{R}^3$, density $\sigma \in [0, 1]$, and a canonical embedding $\boldsymbol{\psi} \in \mathbb{R}^{16}$. These properties are predicted by the Multilayer Perceptron (MLP) networks:

$$\mathbf{c}^t = \mathbf{MLP}_{\mathbf{c}}(\mathbf{X}^*, \mathbf{v}^t, \boldsymbol{\omega}_e^t), \tag{4.1}$$

$$\sigma = \Gamma_\beta(\mathbf{MLP}_{\text{SDF}}(\mathbf{X}^*)), \tag{4.2}$$

$$\boldsymbol{\psi} = \mathbf{MLP}_{\boldsymbol{\psi}}(\mathbf{X}^*). \tag{4.3}$$

As in NeRF, color $\mathbf{c}^t$ also 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 captures environment illumination conditions [157].

The shape is given by $\mathbf{MLP}_{\text{SDF}}$, computing the Signed-Distance Function (SDF) of a point to the surface. To convert SDF to density σ for volume rendering, we use the cumulative of a unimodal distributuion with zero mean and β scale, denoted as $\Gamma_\beta(x)$. β is a learnable parameter that controls the solidness of the object, approaching zero for solid objects [274, 314]. Prior works [274, 314] have explored the cumulative of Logistic and Laplace distribution respectively and we follow VolSDF [314] to use the accumulative of Laplace distribution. Compared with ReLU of Softplus activations used in NeRF, it provides a principled way of extracting surface as the zero level-set of the SDF.

Finally, the $\mathbf{MLP}_{\boldsymbol{\psi}}$ network maps 3D points to a canonical feature embedding $\boldsymbol{\psi}$ that can be matched by pixels from different viewpoints and lighting conditions,

enabling long-range correspondence across videos. This feature can be interpreted as a variant of Continuous Surface Embeddings (CSE) [170] but defined volumetrically.

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 [179] and Neural Scene Flow Fields (NSFF) [132] learn deformations under the assumptions of known camera poses and small object deformation. As detailed in Sec. 4.3.2 and Sec. 4.3.4, we do not make such assumptions; instead, we adopt a neural blend-skinning model that can handle large deformations without a pre-defined skeleton model.

Volume rendering. To render images, we use volume rendering in NeRF [165], but warp the 3D ray to account for deformation [179]. 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 $\mathbf{x}^t$. As the color and density are defined in the canonical space, we *pull back* the sampled points to the canonical space using $\mathbf{X}_i^* = \mathcal{W}^{t,\leftarrow}\left(\mathbf{X}_i^t\right)$. The color $\mathbf{c}$ and the opacity $o \in [0, 1]$ are then given by:

$$\mathbf{c}(\mathbf{x}^t) = \sum_{i=1}^{N} \tau_i \mathbf{c}_i^t, \quad o(\mathbf{x}^t) = \sum_{i=1}^{N} \tau_i, \tag{4.4}$$

where N is the number of samples and τ_i is the free-flight probability that a photon travels between the camera center and the i-th sample, as 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 sample and the next. Color $\mathbf{c}_i$ and density σ_i are computed by Eq. 4.1-4.2. We further compute the expected surface intersection:

$$\mathbf{X}^*(\mathbf{x}^t) = \sum_{i=1}^{N} \tau_i \mathbf{X}_i^*. \tag{4.5}$$

To render 2D flow, we *push forward* the warped ray points to another time t' via

forward warping $\mathcal{W}^{t',\rightarrow}$to find its expected 2D re-projection:

$$\mathbf{x}^{t'} = \sum_{i=1}^{N} \tau_i \Pi^{t'} \left(\mathcal{W}^{t',\rightarrow} \left(\mathbf{X}_i^* \right) \right), \tag{4.6}$$

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:

$$\mathcal{F} \left(\mathbf{x}^t, t \rightarrow t' \right) = \mathbf{x}^{t'} - \mathbf{x}^t. \tag{4.7}$$

4.3.2 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 [42]. Our formulation represents 3D warps as compositions of neural-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 3D coordinate of bones. Let $\mathbf{G}^t \in SE(3)$ be a transformation of the object root body from canonical space to time t, and $\mathbf{J}_b^t \in SE(3)$ be a rigid transformation that moves the b-th bone from its rest configuration to deformed state t, then we have

$$\mathbf{X}^t = \mathcal{W}^{t,\rightarrow}(\mathbf{X}^*) = \mathbf{G}^t \mathbf{J}^{t,\rightarrow} \mathbf{X}^*, \tag{4.8}$$

$$\mathbf{X}^* = \mathcal{W}^{t,\leftarrow}(\mathbf{X}^t) = \mathbf{J}^{t,\leftarrow} (\mathbf{G}^t)^{-1} \mathbf{X}^t, \tag{4.9}$$

where $\mathbf{J}^{t,\rightarrow}$ and $\mathbf{J}^{t,\leftarrow}$ are weighted averages of B rigid transformations $\{\mathbf{J}_b^t\}_{b\in\{1,\ldots,B\}}$ that move the bones between rest configurations and time t configurations. Following linear blend skinning deformation [93], we have

$$\mathbf{J}^{t,\rightarrow} = \sum_{b=1}^{B} \mathbf{W}_b^{t,\rightarrow} \mathbf{J}_b^t, \quad \mathbf{J}^{t,\leftarrow} = \sum_{b=1}^{B} \mathbf{W}_b^{t,\leftarrow} (\mathbf{J}_b^t)^{-1}, \tag{4.10}$$

where $\mathbf{W}_b^{t,\rightarrow}$ and $\mathbf{W}_b^{t,\leftarrow}$ represent pose-dependent skinning weights that assigns point $\mathbf{X}^*$ and $\mathbf{X}^t$ to the b-th bone.

Pose representation. We represent poses with angle-axis rotations and 3D translations, regressed from MLPs:

$$\mathbf{G}^t = \mathbf{MLP}_{\mathbf{G}}(\boldsymbol{\omega}_r^t), \quad \mathbf{J}_b^t = \mathbf{MLP}_{\mathbf{J}}(\boldsymbol{\omega}_b^t) \tag{4.11}$$

where $\boldsymbol{\omega}_r^t$ and $\boldsymbol{\omega}_b^t$ are 128-dimensional latent codes for root pose and body pose at frame t respectively. Compared with directly optimizing SE(3) poses, we find such over-parameterized representations converges better with stochastic first-order gradient methods. Instead of treating pose codes as independent parameters learned per-frame, we represent each dimension of the latent code as linear combinations of sinusoidal basis functions:

$$\omega_t^b = \mathbf{A}_i \mathcal{F}(t) \tag{4.12}$$

where $\mathcal{F}(\cdot)$ is a 1D basis of sines and cosines with linearly-increasing frequencies at log-scale [247], and we learn separate weight matrices $\mathbf{A}_{i\in\{1\ldots,M\}}$ for each video. The frame index t is normalized by the maximum video length $\max_{i=1}^{M} |T_i|$. Using the temporal Fourier basis stabilizes the optimization and produces more smooth deformations.

Skinning weights. Similar to SCANimate [215], we define a skinning weight function $\mathcal{S} : (\mathbf{X}, \boldsymbol{\omega}_b) \rightarrow \mathbf{W} \in \mathbb{R}^B$ that assigns $\mathbf{X}$ to bones given body pose code ω_b. During backward mapping, we apply $\mathcal{S}$ to time t points and pose codes $\boldsymbol{\omega}_b^t$ to compute backward skinning weights $\mathbf{W}^{t,\leftarrow}$. During forward mapping, we apply the same $\mathcal{S}$ to the canonical space points and rest pose code $\boldsymbol{\omega}_b^*$ to compute the forward skinning weights $\mathbf{W}^{t,\rightarrow}$.

Directly representing $\mathcal{S}$ as neural networks can be difficult to optimize. Therefore, we condition neural skinning weights on explicit 3D Gaussian ellipsoids that move along with the bone coordinates. Similar to LASR [306], the Gaussian skinning weights are determined by the Mahalanobis distance between $\mathbf{X}$ and the Gaussian ellipsoids:

$$\mathbf{W}_\sigma = (\mathbf{X} - \mathbf{C}_b)^T \mathbf{Q}_b (\mathbf{X} - \mathbf{C}_b), \tag{4.13}$$

where $\mathbf{C}_b \in \mathbb{R}^{B\times 3}$ are bone centers and $\mathbf{Q}_b = \mathbf{V}_b^T \mathbf{\Lambda}_b^0 \mathbf{V}_b$ are the precision matrices

composed by bone orientations $\mathbf{V}_b \in \mathbb{R}^{B\times3\times3}$ and diagonal scale matrices $\mathbf{\Lambda}_b^0 \in \mathbb{R}^{B\times3\times3}$. When computing backward skinning weights, bone centers and orientations are transformed as $\left(\mathbf{V}_b|\mathbf{C}_b\right) = \mathbf{J}_b\left(\mathbf{V}_b^0|\mathbf{C}_b^0\right)$, where $\mathbf{J}_b$ are bone transforms in Eq. 4.11. $\mathbf{\Lambda}_b^0$, $\mathbf{V}_b^0$ and $\mathbf{C}_b^0$ are learnable rest bone configurations.

To model the skinning weights for fine geometry, we find it helpful to add delta skinning weights after the coarse component is well-optimized. Delta skinning weights are represented as a coordinated-MLP, $\mathbf{W}_\Delta = \mathbf{MLP}_\Delta(\mathbf{X}, \boldsymbol{\omega}_b) \in \mathbb{R}^B$. The final skinning function is the sum of the coarse and fine components, normalized by a softmax function,

$$\mathbf{W} = \mathcal{S}(\mathbf{X}, \omega_b) = \sigma_{\text{softmax}}\left(\mathbf{W}_\sigma + \mathbf{W}_\Delta\right). \tag{4.14}$$

The Gaussian component regularizes the skinning weights to be spatially smooth and temporally consistent, and handles large deformations better than purely implicitly-defined ones. Furthermore, our formulation of the skinning weights are dependent on only pose status by construction, and therefore regularizes the space of skinning weights.

4.3.3 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 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.

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 $\boldsymbol{\psi}(\mathbf{X}^*) \in \mathbb{R}^{16}$ best matches the pixel feature embedding $\boldsymbol{\psi}_I^t(\mathbf{x}^t) \in \mathbb{R}^{16}$. The pixel embeddings $\boldsymbol{\psi}_I^t$ (of frame t) are computed by a CNN. Different from ViSER [307] that learns embeddings from scratch, we initialize pixel embeddings with CSE [170, 171] 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 $\boldsymbol{\psi}(\mathbf{X}^*) = \mathbf{MLP}_{\boldsymbol{\psi}}(\mathbf{X}^*)$ in Eq. 4.3. Intuitively, $\mathbf{MLP}_{\boldsymbol{\psi}}$ 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

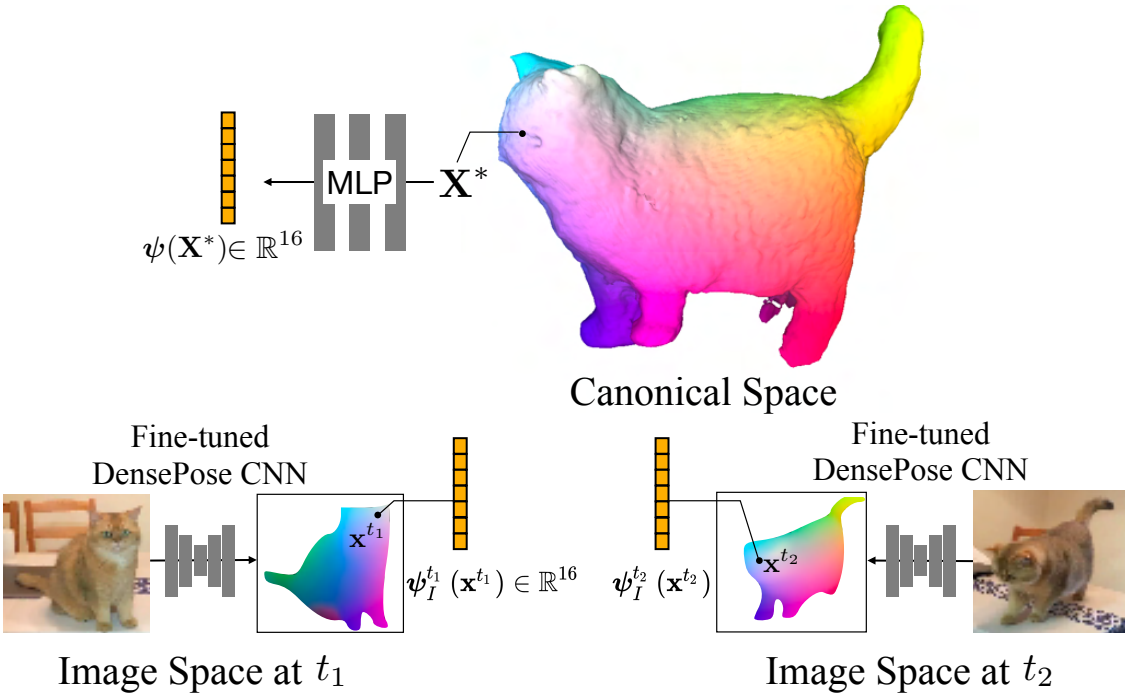

Figure 4.3: **Canonical Embeddings.** We jointly optimize an implicit function to produce canonical embeddings from 3D canonical points that match to the 2D DensePose CSE embeddings [170].

soft argmax descriptor matching [112, 149]:

$$\hat{\mathbf{X}}^*(\mathbf{x}^t) = \sum_{\mathbf{X} \in \mathbf{V}^*} \tilde{\mathbf{s}}^t(\mathbf{x}^t)\mathbf{X}, \tag{4.15}$$

where $\mathbf{V}^*$ are sampled points in a tightly-bounded canonical 3D grid, and $\tilde{\mathbf{s}}$ is a normalized distribution of feature matches over the 3D grid: $\tilde{\mathbf{s}}^t(\mathbf{x}^t) = \sigma_{\text{softmax}}\Big(\alpha_s \big\langle \boldsymbol{\psi}_I^t(\mathbf{x}^t), \boldsymbol{\psi}(\mathbf{X}) \big\rangle\Big)$, where α_s is a learnable scaling to control the peakness of the softmax function and $\langle ., . \rangle$ is the cosine similarity.

Self-supervised canonical embedding learning. As describe later in Eq. 4.17-4.18, 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. 4.4.3).

4.3.4 Optimization

Given multiple videos, we optimize all parameters described above, including MLPs, $\{\mathbf{MLP_c}, \mathbf{MLP}_{\text{SDF}}, \mathbf{MLP}_{\boldsymbol{\psi}}, \mathbf{MLP_G}, \mathbf{MLP_J}, \mathbf{MLP}_{\Delta}\}$, learnable codes $\{\boldsymbol{\omega}_e^t, \boldsymbol{\omega}_r^t, \boldsymbol{\omega}_b^t, \boldsymbol{\omega}_b^*\}$ and pixel embeddings $\boldsymbol{\psi}_I$.

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

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

Reconstruction losses are similar to those in existing differentiable rendering pipelines [165, 313]. Besides color loss $\mathcal{L}_{\text{rgb}}$ and silhouette loss $\mathcal{L}_{\text{sil}}$, we further compute flow reconstruction losses $\mathcal{L}_{\text{OF}}$ by comparing the rendered $\mathcal{F}$ defined in Eq. 4.7 with the observed 2D optical flow $\hat{\mathcal{F}}$ computed by an off-the-shelf flow network:

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

$$\mathcal{L}_{\text{OF}} = \sum_{\mathbf{x}^t, (t,t')} \left\| \mathcal{F}\left(\mathbf{x}^t, t \to t'\right) - \hat{\mathcal{F}}\left(\mathbf{x}^t, t \to t'\right) \right\|^2, \tag{4.16}$$

where $\hat{\mathbf{c}}$ and $\hat{\mathbf{s}}$ are observed color and silhouette. Additionally, we define feature matching losses to enforce 3D points predicted via canonical embedding $\hat{\mathbf{X}}^*(\mathbf{x}^t)$ (Eq. 4.15) to match the prediction from backward warping (Eq. 4.5):

$$\mathcal{L}_{\text{match}} = \sum_{\mathbf{x}^t} \left\| \hat{\mathbf{X}}^*(\mathbf{x}^t) - \mathbf{X}^*(\mathbf{x}^t) \right\|_2^2, \tag{4.17}$$

and a geometric cycle consistency loss [122, 307] that forces the image projection after forward warping of $\hat{\mathbf{X}}^*(\mathbf{x}^t)$ to land back on its original 2D coordinates:

$$\mathcal{L}_{\text{2D-cyc}} = \sum_{\mathbf{x}^t} \left\| \Pi^t \left(\mathcal{W}^{t,\to}(\hat{\mathbf{X}}^*(\mathbf{x}^t)) \right) - \mathbf{x}^t \right\|_2^2. \tag{4.18}$$

Similar to NSFF [132], 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:

$$\mathcal{L}_{\text{3D-cyc}} = \sum_i \tau_i \left\| \mathcal{W}^{t,\rightarrow}\left(\mathcal{W}^{t,\leftarrow}(\mathbf{X}_i^t)\right) - \mathbf{X}_i^t \right\|_2^2, \tag{4.19}$$

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. To improve the robustness of optimization, we consider the following initialization strategy for root body poses.

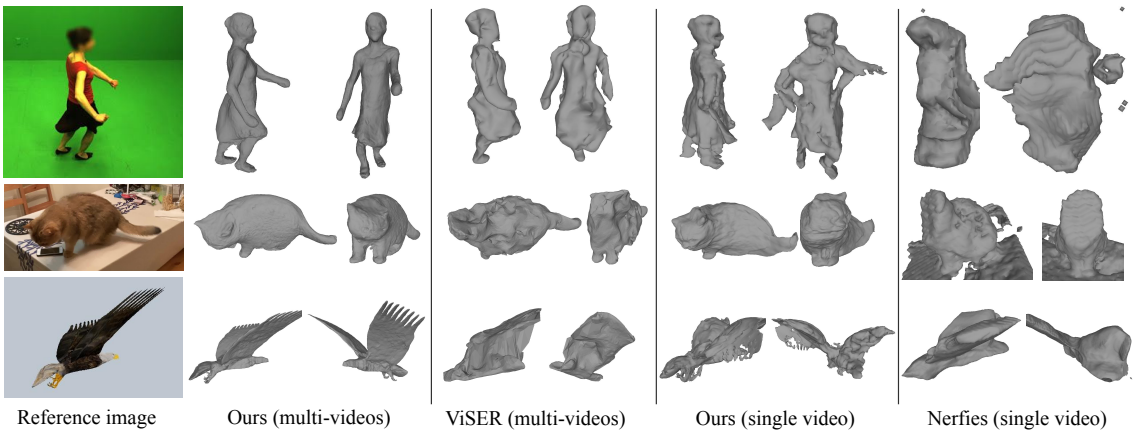

Figure 4.4: **Qualitative comparison of our method with prior art [179, 307].** From top to bottom: AMA's `samba`, `casual-cat`, `eagle`.

Root pose initialization. We provide a rough per-frame initialization of root poses ($\mathbf{G}^t$ in Eq. 4.8), similar to NeRS [322]. Specifically, we train a separate network PoseNet , which is applied to every test video frame. Similar to DenseRaC[299], PoseNet takes a DensePose CSE [170] 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 [170] from an RGB image $\boldsymbol{I}_t$. 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:

$$\mathbf{G}^t = \mathbf{MLP}_{\mathbf{G}}(\boldsymbol{\omega}_r^t)\mathbf{G}_0^t. \tag{4.20}$$

Active sampling over pixels Inspired by iMAP [239], our ray sampling strategy follows an easy-to-hard curriculum. At the early iterations, we randomly sample a batch of N^p pixels for volume rendering and compute reconstruction losses. At the same time, we optimize a 8-layer MLP function to represent the uncertainty over the image coordinate and frame index: $\hat{\mathbf{U}}(x, y, t) = \mathbf{MLP}_{\mathbf{U}}(x, y, t)$. The uncertainty MLP is optimized by comparing against the color reconstruction errors in the current forward step:

$$\mathcal{L}_{\mathbf{U}} = \sum_{\mathbf{x},t} \left\| \mathcal{L}_{\text{rgb}}(\mathbf{x}^t) - \hat{\mathbf{U}}(\mathbf{x}^t) \right\|. \tag{4.21}$$

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 some optimization steps, we replace half of the samples with *active* samples from pixels with high uncertainties. To do so, we randomly sample $N^{a'} = 24576$ pixels, and evaluate their uncertainties by passing their coordinates and frame indices to $\mathbf{MLP}_{\mathbf{U}}$. Active samples dramatically improves reconstruction fidelity, as shown in Fig. 4.15.

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 $[x_{\min}, x_{\max}]$, $[y_{\min}, y_{\max}]$, and $[z_{\min}, z_{\max}]$. 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 $\boldsymbol{L}$ 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 $d_i^t = (\mathbf{\Pi}^t\mathbf{G}^t\mathbf{X}_i^*)_2$. 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

Table 4.1: Table of hyper-parameters.

Name	Value	Description
B	25	Number of bones
N	128	Sampled points per ray
N^p	6144	Sampled rays per batch
(H, W)	(512,512)	Resolution of observed images

points.

Hyper-parameters. We use 1cycle learning rate scheduler, which warms-up with a low learning rate to the maximum, and anneals the learning rate to a final learning rate. We apply $lr_{init} = 2e-5$, $lr_{max} = 5e-4$, $lr_{final} = 1e-4$. We refer readers to a complete list of hyper-parameters in Tab. 4.1.

Multi-stage optimization The final optimization takes three stages, where the optimizable parameters and the loss functions used are different. The first stage uses all the losses and updates all the parameters described in the paper. Typically, the first stage already produces 3D reconstructions with good shape and deformation. The goal of the stage 2 is to improve the articulations (e.g., to correctly articulate the crossing legs for `cat-pikachiu`) with coordinate gradient descent, where we turn off the reconstruction losses and only use the 2D cycle consistency loss to update the articulation parameters while keeping shape parameters fixed. Finally, stage 3 improves the details of the geometry by active sampling and importance depth sampling while keeping the root body poses fixed.

4.3.5 Root Pose Initialization

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. 4.5.

Preliminary DensePose CSE [170, 171] trains pixel embeddings $\boldsymbol{\psi}_I$ and surface feature embeddings $\boldsymbol{\psi}$ for humans and quadruped animals using 2D keypoing annotations. It represents surface embeddings by a canonical surface with N vertices and vertex features $\boldsymbol{\psi} \in \mathbb{R}^{N\times 16}$. A SMPL mesh is used for humans, and a sheep mesh is

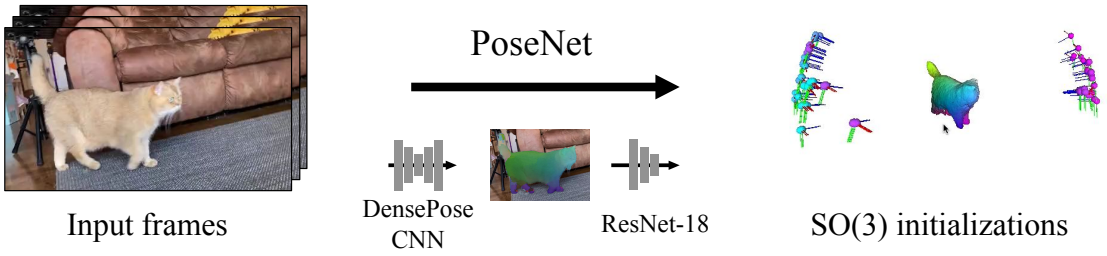

Figure 4.5: **Inference pipeline of PoseNet.** To initialize the optimization, we train a CNN PoseNet to predict root poses given a single image. PoseNet uses a DensePose-CNN to extract pixel features and decodes the pixel features into root pose predictions with a ResNet-18. We visualize the initial root poses on the right. Cyan color represents earlier time stamps and magenta color represent later timestamps.

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. 4.6. Specifically, we render surface features as feature images $\boldsymbol{\psi}_{\text{rnd}} \in \mathbb{R}^{112\times112\times16}$ given viewpoints $\mathbf{G}^* = (\mathbf{R}^*, \mathbf{T}^*)$ randomly generated on a unit sphere facing the origin. We apply occlusion augmentations [231] 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.

Loss and inference. We use the geodesic distance between the ground-truth and

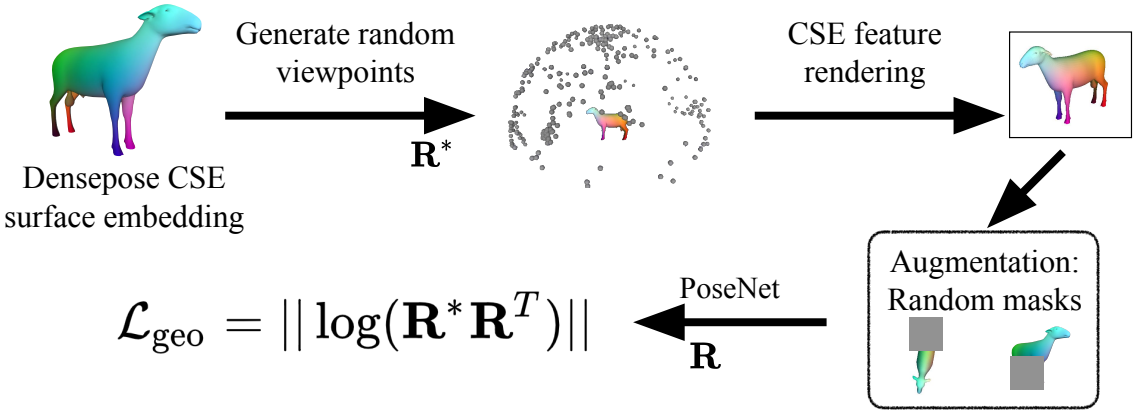

Figure 4.6: **Training pipeline of PoseNet.** To train PoseNet, we use DesePose CSE surface embeddings, which is pertained on 2D annotations of human and quadruped animals. We first generate random viewpoints on a sphere that faces the origin. Then we render surface embeddings as 16-channel images. We 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.

predicted rotations as a loss to update PoseNet,

$$\mathcal{L}_{\text{geo}} = || \log(\mathbf{R}^* \mathbf{R}^T) ||, \quad \mathbf{R} = \text{PoseNet}(\boldsymbol{\psi}_{\text{rnd}}), \tag{4.22}$$

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:

$$\mathbf{G}^t = \mathbf{MLP}_{\mathbf{G}}(\omega_r^t)\mathbf{G}_0^t. \tag{4.23}$$

4.4 Experiments

Implementation details. Our implementation of implicit shape and appearance models follows NeRF [165], 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 $\mathcal{W}^{t,\rightarrow}$.

Optimization details. We initialize $\mathbf{MLP}_{\text{SDF}}$ such that it approximates a unit sphere [313]. We use $B = 25$ rest bones, which are initialized with unit scale, identity orientation, and centers uniformly spaced on the initial rest surface. During optimization, we reinitialize the rest bones at $\{20\%, 67\%\}$ of total iterations and further encourage them to stay close to the surface with a Sinkhorn divergence loss [56]. In a batch, we sample $N^I = 512$ image pairs and sample $N^p = 6144$ pixels from all image pairs for rendering. The interval between image pairs is randomly chosen $\Delta T \in \{1, 2, 4, 8, 16, 32\}$. To stabilize optimization, we observe that N_I needs to roughly match the number of input frames. The reconstruction quality improves with more iterations and we find 36k iterations (15 hours on a V100 GPU) already produces high-fidelity details. Please find a list of hyper-parameters in supplement.

4.4.1 Dataset and Metrics

Qualitative: Casual videos dataset. We demonstrate BANMo's ability to reconstruct 3D models from casual videos of animals and humans. Object silhouette and optical flow (for computing reconstruction losses Eq. 4.16) are extracted by off-the-shelf models, PointRend and VCN-robust [116, 301]. Two special challenges arise from the casual nature of the video captures. First, each video collection contains around 1k images, an order of magnitudes larger those used in prior work [132, 165, 179, 307], which requires the method to handle reconstructions at a larger scale. Second, the dataset makes no control over camera movement or object movement. In particular, objects freely moves in a video and background changes across videos, posing challenges to standard SfM pipelines to estimate the object root poses. We show results on 11 videos (totaling 900 images) of a British shorthair cat denoted as `casual-cat` below. Please find other results in the project webpage.

Quantitative: AMA human dataset. Articulated Mesh Animation (AMA) dataset [265] contains multi-view videos captured by 8 synchronized cameras. It provides high-fidelity ground-truth meshes with clothing. We use 2 sets of videos of the same actor (`swing` and `samba`), totaling 2600 frames, as the input to optimization. We use the ground-truth object silhouettes. Time synchronization and camera extrinsics

Table 4.2: **Quantitative results on AMA and Animated Objects.** 3D Chamfer distance (cm, ↓) and F-score (%, ↑) averaged over all frames. The 3D models for `eagle` and `hands` are resized such that the longest edge of the axis-aligned object bounding box is 2m. * *with ground-truth root poses.* S: single-video results. All methods are assigned with the same initial root pose.

Method	AMA-swing		Eagle*		Hands*	
	CD	F@2%	CD	F@2%	CD	F@2%
Ours	**9.1**	**57.0**	**8.1**	**56.7**	**7.5**	**49.6**
ViSER	15.7	52.2	23.0	20.6	16.8	21.3
Ours^S	9.4	56.8	10.8	48.6	10.5	35.2
Nerfies^S	22.6	13.2	18.4	18.0	24.4	14.9

are *not* used.

Quantitative: Animated Objects dataset. 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 and object silhouettes to BANMo and baselines.

Metrics. We quantify the results using both Chamfer distances and F-scores. Chamfer distance computes the average distance between the ground-truth and the estimated surface points by finding the nearest neighbour matches, but it is sensitive to outliers. Therefore, we further report the F-score at distance thresholds $d = 2\%$ of the longest edge of the axis-aligned object bounding box [248]. To account for the unknown scale and global rigid motion, we pre-align the estimated shape to the ground-truth via Iterative Closest Point (ICP) up to a 3D similarity transformation.

4.4.2 Reconstruction Results

We show qualitative comparison in Fig. 4.4 and quantitative comparison in Tab. 4.2.

Baseline setup.

We compare with Nerfies and ViSER and summarize the differences in Tab. 4.3. Nerfies [179] is designed for a single continuously captured video, assuming object

Table 4.3: **Difference between Nerfies, ViSER, and BANMo.**

Method	shape	deformation	registration
Nerfies	implicit	dense SE(3)	photometric
ViSER	mesh	control points	self-supervised feature
BANMo	implicit	control points	pre-trained feature

root body pose can be compensated by background-SfM. In our setup, object moves and background SfM does not provide root poses for the object. When focused on the deformable object, SfM (such as COLMAP) failed to converge due to violation of rigidity, leading to very few successful registrations (18 over 900 images registered on `casual-cat`). To make a fair comparison, we provide Nerfies with rough initial root poses (obtained from our PoseNet, Sec. 4.3.4). After optimization, meshes are extracted by running marching cubes on a 256^3 grid. Another baseline, ViSER [307], directly optimizes object shape and poses using optical flow, silhouette, and color reconstruction losses. It does not assume category-level priors such as CSE features, and therefore applicable to generic object categories. However, ViSER's root pose estimation is sensitive to large deformation and a large number of input frames (more than 20). Since it produces worse results than our PoseNet, we provide ViSER the same root poses from our initialization pipeline.

Comparison with Nerfies. Nerfies optimizes SE(3) fields with photometric error, which fails at large motion and fails to register pixels across videos. In contrast, BANMo optimizes an articulated bones model using *featuremetric* consistency w.r.t. a pre-trained CSE feature embedding. As shown in Fig. 4.4, although single-video 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). Furthermore, as shown in Fig. 4.11, Nerfies is not able to leverage more videos to improve the reconstruction quality, while the reconstruction of BANMo improves given more videos. The results in Tab. 4.2 suggests BANMo produces more accurate geometry than Nerfies for all sequences.

Comparison with ViSER. As shown in Fig. 4.4, 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

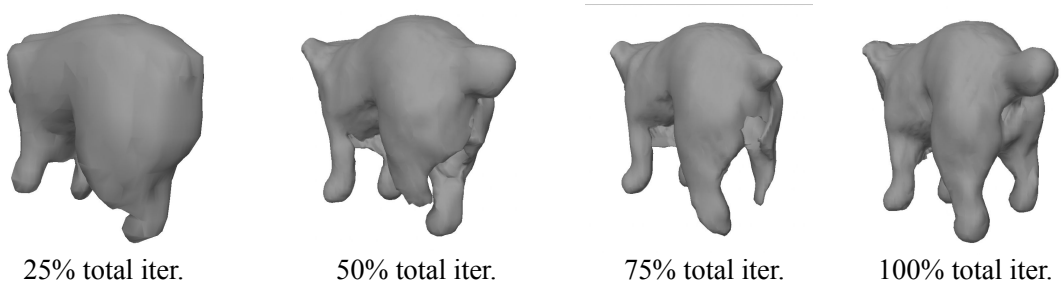

Figure 4.7: **Compliance to topology changes in optimization.** BANMo incorrectly reconstructs a single rear leg of the dog, but automatically corrects the topology with gradient updates.

rotation and leg switching are not recovered. In contrast, BANMo faithfully recovers these high-fidelity geometry and motion. We observed that the neural implicit volume representation is compliant to topology changes during gradient updates (see Fig. 4.7), and is therefore able to recover from bad local optima. In contrast, sub-optimal topology that happens during optimization, such as inverted faces, prevents ViSER to improve given more iterations. Compared to meshes with finite number of vertices, implicit shape representation maintains a continuous geometry, enabling us to recover detailed shape without additional cost in rendering high-res meshes.

4.4.3 Diagnostics

We ablate the importance of each component on AMA's `samba` and `swing` (325 frames in total). In Tab. 4.4, we report the results of ablations followed by analysis.

Root pose initialization. We show the effect of PoseNet for root pose initialization (Sec 4.3.4) in Fig. 4.8: without it, the root poses (or equivalently camera poses) collapsed to a degenerate solution after optimization.

Registration. In Fig. 4.9, we show the benefit of using canonical embeddings (Sec 4.3.3), and measured 2D flow (Eq. 4.16) to register observations across videos and within a video. Without the canonical embeddings and corresponding losses (Eq. 4.17-4.18), each video will be reconstructed separately. With no flow reconstruction loss, multiple ghosting structures are reconstructed due to failed registration.

Deformation modeling. We demonstrate the benefit of using our neural blend skinning model (Sec 4.3.2) on an `eagle` sequence, which is challenging due to its

Table 4.4: **Results on AMA swing and samba.** 3D Chamfer distance (cm, ↓) and F-score (%, ↑) averaged over all frames.

Method	CD	F@1%	F@2%	F@5%
number-bone=4	9.88	28.1	52.4	84.1
number-bone=9	9.08	31.2	56.4	86.8
number-bone=16	**9.02**	**31.8**	**57.2**	87.2
number-bone=25	9.08	**31.8**	57.0	87.1
–w/o in-surface loss	9.14	29.9	54.8	86.7
–quad. embedding	9.70	29.8	54.2	85.4
number-bone=64	9.18	31.1	56.6	**87.5**
number-bone=100	9.11	31.4	56.7	87.3
pose error ϵ=20°	**8.75**	**30.9**	**57.0**	**88.1**
pose error ϵ=50°	8.91	29.8	56.1	**88.1**
pose error ϵ=90°	9.91	28.4	54.8	85.7
coverage=90° (2 vids)	10.61	29.3	54.3	84.1
coverage=180° (4 vids)	**8.94**	**33.0**	**59.8**	**87.9**
coverage=270° (6 vids)	9.09	29.8	56.1	87.6
active-sample=0%	9.63	29.1	53.7	85.8
active-sample=25%	**8.60**	**32.3**	**57.9**	**88.0**
active-sample=50%	9.14	29.9	54.8	86.7

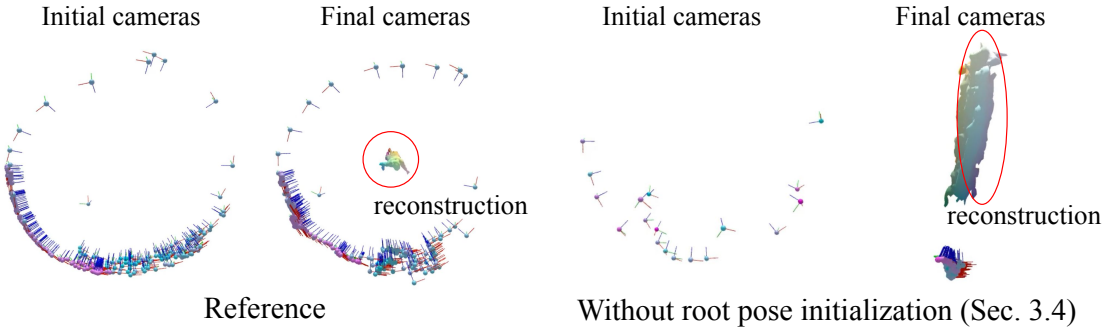

Figure 4.8: **Diagnostics of root pose initialization (Sec.4.3.4).** With randomly initialized root poses, the estimated poses (on the right) collapsed to a degenerate solution, causing reconstruction to fail.

large wing articulations. If we swap neural blend skinning for MLP-SE(3) [179], the reconstruction is less regular. If we swap for MLP-translation [132, 192], we observe

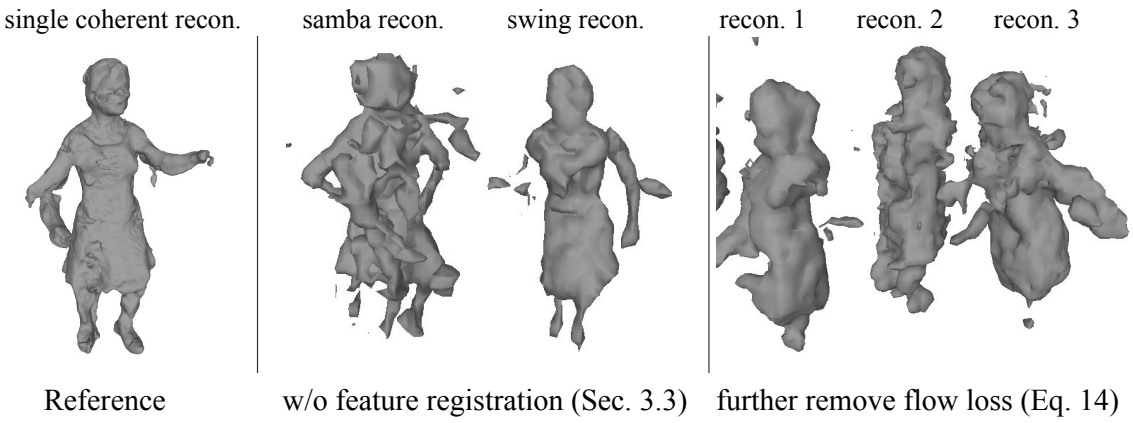

Figure 4.9: **Diagnostics of registration (Sec. 4.3.3).** Without canonical embeddings (middle) or flow loss (right), our method fails to register frames to a canonical model, creating ghosting effects.

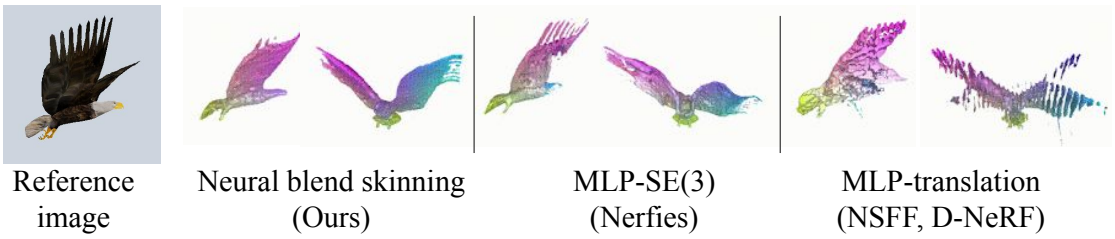

Figure 4.10: **Diagnostics of deformation modeling (Sec.4.3.2).** Replacing our neural blend skinning with MLP-SE(3) results in less regular deformation in the non-visible region. Replacing our neural blend skinning with MLP-translation as in NSFF and D-Nerf results in reconstructing ghosting wings due to significant motion.

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.

Ability to leverage more videos. We compare BANMo to Nerfies in its 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. As shown in Fig. 4.11, 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

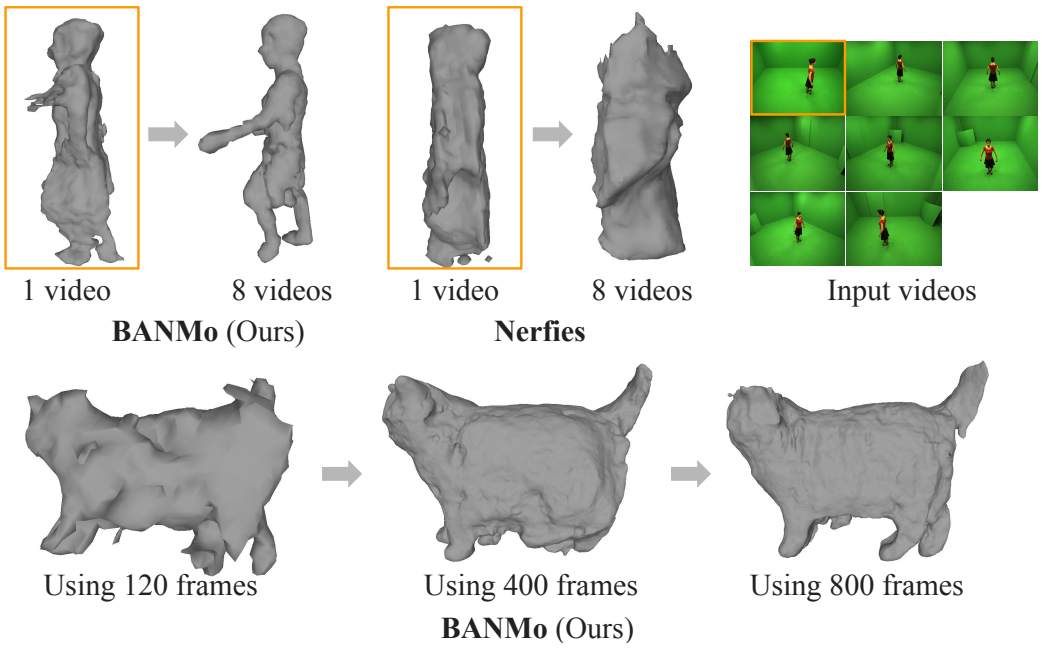

Figure 4.11: **Reconstruction completeness vs number of input videos and video frames.** BANMo is capable of registering more input videos if they are available, improving the reconstruction.

Number and location of bones As shown in the first group of Tab. 4.4 and Fig. 4.12, using too few bones fails to recover all body parts due to over-regularization. Using more than 16 bones produces good reconstructions, but consumes more memory when computing skinning weights. Enforcing them to stay close to the surface with a sinkhorn divergence loss improves the results (Tab. 4.4, L16-17).

Sensitivity to incorrect initial pose We inject different levels of Gaussian noise into the initial poses, leading to average rotation errors $\epsilon \in \{20, 50, 90\}^\circ$. As shown in the second group of Tab. 4.4, BANMo is stable up to 50° rotation error.

Pre-trained embeddings Pre-trained embeddings help BANMo outperform Nerfies, but it is not crucial given good initial root poses ($\epsilon = 12.8 \pm 8.9^\circ$). As shown in Tab. 4.4, using embeddings pre-trained for quadruped animals for human optimization produces slightly worse results.

How much data are needed? To reconstruct a complete shape, BANMo requires all object surface to be visible from at least one frame. Beside completeness, more videos allows to estimate better skinning weights and a more regular motion. We

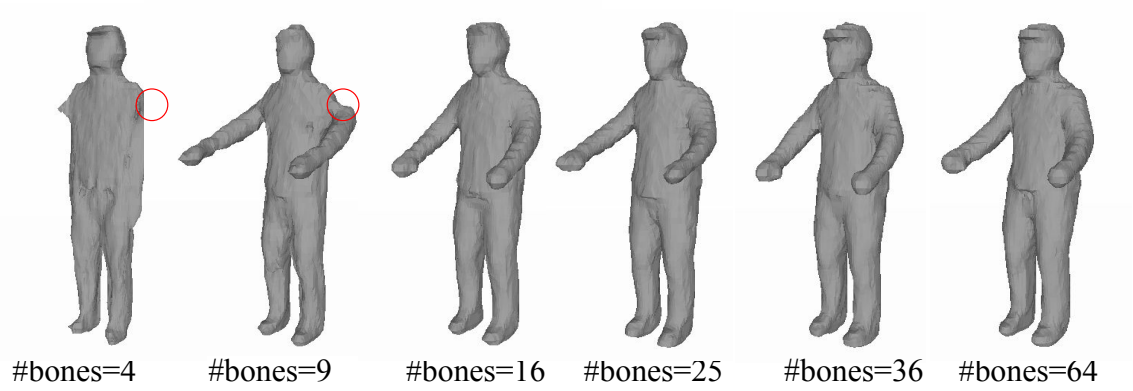

Figure 4.12: **Sensitivity to number of bones.**

evaluate view coverage in the third group of Tab. 4.4.

Importance sampling We use active sampling to avoid sampling from uninformative frames and pixels. It consistently improves reconstruction results as shown in the last group of Tab. 4.4.

Bone re-initialization We qualitatively evaluate the effect of rest bone re-initialization, which re-initializes bone parameters according to the current estimation of shape. As shown in Fig. 4.13, without re-initializing the bones, the optimization may stuck at bad local optima and the final reconstruction may become less accurate.

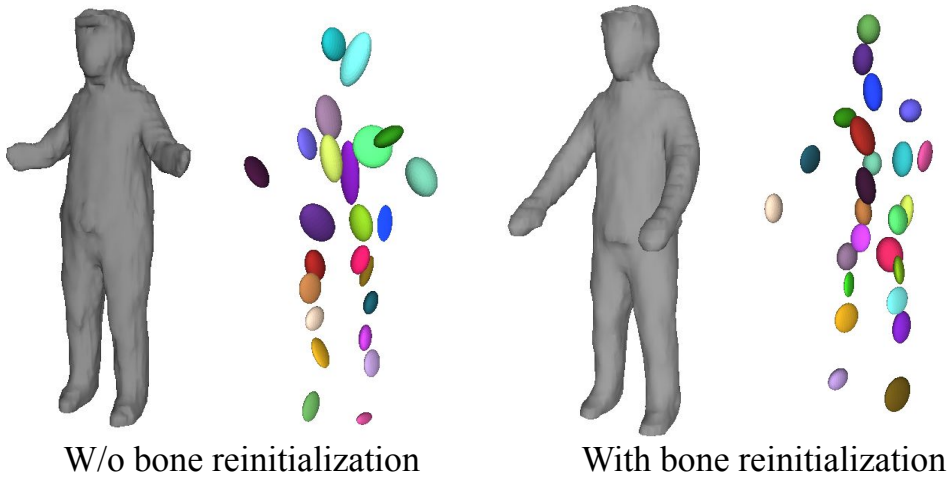

Figure 4.13: **Effect of bone re-initialization.** We find it important to re-initialize rest bone parameters after finding a better approximation of object geometry.

Delta skinning weights We qualitatively evaluate the effect of delta skinning weights. As shown in Fig. 4.14, without learning a delta skinning weights specific to each 3D point, the reconstructed shape and motion may be over-regularized by the 3D Gaussians.

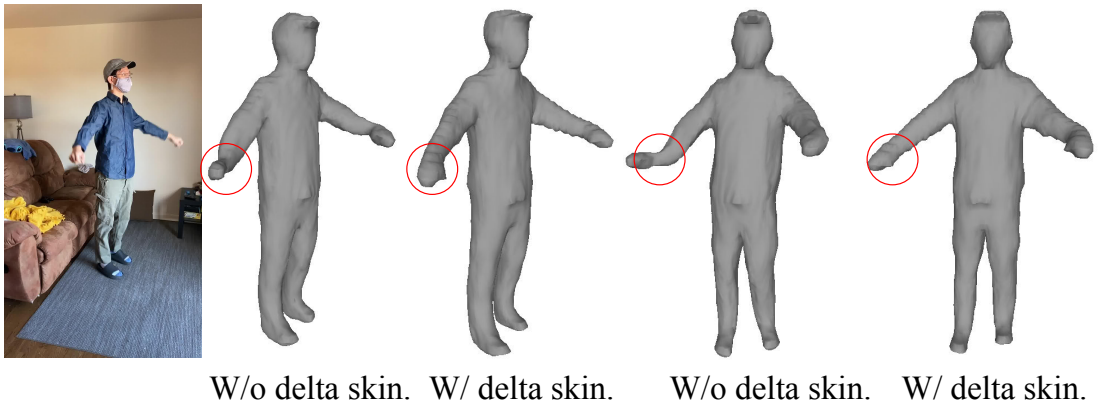

Figure 4.14: **Effect of delta skinning weights.** We find it important to learn a point-specific delta skinning weight function to reconstruction motions in high-quality.

Active sampling. We show the effect of active sampling on a `casual-cat` video (Fig. 4.15): removing it results in slower convergence and inaccurate geometry.

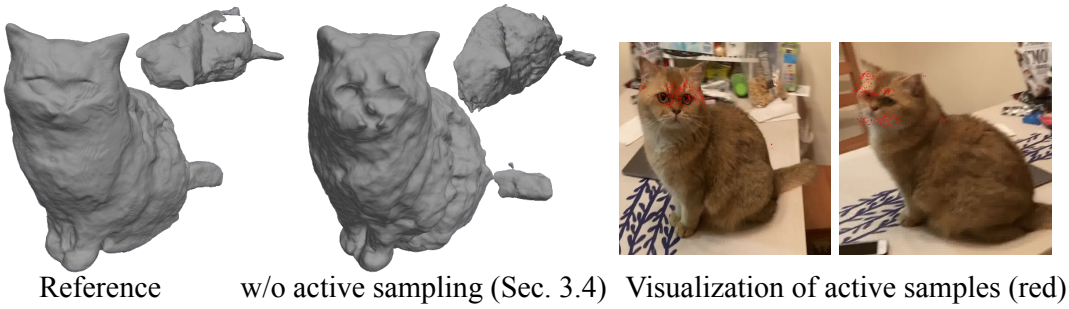

Figure 4.15: **Diagnostics of active sampling over** (x, y)**.** With no active sampling, our method converges slower and misses details (such as ears and eyes). Active samples focus on face and boundaries pixels where the color reconstruction errors are higher.

4.4.4 SFM root pose initialization

COLMAP [221, 223] failed to converge when focused on the deformable object due to violation of rigidity, leading to very few successful registrations (18 over 811 images registered on `casual-cat`). A recent end-to-end method, DROID-SLAM [250], registered all the images but the accuracy is low compared to PoseNet, as shown in Tab. 4.5. We also tried SFM to estimate and compensate for the camera motion (using background as rigid anchor), but this did not help to recover the pose of the object due to its global movement w.r.t. to the background.

Table 4.5: **Evaluation on root pose prediction.** Mean and standard deviation of the rotation error (°) over all frames (↓). We use BANMo-optimized poses as ground-truth. Rotations are aligned to the ground-truth by a global rotation under chordal L2 distance.

Method	`c-cat`	`c-human`	`ama-human`
CSE-PoseNet	**18.6**±16.2	**12.8**±8.9	**11.8**±17.4
DROID-SLAM	65.5 ± 44.5	55.8 ± 39.2	83.6 ± 50.5

4.4.5 Qualitative results

We refer readers to our supplementary webpage https://banmo-www.github.io/ for complete qualitative results.

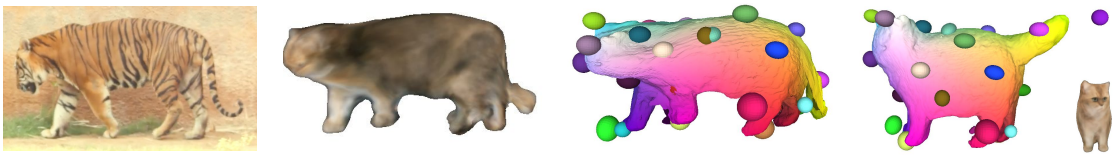

Figure 4.16: **Motion re-targeting from a pre-optimized cat model to a tiger.** Color coded by point locations in the canonical space.

4.4.6 Application: Motion retargeting

As a distinctive application, we demonstrate BANMo's ability of to retarget 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. 4.16. 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 $\boldsymbol{\omega}_e^t$.

Part II

Building 4D Models of Categories

Chapter 5

Reconstructing Animatable Categories from Videos

5.1 Introduction

We aim to build animatable 3D models for deformable object categories. Prior work has done so for targeted categories such as people (e.g., SMPL [7, 144]) and quadruped animals (e.g., SMAL [24]), but such methods appear challenging to scale due to the need of 3D supervision and registration. Recently, test-time optimization through differentiable rendering [179, 180, 192, 276, 308] provides a pathway to generate high-quality 3D models of deformable objects and scenes from monocular videos. However, such models are typically built *independently* for each object instance or scene. In contrast, we would like to build *category* models that can generate different instances along with deformations, given *causally-captured video collections*.

Though scalable, such data is challenging to leverage in practice. One challenge is how to learn the *morphological variation* of instances within a category. For example, `husky`s and `corgi`s are both `dog`s, but have different body shapes, skeleton dimensions, and texture appearance. Such variations are difficult to disentangle from the variations *within* a single instance, e.g., as a dog articulates, stretches its muscles, and even moves into different illumination conditions. Approaches for disentangling such factors require enormous efforts in capture and registration [7, 26], and doing so without explicit supervision remains an open challenge.

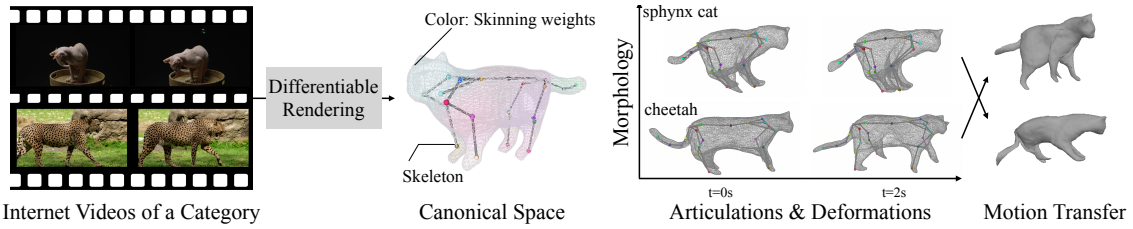

Figure 5.1: Given videos of a deformable category and a skeleton, we reconstruct an animatable 3D model that factorizes variations *across* instances (e.g., `cheetah`'s and `sphynx`'s are both `cat`s but with different shape morphology, skeleton dimensions, and texture) from time-specific variations *within* an instance (e.g., skeleton articulations and elastic shape deformation). **Left**: Input videos; **Middle-left**: 3D shape, skeleton, and skinning weights (visualized as surface colors) in the canonical space; **Middle-right**: Disentangled between-instance and within-instance variations over time. **Right**: Morphology and motion transferred across the two instances.

Another challenge arises from the impoverished nature of in-the-wild videos: objects are often *partially observable* at a limited number of viewpoints, and input signals such as segmentation masks can be inaccurate for such "in-the-wild" data. When dealing with partial or impoverished video inputs, one would want the model to listen to the common structures learned across a category – e.g., dogs have two ears. On the other hand, one would want the model to stay faithful to the input views.

Our approach addresses these challenges by exploiting three insights: (1) We learn skeletons with constant bone lengths within a video, allowing for better disentanglement of between-instance morphology and within-instance articulation. (2) We regularize the unobserved body parts to be coherent across instances while remaining faithful to the input views with a novel code-swapping technique. (3) We make use of a category-level background model that, while not 3D accurate, produces far better segmentation masks. We learn animatable 3D models of cats, dogs, and humans which outperform prior art. Because our models register different instances with a canonical skeleton, we also demonstrate motion transfer across instances.

5.2 Related Works

Model-based 3D Reconstruction. A large body of work in 3D human and animal reconstruction uses parametric shape models [144, 182, 267, 287, 333, 334], which are built from registered 3D scans of human or animals, and serve to recover 3D shapes given a single image or video at test time [9, 25, 25, 117, 117, 211, 335]. A recent research focus is to combine statistical human body mode with implicit functions [101, 128, 213, 214, 215, 291, 326] to improve the robustness and fidelity of the reconstruction. Although parametric body models achieve great success in reconstructing humans with large amounts of ground-truth 3D data, it is unclear how to apply the same methodology to categories with limited 3D data, such as animals, and how to scale to real-life imagery with diverse clothing and body poses. RAC builds category-level shape models from in-the-wild videos and demonstrates the potential to reconstruct 3D categories without sophisticated manual processing.

Category Reconstruction from Image Collections. A number of recent methods build deformable 3D models of object categories from images with weak 2D annotations, such as keypoints and object silhouettes, obtained from human annotators or predicted by off-the-shelf models [73, 108, 118, 131, 259, 283, 316]. However, those methods do not distinguish between morphological variations and motion over time. Moreover, they often apply heavy regularization on shape and deformation to avoid degenerate solutions, which also smooths out fine-grained details. Recent research combines neural implicit functions [163, 165] with category modeling in the context of 3D data generation [39, 40, 173], where shape and appearance variations over a category are modeled with conditional NeRFs. However, reconstructions are typically focused on near-rigid objects such as faces and vehicles.

Articulated Object Reconstruction from Videos. Compared to image collections, videos provide signals to reconstruct object motions and disentangle them from morphological variations. Some works [129, 175, 238] reconstruct articulated bodies from videos, but they either assume synchronized multi-view recordings or articulated 3D skeleton inputs that make their approaches less general. Some other works [306, 307, 308] learn animatable 3D models from monocular videos capturing the same object instance, without disentangling morphology and motion. There are recent methods [130, 282, 307] using in-the-wild videos to reconstruct 3D models

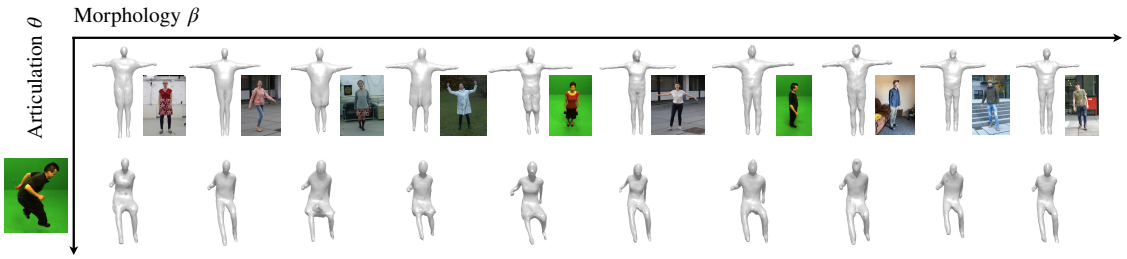

Figure 5.2: **Disentangling morphologies β and articulation θ.** We show different morphologies (body shape and clothing) given the same rest pose (**top**) and bouncing pose (**bottom**).

animals, but their quality are relatively low.

5.3 Method

Given video recordings of different instances from a category and a pre-defined skeleton, we build animatable 3D models including instance-specific morphology (Sec. 5.3.1), time-varying articulation and deformation (Sec. 5.3.2), as well as a video-specific 3D background model (Sec. 5.3.3). The models are optimized using differentiable rendering (Sec. 5.3.4). An overview is shown in Fig. 5.3.

5.3.1 Between-Instance Variation

Fusing videos of different instances into a category model requires handling the morphological variations, which includes the changes in both *internal skeleton* and *outward appearance* (shape and color). We define a video-specific morphology code $\boldsymbol{\beta}$ to control the variations of both the shape and the skeleton.

To model between-instance shape variations, one could use dense warping fields to deform a canonical template into instance-specific shapes [288]. However, warping fields cannot explain topological changes (e.g., different clothing). Instead, we define a hierarchical representation: a conditional canonical field [41, 180, 280] to handle fine-grained variations over a category (e.g., the ears of dogs) and a stretchable bone model [92, 284] to represent coarse shape variations (e.g., height and size of body parts).

Conditional Field T. In the canonical space, a 3D point $\mathbf{X} \in \mathbb{R}^3$ is associated with

three properties: signed distance $d \in \mathbb{R}$, color $\mathbf{c} \in \mathbb{R}^3$, and canonical features $\boldsymbol{\psi} \in \mathbb{R}^{16}$, which is used to register pixel observations to the canonical space [170, 307]. These properties are predicted by multi-layer perceptron (MLP) networks:

$$(d, \mathbf{c}^t) = \mathbf{MLP}_{\text{SDF}}(\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\omega}_a), \tag{5.1}$$

$$\boldsymbol{\psi} = \mathbf{MLP}_{\boldsymbol{\psi}}(\mathbf{X}), \tag{5.2}$$

where the shape and color are conditioned on a video-specific morphology code $\boldsymbol{\beta} \in \mathbb{R}^{32}$ [97, 173]. We further ask the color to be dependent on an appearance code $\boldsymbol{\omega}_a \in \mathbb{R}^{64}$ that captures frame-specific appearance such as shadows and illumination changes [157].

Skeleton J. Unlike shape and color, the bone structures are not directly observable from imagery, making it ambiguous to infer. Methods for automatic skeletal rigging [125, 174, 300] either heavily rely on shape priors, or appear sensitive to input data. Instead, we provide a category-level skeleton topology, which has a fixed tree topology with (B+1) bones and B joints (B=25 for quadruped and B=18 for human). To model cross-instance morphological changes, we define per-instance joint locations as:

$$\mathbf{J} = \mathbf{MLP}_{\mathbf{J}}(\boldsymbol{\beta}) \in \mathbb{R}^{3 \times B}. \tag{5.3}$$

As we will discuss next, the change in joint locations not only stretches the skeleton, but also results in the elongation of canonical shapes as shown in Fig. 5.3 (c). The skeleton topology is fixed through optimization but $\mathbf{J}$ is specialized to each video.

Skinning Field W. For a given 3D location $\mathbf{X}$, we define skinning weight $\mathbf{W} \in \mathbb{R}^{B+1}$ following BANMo:

$$\mathbf{W} = \sigma_{\text{softmax}}\big(d_\sigma(\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\theta}) + \mathbf{MLP}_{\mathbf{W}}(\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\theta})\big), \tag{5.4}$$

where $\boldsymbol{\theta}$ is a articulation code and $d_\sigma(\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\theta})$ is the Mahalanobis distance between $\mathbf{X}$ and Gaussian bones under articulation $\boldsymbol{\theta}$ and morphology $\boldsymbol{\beta}$, refined by a delta skinning MLP. Each Gaussian bone has three parameters for center, orientation, and scale respectively, where the centers are computed as the midpoint of two adjacent joints, the orientations are determined by the parent joints, and the scales are

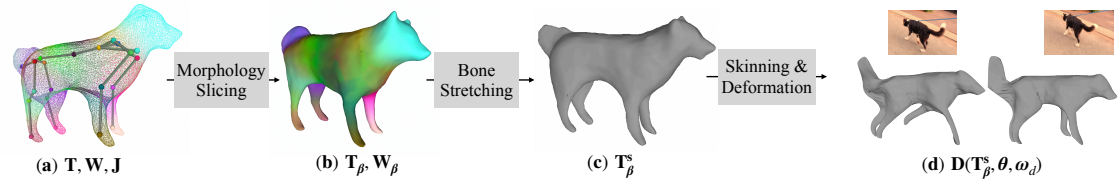

Figure 5.3: **Morphological variations vs time-varying articulation and deformation.** (a) Canonical shape $\mathbf{T}$, skinning weights $\mathbf{W}$, and joint locations $\mathbf{J}$. (b) To represent morphological differences between instances, we use a morphology code $\boldsymbol{\beta}$ that specifies instance shape and appearance $\mathbf{T}_{\boldsymbol{\beta}}$, skinning weights $\mathbf{W}_{\boldsymbol{\beta}}$ for a canonical skeleton $\mathbf{J}$. (c) $\boldsymbol{\beta}$ also predicts a change in bone lengths $\Delta\mathbf{J}_{\boldsymbol{\beta}}$ which further *stretches* instance shape into $\mathbf{T}^s_{\boldsymbol{\beta}}$ by elongating body parts. (d) Time-varying articulations are modeled with an articulation vector $\boldsymbol{\theta}$ by linearly blending rigid bone transformations in the dual quaternion space. Time-varying deformations (such as muscle deformation) are modeled with a deformation vector $\boldsymbol{\omega}_d$ through invertible 3D warping fields.

optimized.

Stretchable Bone Deformation. To represent variations of body dimension and part size, prior work [144, 333] learns a PCA basis from registered 3D scans. Since 3D registrations are not available for in-the-wild videos, we optimize a parametric model through differentiable rendering. Given the stretched joint locations, the model deforms the canonical shape $\mathbf{T}_{\boldsymbol{\beta}}$ with blend skinning equations,

$$\mathbf{T}^s_{\boldsymbol{\beta}} = (\mathbf{W}_{\boldsymbol{\beta}}\mathbf{G}_{\boldsymbol{\beta}})\mathbf{T}_{\boldsymbol{\beta}}, \tag{5.5}$$

where $\mathbf{G}_{\boldsymbol{\beta}}$ transforms the bone coordinates, and $\mathbf{W}_{\boldsymbol{\beta}}$ is the instance-specific skinning weights in Eq. (5.4).

5.3.2 Within-Instance Variation

We represent within-instance variations as time-varying warp fields between the canonical space and posed space at time t. Similar to HumanNeRF [280], we decompose motion as *articulations* that explains the near-rigid component (e.g., skeletal motion) and *deformation* that explains the remaining nonrigid movements (e.g., cloth deformation). Note given the complexity of body movements, it is almost certain the pre-defined skeleton would ignore certain movable body parts. Adding deformation is crucial to achieving high-fidelity reconstruction.

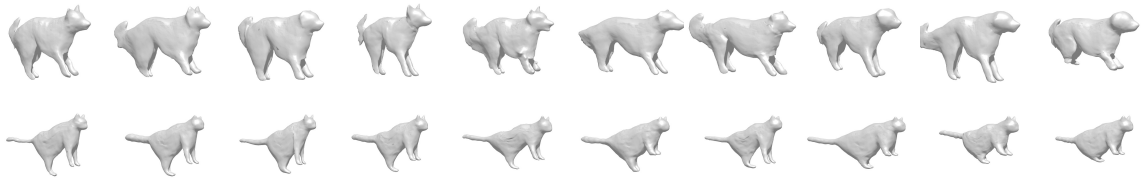

Figure 5.4: **Different $\boldsymbol{\beta}$ morphologies of dogs (top) and cats (bottom)**. Our reconstructions show variance in ear shape and limb size over dog breeds, as well as variance in limb and body size over cat breeds.

Time-varying Articulation. To model the time-varying skeletal movements, we define per-frame joint angles:

$$\mathbf{Q} = \mathbf{MLP}_{\mathbf{A}}(\boldsymbol{\theta}) \in \mathbb{R}^{3\times B} \tag{5.6}$$

where $\boldsymbol{\theta} \in \mathbb{R}^{16}$ is a low-dimensional articulation parameter, and each joint has three degrees of freedom. Given joint angles and the per-video joint locations, we compute bone transformations $\mathbf{G} \in \mathbb{R}^{3\times 4\times B}$ via forward kinematics. We apply dual quaternion blend skinning (DQS) [111] to get the warping field for each spatial point,

$$\mathbf{D}(\boldsymbol{\beta}, \theta) = (\mathbf{W}_{\boldsymbol{\beta}}\mathbf{G})\mathbf{T}^{s}_{\boldsymbol{\beta}}. \tag{5.7}$$

Compared to linear blend skinning (LBS), DQS blends SE(3) transformations in dual quaternion space and ensures valid SE(3) after blending, which reduces artifacts around twisted body parts. For more analysis and comparisons, we refer readers to a concurrent work [234] that also applies DQS for deformable object reconstruction. Note the stretching operation in Eq. (5.5) can be fused with articulation as a single blend skinning operation.

Time-varying Soft Deformation. To further explain the dynamics induced by non-skeleton movements (such as the cat belly and human clothes), we add a neural deformation field [126, 179] $\mathcal{D}(\cdot)$ that is flexible enough to model highly nonrigid deformation. Applying the fine-grained warping after blend skinning, we have

$$\mathbf{D}(\boldsymbol{\beta}, \theta, \omega_d) = \mathcal{D}(\mathbf{D}(\boldsymbol{\beta}, \theta), \omega_d), \tag{5.8}$$

where ω_d is a frame-specific deformation code. Inspired by CaDeX [126], we use

real-NVP [49] to ensure the 3D deformation fields are invertible by construction.

Invertibility of 3D Warping Fields. For a given time instance t, we have defined a forward warping field $\mathcal{W}^{t,\rightarrow}$ that transforms 3D points from the canonical space to the specified time instance, and a backward warping field $\mathcal{W}^{t,\leftarrow}$ to transform points in the inverse direction. Both warping fields include stretching (Eq. (5.5)), articulation. (Eq. (5.7)), and deformation (Eq. (5.8)) operations. Notably, we only need to define each operation in the forward warping fields. The deformation operation is, by construction, invertible. To invert stretching and articulation, we invert SE (3) transformations $\mathbf{G}$ in the blend skinning equations and compute the skinning weights with Eq. (5.4) using the corresponding morphology and articulation codes. A 3D cycle loss is used to ensure that the warping fields are self-consistent after a forward-backward cycle [132, 308].

5.3.3 Scene Model

Reconstructing objects from in-the-wild video footage is challenging due to failures in segmentation, which is often caused by out-of-frame body parts, occlusions, and challenging lighting. Inspired by background subtraction [95, 226], we build a model of the background to *robustify* our method against inaccurate object segmentation.

In background subtraction, moving objects can be segmented by comparing input images to a background model (e.g., a median image). We generalize this idea to model the background scene in 3D as a per-video NeRF, which can be rendered as color pixels at a moving camera frame and compared to the input frame. We design a conditional background model that generates density and color of a scene conditioned on a per-video background code $\boldsymbol{\gamma}$:

$$(\sigma, \mathbf{c}^t) = \mathbf{MLP_{bg}}(\mathbf{X}, \mathbf{v}, \boldsymbol{\gamma}), \tag{5.9}$$

where $\mathbf{v}$ is the viewing direction. To render images, we compose the density and color of the object field and the background NeRF in the view space [173], and compute the expected color and optical flow. Background modeling and composite rendering allows us to remove the object silhouette loss, and improves the quality of results. Interestingly, we find that even *coarse* geometric reconstructions of the background still can improve the rendered 2D object silhouette, which in turn is

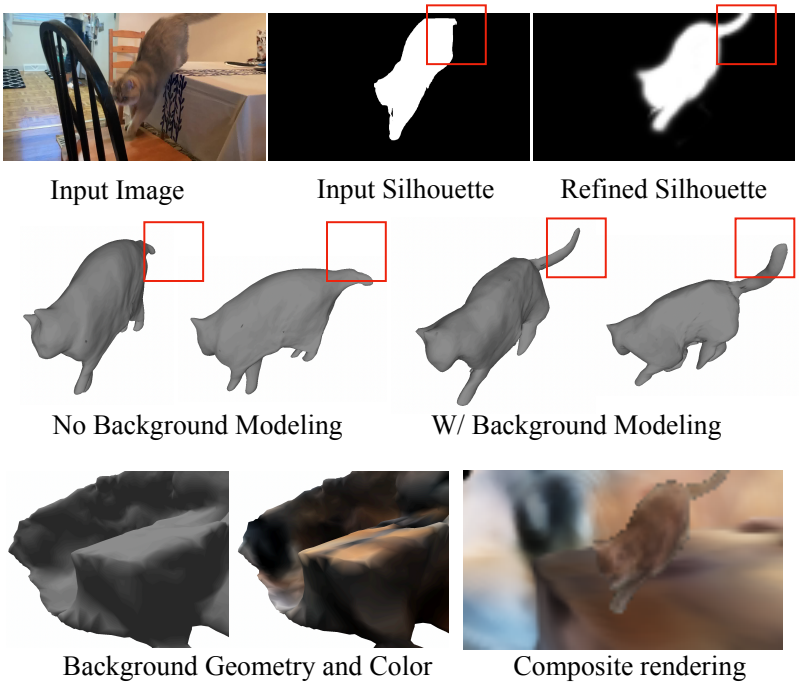

Figure 5.5: **Joint foreground and background reconstruction.** We jointly reconstruct objects and their background, while refining the segmentation. Note the input silhouette is noisy (e.g., tail was not segmented), and background modeling helps produce an accurate refined silhouette. As a result, RAC is robust to inaccurate segmentation (e.g., tail movements marked by the red box).

useful for improving the quality of object reconstructions (Fig. 5.5). We ablate the design choice in Tab. 5.1.

5.3.4 Losses and Regularization

Given the videos and a predefined skeleton, we optimize the parameters discussed above: (1) canonical parameters $\{\boldsymbol{\beta}, \mathbf{J}, \mathbf{T}, \mathbf{W}\}$ including per-video morphology codes and canonical templates; (2) motion parameters $\{\boldsymbol{\theta}, \boldsymbol{\omega}_{\mathbf{d}}, \mathbf{A}, \mathbf{D}\}$ including per-frame codes as well as articulation and soft deformation MLPs. (3) background parameters $\{\boldsymbol{\gamma}, \mathbf{B}\}$ including video background codes and a background NeRF. The overall objective function contains a image reconstruction loss term and regularization terms.

Reconstruction Losses. The reconstruction loss is defined as the difference between

rendered and observed images, including object silhouette, color, flow, and features:

$$\mathcal{L} = \mathcal{L}_{\text{sil}} + \mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{OF}} + \mathcal{L}_{\text{feat}}. \tag{5.10}$$

We update the model parameter by minimizing $\mathcal{L}$ through differentiable volume rendering in the same way as BANMo [308]. Off-the-shelf estimates of object silhouettes are used as supervision to kick-start the optimization. Then the weight of silhouette term is set to 0 after several iterations of optimization, while composite rendering of foreground and background itself is capable of separating the object and the non-object components.

Morphology Code ($\boldsymbol{\beta}$) Regularization. Existing differentiable rendering methods are able to faithfully reconstruct the input view but not able to hallucinate a reasonable solution for occluded body parts [179, 308]. See Fig. 5.9 for an example. One solution is to regularize the instance-specific morphology code $\boldsymbol{\beta}$ to be consistent with the body shapes observed in *other* videos. Traditional approaches might do this by adding variational noise (as in a VAE) or adversarial losses (as in a GAN). We found good results with the following easy-to-implement approach: we randomly *swap* the morphology code $\boldsymbol{\beta}$ of two videos during optimization; this regularizes the model to effectively learn a single morphology code that works well across all videos. But naively applying this approach would produce a single morphology that would not specialize to each object instance. To enable specialization, we *gradually* decrease the probability of swaps during the optimization, from $\mathcal{P} = 1.0 \rightarrow 0.05$.

Joint J Regularization. Due to the non-observable nature of the the joint locations, there might exist multiple joint configurations leading to similar reconstruction error. To register the skeleton with the canonical shape, we minimize Sinkhorn divergence [56] between the canonical surface $\mathbf{T}_{\boldsymbol{\beta}}$ and the joint locations $\mathbf{J}_{\boldsymbol{\beta}}$, which forces them to occupy the same space. We extract the canonical mesh with marching cubes [145] as a proxy of the canonical surface. Sinkhorn distance interpolates between Wasserstein and kernel distances and defines a soft way to measure the distance between shapes with different density distributions.

Soft Deformation Regularization The soft deformation field has the capacity of explaining not only the soft deformations, but also the skeleton articulations. Therefore, we penalize the L2 norm of the soft deformation vectors at randomly

Table 5.1: **Quantitative results on AMA sequences.** 3D Chamfer distance (cm, ↓) and F-score (%, ↑) averaged over all frames. Our model is trained on 47 human videos spanning existing human datasets (as described in Sec.4.2); we also train BANMo on the same set. Other baselines are trained on 3D human data and relies on SMPL model. Results with ^{S} indicates variants trained on single instances. Our model outperforms prior works.

Method	samba			bouncing		
	CD	F@2%	F@5%	CD	F@2%	F@5
HuMoR	9.8	47.5	83.7	11.5	45.2	82.3
ICON	10.1	39.9	85.2	9.7	53.5	86.4
BANMo^S	8.0	62.2	89.1	7.6	64.7	91.1
BANMo	9.3	54.4	85.5	10.2	54.4	86.5
RAC^S	6.4	70.9	93.2	**6.9**	**66.7**	**92.8**
RAC	**6.0**	**72.5**	**94.4**	8.0	63.8	91.4
w/o skeleton	8.6	59.6	87.7	9.3	59.5	87.8
w/o $\boldsymbol{\beta}$	8.5	58.9	87.5	8.4	62.5	90.6
$\boldsymbol{\beta}$ swap$\rightarrow ||\boldsymbol{\beta}||_2^2$	6.5	69.0	93.8	8.0	64.8	91.3
+ bkgd NeRF	6.3	70.9	93.7	7.4	65.5	91.8

sampled morphology and articulations,

$$\mathcal{L}_{\text{soft}} = \|\mathbf{D}(\boldsymbol{\beta}, \theta, \omega_d) - \mathbf{D}(\boldsymbol{\beta}, \theta)\| \,. \tag{5.11}$$

5.4 Experiments

Implementation Details. We build RAC on BANMo and compute bone transformations from a kinematic tree. The soft deformation field follows CaDeX, where we find that two invertible blocks are capable of handling moderate deformations. To evaluate surface reconstruction accuracy, we extract the canonical mesh $\mathbf{T}$ by finding the zero-level set of SDF with marching cubes on a 256^3 grid. To get the shape at a specific time, the canonical mesh is forward-warped with $\mathcal{W}^{t,\rightarrow}$.

Optimization Details We use AdamW to optimize the model for 36k iterations with 16384 rays per batch (taking around 24 hours on 8 RTX-3090 GPUs). We fist pre-train the background model with RGB, optical flow, and surface normal losses while ignoring foreground pixels. Then we combine background models with the

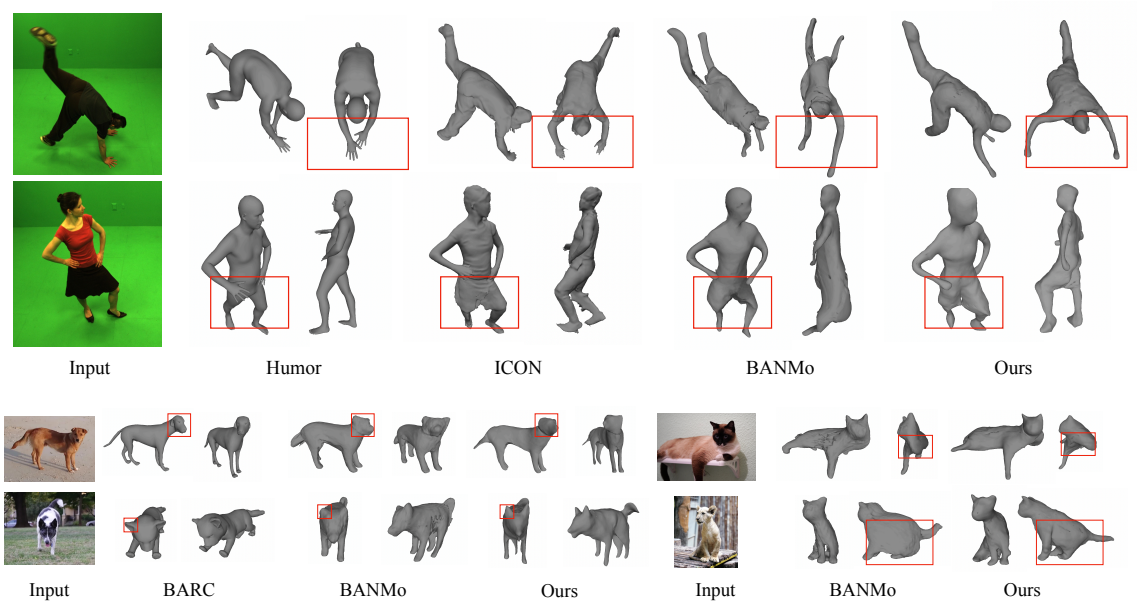

Figure 5.6: **Qualitative comparison.** We compare with BANMo and model-based methods (HuMoR, ICON, BARC). **Top**: human reconstruction on (`AMA`). **Bottom:** dogs and cats reconstruction on internet videos.

object model for composite rendering optimization. The weights for the loss terms are tuned to have similar initial magnitude. The object root poses are initialized with single-image viewpoint networks trained for humans and quadruped animals following BANMo [308]. For all categories, we start with the same shape (a unit sphere) and a known skeleton topology. Both the shape and the joint locations are specialized to the input dataset, as shown in Fig. 5.7.

5.4.1 Reconstructing Humans

Dataset. We combine existing human datasets, including AMA, MonoPerfCap, DAVIS, and BANMo to get 47 human videos with 6,382 images [189, 266, 297]. AMA contains multi-view videos, but we treat them as monocular videos and *do not* use the time-synchronization or camera extrinsics. During preprocessing, we use PointRend [116] to extract object segmentation, CSE [171] for pixel features, VCN-robust [301] for optical flow, and omnidata [53] for surface normal estimation.
Metrics. We use AMA for evaluation since it contains ground-truth meshes and follow BANMo to compute both Chamfer distances and F-scores. Chamfer distance

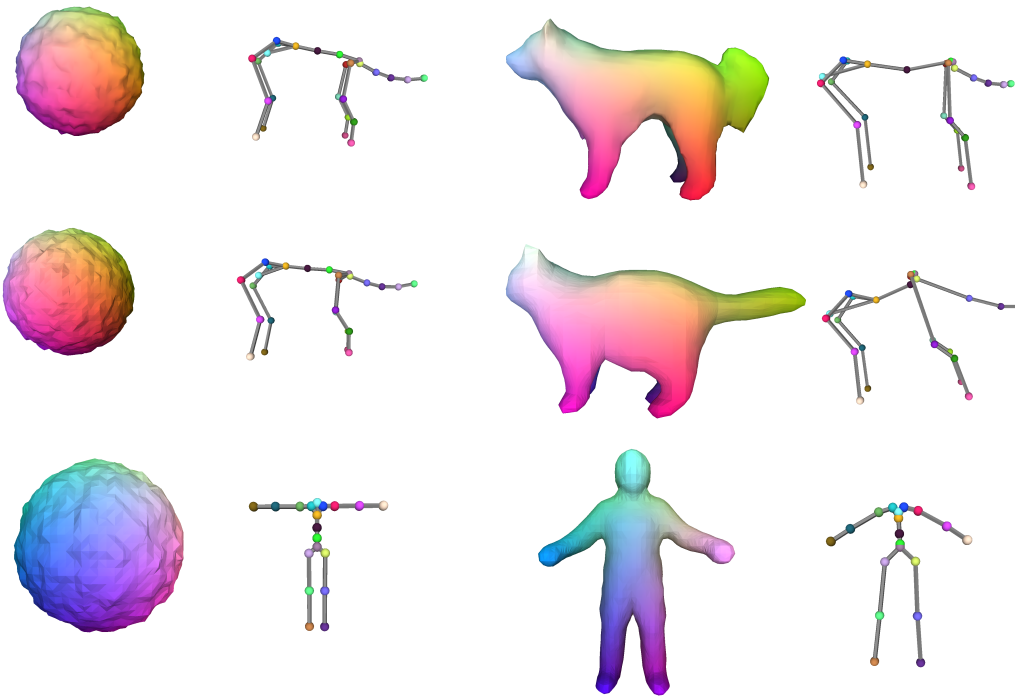

Figure 5.7: **Shape and skeleton optimization.** From top to bottom, we visualize the canonical shape and skeleton of our dog, cat, and human models. **Left**: Canonical shape and skeleton before optimization. **Right**: Canonical shape and skeleton after optimization.

computes the average distance between the ground-truth and the estimated surface points by finding the nearest-neighbor matches, but it is sensitive to outliers. F-score at distance thresholds $d \in \{1\%, 2\%, 5\%\}$ of the human bounding box size [248] provides a more informative quantification of surface reconstruction error at different granularity. To account for the unknown scale, we align the predicted mesh with the ground-truth mesh using their depth in the view space.

Baselines. On AMA, we compare with template-free BANMo [308] and model-based methods, including HuMoR [205] and ICON [291]. BANMo reconstructs an animatable 3D model from multiple monocular videos of the same instance, powered by differentiable rendering optimization. We optimize BANMo on the same dataset with the same amount of computation and iterations as ours. HuMoR is a temporal pose and shape predictor for humans. It performs test-time optimization on video sequences leveraging OpenPose keypoint detection and motion priors trained on

large-scale human motion capture dataset. We run it on each video sequence, and processing 170 video frames takes around two hours on a machine with Titan-X GPU. ICON is the recent SOTA method on single view human reconstruction. It combines statistical human body models (SMPL) with implicit functions and is trained on 3D human scans with clothing. Notably, it performs test-time optimization to fit surface normal predictions to improve the pose accuracy and reconstruction fidelity. We run it per frame, and processing a 170 frame video takes around three hours on an RTX-3090 GPU.

Results. We show qualitative comparison with baselines in Fig. 5.6 top row, and quantitative results in Tab. 5.1. On the handstand sequence, HuMoR works well for common poses but fails where the performer is not in an upright pose. ICON works generally well, but the hand distances appear not physically plausible (too short) from a novel viewpoint. BANMo reconstruction also failed to reconstruct the unnatural upside-down pose. In contrast, RAC successfully reconstructs the handstand pose with plausible hand distances. On the samba sequence, HuMoR correctly predicts body poses, but fails to reconstruct the cloth and its deformation. ICON predicts a broken dress and distorted human looks from a novel viewpoint, possibly due to lack of diverse training data from dressed humans. When applied to 47 videos of different humans, BANMo fails to model the cloth correctly, possibly because a limited number of control points are not expressive enough to model the morphological variations over humans wearing different clothes. RAC models between-shape variations using a conditional canonical model and successfully reconstructs cloth deformation using the soft deformation field.

Our quantitative results align with qualitative observations, where RAC outperforms all baselines except being slightly worse than BANMo trained on single instances (S). However, RAC trained on single instances (S) or multiple instances (M) outperforms BANMo trained in either fashion. In particular, BANMo results are notably worse when trained on multiple instances, indicating the difficulty in building category models. In contrast, RAC become better when trained on multiple instances.

5.4.2 Reconstructing Cats and Dogs

Dataset. We collect 76 cat videos and 85 dog videos from Internet videos, as well as public data from BANMo. All the videos are casually-captured monocular videos. We extract video frames at 10 FPS, including 9,734 frames for cats and 11,657 frames for dogs. We perform the same pre-processing as human reconstruction.

Baselines. We compare with BANMo and model-based BARC [211]. BARC is the current SOTA for dog shape and pose estimation. It trains a feed-forward network using CG dog models and images with keypoint labels. The shape model is based on SMAL, which uses manual rigging and registration to fit 3D animal toys. We run BARC on individual images.

Results. We show qualitative results in Fig. 5.6 bottom row. For dog videos, we find that BARC worked well to predict coarse shapes and poses. However, the results are biased towards certain dog breeds. For instance, BARC predicts a long jaw when the dog has a short jaw (top row), and predicts round ears when the dog has sharp ears (bottom). BANMo was able to reconstruct a reasonable coarse shape, but failed to capture the fine details (e.g., the shape of the ear and the size of the head) with only 25 control points. In contrast, RAC was able to faithfully capture the coarse shape and fine details of the dogs. Unlike dogs, cats have fewer variations in body shape and size, where we find that BANMo works well in most cases. However, for the body parts not visible from the reference viewpoint, BANMo often estimates a squashed shape, which may be caused by the entangled morphology and articulations. In contrast, RAC accurately infers reasonable body parts and articulations even when they are not visible.

5.4.3 Diagnostics

Large Morphological Changes. We reconstruct eight videos of different quadruped animals together to "stress test" our method. The dataset contains two dog videos, two cat videos, and one video for goat, bear, camel, and cow, respectively. The result is shown in Fig. 5.8.

Morphology Code $\boldsymbol{\beta}$ Removing morphology code $\boldsymbol{\beta}$ from the canonical field degrades it to a standard NeRF. We rerun the experiments and the results are shown in Tab. 5.1 as well as Fig. 5.9. Without the morphology code, our reconstructions are forced to

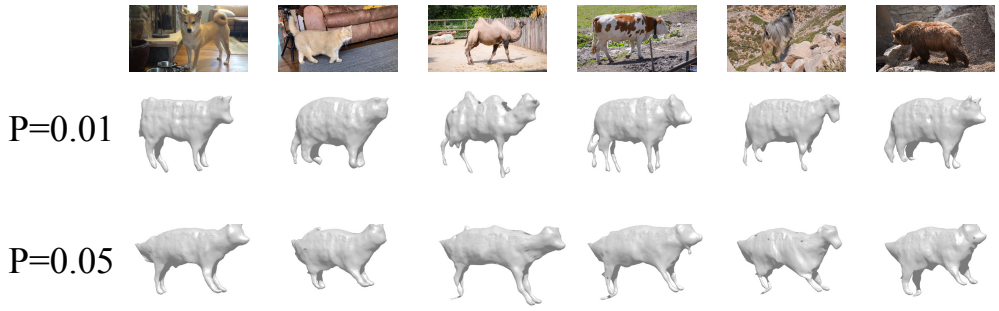

Figure 5.8: **Quadruped Category Reconstruction.** Using a smaller code swapping probability $P = 0.01$ results in more faithful instance shape, but less smooth results. A larger P produces smoother results, but some instance-specific features disappear.

share the same canonical shape, which as discussed in Sec 5.3.1, failed to handle fine deformations and topological changes (e.g., clothing), leading to worse results in all metrics.

Morphology Code Regularization To test the effectiveness of the morphology code regularization, we set $\mathcal{P} = 0$ throughout the optimization. The results are shown in Tab. 5.1 and Fig. 5.9. Without regularization of the morphology code, the reconstructed shape may appear reasonable from the reference viewpoint, but severely distorted from a novel viewpoint. We posit the body parts that are not well-covered in the video are inherently difficult to infer. As a result, the shapes become degenerate without relying on priors from other videos in the dataset. Tab. 5.1 also shows that code swapping outperforms norm regularization [69] ($||\boldsymbol{\beta}||_2^2$). We posit norm regularization forces codes to be similar, but does not constrain their output space, while code swapping encourages *any* code to explain *any* image in the data. In practice, we find that code swapping generates valid outputs when we interpolate between codes.

Soft Deformation. After removing the soft deformation field, RAC fails to recover body parts that are not controlled by the skeleton (Fig. 5.10), such as the ears of the dog.

Motion Transfer. Given the category model with disentangled morphology and articulations, we can easily transfer an articulation in frame t to other video by setting the articulation parameter to $\boldsymbol{\theta}_t$, while keeping the morphology parameter $\boldsymbol{\beta}$ the same. We show motion transfer across human in Fig. 5.2. Please visit our website for video

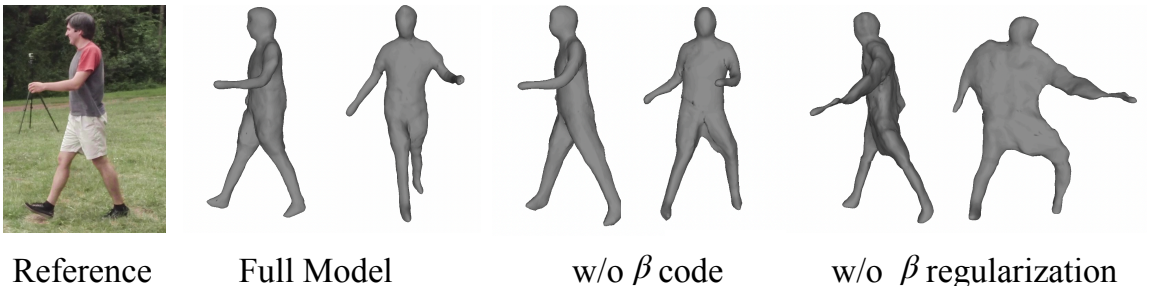

Figure 5.9: **Ablation study on morphology modeling**.

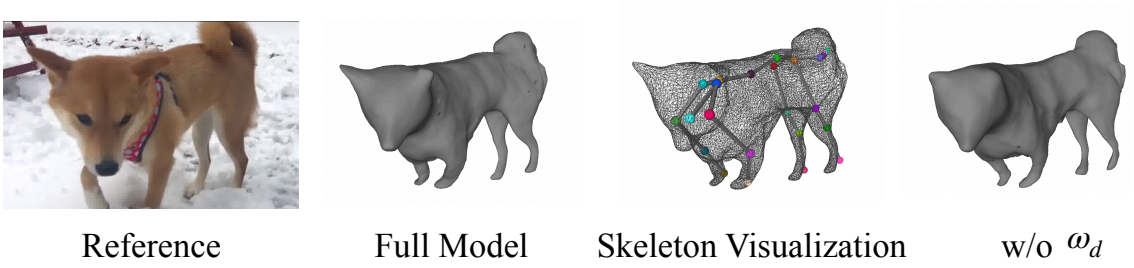

Figure 5.10: **Ablation study on soft deformation ω_d**.

results of dogs, cats, and humans.

Skeleton vs Control Points. Control point deformations are flexible but do not preserve body dimensions (e.g., a line segment can be stretched longer by its end points). As a result, body and limb dimensions can change, creating two problems: (1) articulated shapes look squashy from novel views, and (2) variations in body dimensions are entangled with control-point deformations. In contrast, skeleton deformation preserves body dimensions. It produces better results (Tab. 5.1) and better motion re-targeting (Fig. 5.11).

Stretchable Bones allow for control of bone dimensions after optimization. We show an example of a Dachshund (Source1) warped to a Heeler (Source2) by modifying bone dimensions while keeping the shape unchanged.

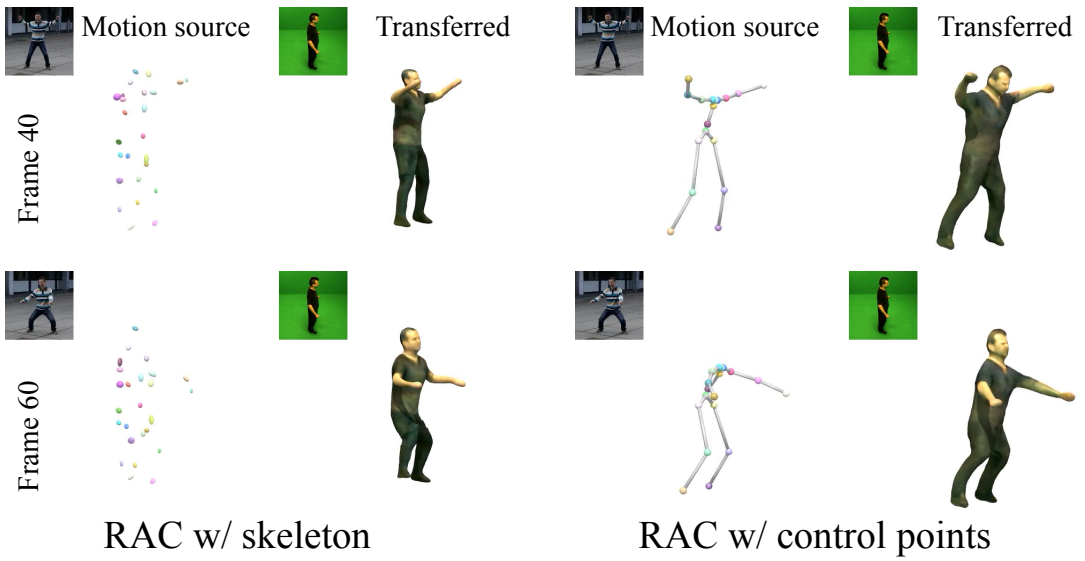

Figure 5.11: **Skeleton vs control-points for motion transfer**. RAC with control points fails to maintain the body dimensions during motion, and produces squashy results when transferred to a new morphology. RAC with skeleton disentangles motion from morphology, allowing for better motion transfer.

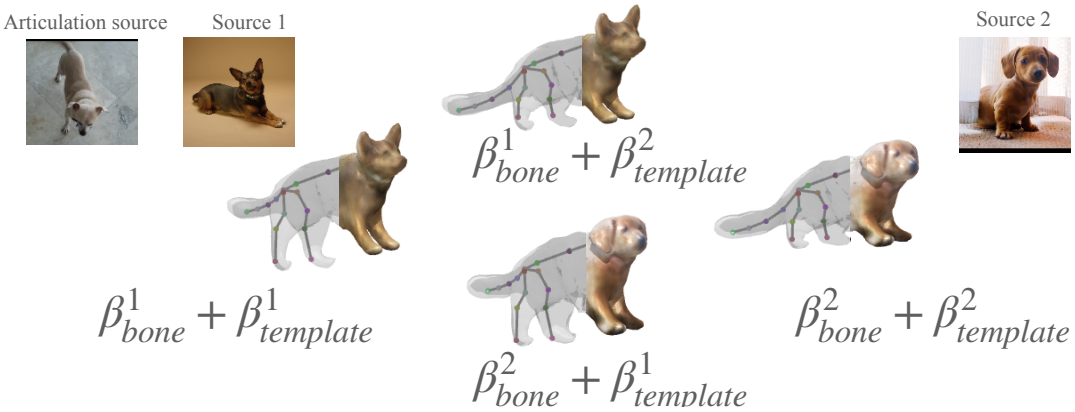

Figure 5.12: Disentanglement of bone dimensions and shape.

Part III

Building 4D Models of Dynamic Scenes

Chapter 6

Physically Plausible Reconstruction from Monocular Videos

6.1 Introduction

Given casually captured monocular RGB videos, we aim to build 3D models of articulated objects and the environment, whose configurations (geometry, motion trajectory, force, and mass distribution) satisfy physics constraints, and can be replayed in a physics simulator.

Reconstructing dynamic 3D structures from monocular videos is challenging due to the under-constrained nature of the problem. Prior works often leverage *first order* constraints. For instance, Nonrigid-SfM explores temporal smoothness and low-rank priors [44] to constrain the problem. Recent works on differentiable rendering and dynamic NeRF utilize divergence-free motion fields [179] or as-rigid-as-possible priors [108]. Although those methods are able to obtain visually appealing reconstruction results from the reference viewpoint, physically-implausible configurations, such as foot sliding, statically-unstable poses, etc., are often observed from a novel viewpoint. An illustrative example is shown in Fig. 6.2.

Physics as a prior. We seek a more principled way to model the time-varying behavior of an object and its interaction with the environment using physics constraints. Physical priors tend to be relatively unexplored as a tool for aiding reconstruction, though important exceptions exist in the domain of monocular human motion capture [62, 204, 228]. One reason that such methods are not more widespread is that

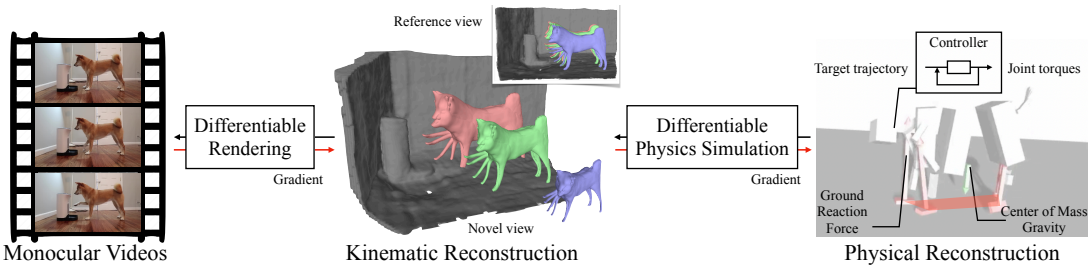

Figure 6.1: Given casually-captured monocular videos (left), PPR builds 3D models of articulated objects and the surrounding environment. Naive kinematic reconstruction (middle) generates a family of solutions, some containing inconsistent physical support and contact dynamics (blue and green color), such as floating or walking with sliding feet. We show that differentiable physics simulation acts as effective regularizer for improving the physical plausibility of visual reconstruction algorithms. As PPR reconstructs the dynamics scene, it also drives a ragdoll in a physics simulator to track the kinematic reconstruction. This ensures the reconstructions are statically stable with ground contact (right), and the center of mass is projected within the support polygon (marked with red). PPR also reports physics estimations, such as ground reaction forces (red arrows) and center of mass (green arrow).

they often make strong assumptions about the target and the scene, for instance, accurate 2D/3D keypoint tracking, known ground plane, and contact state annotations. Moreover, operationalizing such constraints requires the use of heavyweight simulators that may be difficult to integrate with visual reconstruction algorithms.

In this work, we couple differentiable rendering with differentiable physics-based simulation to jointly solve for the object geometry, motion, background scene, and physics parameters including body mass distributions and control parameters. We posit that just as differentiable renderers have lowered the barrier of entry for (neural) 3D modeling, differentiable simulators are also lowering the barrier for incorporating physical constraints. Compared to prior approaches that rely on strong human priors to estimate 3D pose and ground contact, PPR works for more unconstrained settings including both humans and animals in an unknown environment, enabled by end-to-end optimization from videos to physics.

Specifically, we (1) introduce an end-to-end framework for reconstructing physically-plausible dynamic objects and scene configurations from monocular videos; (2) propose an alternating-direction 3D-reconstruction formulation by coupling differentiable physics simulation and differentiable rendering; (3) demonstrate improved reconstruc-

(a) Relative scale (b) 3D tracking with occlusion (c) Ground contact

Figure 6.2: **Physics helps monocular reconstruction.** Naive reconstructions of dynamic scenes from monocular videos are often physically implausible. (**a**) Although the 2D projections of both the red and blue reconstructions align with the input frame (top), their scales and 3D motions are wildly different due to the inherent ambiguity between camera and object motion. Solution 1 (red) correctly touches the ground while solution 2 (blue) appears smaller than its true size and floats in the air. (**b**) Tracking is challenging when the feet of the cat are occluded during walking (top). PPR tracks dense surface points leveraging rigid body dynamics constraints. As a result, the left-rear feet and tail of the cat are correctly tracked in world coordinates despite undergoing heavy self-occlusion (bottom). The contact constraints also effectively reduce infeasible motion such as foot skating. (**c**) The kinematic solution does not often produce foot trajectories in contact with the ground (black horizontal line). Applying ground-fitting (to satisfy joints being above the ground) still produces a human that is floating and titled. PPR jointly optimizes scale, reconstructions, and physics to produce an upright human in contact with the ground.

tion quality on examples including humans and animal videos. To our knowledge, PPR is the first attempt at a generalized, end-to-end framework for *jointly* optimizing dynamic 3D reconstructions and physical systems from monocular videos.

6.2 Related Work

Parametric Body Models. A large body of work in human and animal reconstruction uses parametric models [144, 182, 267, 287, 333, 334], which are built from registered 3D scans of human or animals, and serve to recover 3D shapes given a single image or video at test time [9, 25, 117, 335]. Although parametric body models achieve great success in reconstructing human with large amounts of ground-truth 3D data, it is challenging to apply the same methodology to categories with limited 3D data, such as animals.

Nonrigid Reconstruction from Imagery. Non-rigid structure from motion (NRSfM) methods [29, 74, 121, 123, 230] reconstruct non-rigid 3D shapes from 2D point trajectories in a class-agnostic way. However, due to difficulties in estimating long-range correspondences [216, 243], they do not work well for videos in the wild. Recent works apply differentiable rendering to reconstruct articulated objects from videos [192, 282, 306, 307, 308] or images [73, 108, 118, 130, 131, 282, 316]. However, their recovered 3D configurations are often physically implausible due to the ill-posed nature of the monocular reconstruction problem.

Physics-informed 3D Reconstruction. Prior works have shown that the physical realism of 3D human reconstruction can be improved by differentiable physics simulation [62] and soft physics constraints [204, 228, 267]. However, their methods typically require pre-computed 3D human poses and a template body shape, mass, and skeletons [144, 292], which makes it difficult to generalize to non-human categories. Most of them also require ground plane and foot contact annotations [228, 229]. Recently, DiffPhy [62] optimizes control parameters that generate the motion through a physics simulator that removes the dependency on contact annotations; however, it still relies on the 3D human pose and assumes a fixed camera frame. Beyond the human category, some recent works [105, 164] explore more generic physic priors to regularize shape and cloth deformation.

Differentiable Simulation with Contact. Differentiable contact reasoning in

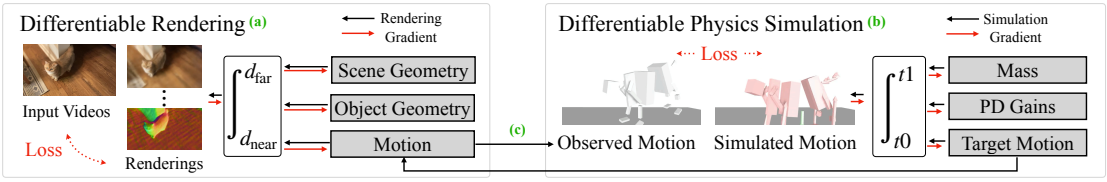

Figure 6.3: **Method overview.** Given monocular videos of an articulated object, PPR solves for a physically plausible representation of the object and the surrounding environment. **(a)** It leverages differentiable volume rendering to optimize the object and the background geometry (Sec. 6.3.1), as well as the object and camera motion parameters (Sec. 6.3.2). **(b)** It leverages a differentiable physics simulator to recover the underlying parameters of an articulated body system, including mass distribution, parameters of a controller, and the target motion tracked by the controller (Sec. 6.3.3). **(c)** We alternate between the differentiable rendering optimization and differentiable physics optimization, such that the reconstructions are consistent with visual observations while satisfying physics constraints (Sec. 6.3.4).

graphics and robotics has seen great advancement in recent years [82, 85, 153, 281, 328]. A crucial challenge for contact simulation and gradient computation arises from its non-smoothness nature. Some methods solve a set of complementarity problems governing contact forces via optimization and derive the gradients [82, 84, 193, 237, 281]. An alternative approach is to soften contact forces by allowing inter-penetration that produces elastic forces to push collided objects away [60, 153]. We leverage the soft contact model that can be easily parallelized on GPUs, and couple differentiable contact physics with differentiable rendering to jointly reason about 3D reconstruction and physics from videos.

6.3 Approach

Given casually-taken videos of an articulated object, we apply differentiable rendering (DR) to solve for a kinematic reconstruction of the object and the surrounding environment that faithfully explains the input videos (Sec. 6.3.1 and Sec. 6.3.2). Meanwhile, we perform differentiable physics (DP) optimization to constrain the kinematic estimation to be physically plausible, such that it can be replayed in a simulator (Sec. 6.3.3). We alternate between DR and DP using coordinate descent until the optimization converges (Sec. 6.3.4). An overview is shown in Fig 6.3.

6.3.1 Object and Scene Model

To enable physics-based modeling of the interactions between the object and the environment from videos, we build a dense 3D representation of their kinematic states. We model the scene $\mathbf{T}$ as the composition of a dynamic object and a rigid background, each of which is represented as neural fields defined in their respective canonical space.

Object Fields $\mathbf{T}^o$. Our canonical model for the object is similar to BANMo [308]. A 3D point $\mathbf{X} \in \mathbb{R}^3$ is associated with three properties: signed distance $d \in \mathbb{R}$, color $\mathbf{c} \in \mathbb{R}^3$, and canonical features $\boldsymbol{\psi} \in \mathbb{R}^{16}$, which are used to register pixel observations to the canonical space. These properties are predicted by multi-layer perceptron (MLP) networks:

$$(d, \mathbf{c}^t) = \mathbf{MLP}_\sigma(\mathbf{X}, \boldsymbol{\omega}_a), \tag{6.1}$$

$$\boldsymbol{\psi} = \mathbf{MLP}_{\boldsymbol{\psi}}(\mathbf{X}). \tag{6.2}$$

Besides the 3D point locations, we further condition the color on an appearance code $\boldsymbol{\omega}_a \in \mathbb{R}^{64}$ that captures frame-specific appearance such as shadows and illumination changes [157]. To capture articulations and soft deformations, we use a deformable variant of NeRF [308]. For a given time t, it defines a forward warping field $\mathcal{W}^{t,\rightarrow}$ that transforms 3D points from the canonical space to the specified time instance, and a backward warping field $\mathcal{W}^{t,\leftarrow}$ to transform points in the inverse direction. The warping fields are further explained in Sec. 6.3.2.

Scene Fields $\mathbf{T}^s$. We leverage VolSDF [314] to build a high-quality background reconstruction. One crucial modification is that we supervise the background fields with not only RGB, but also optical flow and surface normal estimations. Optical flow supervision acts similarly to direct bundle adjustment in DroidSLAM [250], which effectively optimizes camera parameters and scene geometries in challenging conditions (e.g., low-texture, large camera motion). Surface normal supervision [273, 318] provides a signal to regularize geometry and reconstruct the background when there is little to no motion parallax. Those supervisions are extracted from pre-trained optical flow [249, 301] and surface normal [53] predictors.

Composite Rendering. To render images, we compose the density and color of

the object and the scene fields in the view space [173], and compute the expected color, optical flow and surface normal maps. During optimization, we minimize the difference between the rendered and observed color, flow, and surface normal values.

6.3.2 Motion Representation

The warping function $\mathcal{W}$ models object motion is modeled at three levels: root body movements $\mathbf{G}_o$, skeleton articulations $\mathbf{A} = \{\mathbf{J}, \mathbf{W}, \mathbf{Q}\}$ and soft deformations $\mathbf{S}$.

Global Movement $\{\mathbf{G_b}, \mathbf{G}_o\}$. We model the background-to-camera transformations $\mathbf{G}_b$ and object root-to-camera transformations $\mathbf{G}_o$ as per-frame SE(3) represented as Fourier-based MLP networks.

Skeleton Articulation $\{\mathbf{J}, \mathbf{Q}, \mathbf{W}\}$. The coarse-level motion of the object is controlled by an articulated skeleton model. The skeleton has bones connected by 3-DoF spherical joints, specifically a tree topology with joint locations $\mathbf{J} \in \mathbb{R}^{3\times(B-1)}$ and Gaussian bones of size $\mathbf{L} \in \mathbb{R}^{9\times B}$. We set B=26 for quadrupeds and B=19 for humans. The skeleton topology is fixed through optimization but $\mathbf{J}$ and $\mathbf{L}$ are specialized to input videos. To model time-varying skeletal movements, we define per-frame joint angles:

$$\mathbf{Q} = \mathbf{MLP}_{\mathbf{A}}(\boldsymbol{\theta}) \in \mathbb{R}^{3\times(B-1)}, \tag{6.3}$$

where $\boldsymbol{\theta} \in \mathbb{R}^{16}$ is a low-dimensional articulation code. Given joint angles and the per-video joint locations, we compute bone transformations $\mathbf{G} \in \mathbb{R}^{3\times4\times B}$ in the object root coordinate via forward kinematics [168].

To drive the space deformation with bone articulations, we define the skinning weights as the membership probability of bones assigned to 3D points $\mathbf{X}$ [240, 306],

$$\mathbf{W} = \sigma_{\text{softmax}}\big(d_\sigma(\mathbf{X}, \boldsymbol{\theta}) + \mathbf{MLP}_{\mathbf{W}}(\mathbf{X}, \boldsymbol{\theta})\big) \in \mathbb{R}^{B}, \tag{6.4}$$

where θ is a pose code and $d_\sigma(\mathbf{X}, \boldsymbol{\theta})$ is the Mahalanobis distance between $\mathbf{X}$ and Gaussian bones under pose θ, refined by a delta skinning MLP [308]. Each bone has three parameters for center, orientation, and scale respectively. The orientations are determined by the parent joints, and the centers as well as the scales are optimized.

Then the motion of a spatial point is driven by blending skinning,

$$\mathbf{X}(\theta) = \left(\sum_{b=1}^{B} \mathbf{W}_b \mathbf{G}_b\right) \mathbf{X}. \tag{6.5}$$

Soft Deformation S. To account for the deformation caused by non-skeletal movements (such as the clothes of human), we add a neural deformation field [126, 179] $\mathbf{S}(\cdot)$ that is capable of representing highly nonrigid deformations. We use real-NVP [49] to produce 3D deformation fields that are invertible by construction. The soft deformation is applied to the canonical space similar to Human-NeRF [279],

$$\mathbf{X}(\omega_d) = \mathbf{S}(\mathbf{X}, \omega_d), \tag{6.6}$$

where ω_d is a per-frame soft deformation code. Compared to applying $\mathbf{S}$ to the articulated space, we found this formulation to be easier to optimize since it operates on the fixed canonical space.

Invertibility of Warping Fields. To summarize, both the forward and backward warping fields $\{\mathcal{W}^{t,\rightarrow}, \mathcal{W}^{t,\leftarrow}\}$ include an articulation operation in Eq. (6.3) and a deformation operation in Eq. (6.6). Notably, we only need to define each operation in the forward direction. The deformation operation is invertible by construction. To invert the articulations, we invert the SE(3) transformations $\mathbf{G}$ in the blend skinning equation, and compute the skinning weights with Eq. (6.4) using the corresponding articulation codes. To ensure the articulation fields are self-consistent, we use a 3D cycle loss following prior works [132, 308].

6.3.3 Physics-Informed Reconstruction

We define a differentiable physics simulation module to constrain the scene and object representations.

Coordinate Transforms. To simulate physics, we define a world coordinate system where gravity points in the -y direction and the ground is located at the x-z plane. A point from object space, denoted by $\mathbf{X}$, can be transformed into world space as:

$$\mathbf{X}_w = \mathbf{G}_{o\rightarrow w}\mathbf{X} = \mathbf{G}_{b\rightarrow w}{\mathbf{G}_b}^{-1}\mathbf{G}_o\mathbf{X}, \tag{6.7}$$

where $\mathbf{G}_o$ is the object root to camera transform and $\mathbf{G}_b$ is the background to camera transform, both of which can be estimated with differentiable rendering optimization [308]. $\mathbf{G}_{b\to w}$ is the background to world transform, and we solve it by fitting ground planes to the scene geometry (extracted from density fields by marching cubes [145]) per video.

Relative Scale Ambiguity. Notably, there is scale ambiguity between each independently moving scene element [79, 319], including the objects and the background (Fig. 6.2). For instance, one may reconstruct a regular-sized cat walking on the ground, or a tiny cat floating in the air, such that both interpretations explain the input video. To tackle the relative scale ambiguity, we use physics cues to optimize a scale factor s applied to the background geometry and camera translation, $\mathbf{T}_b^* = s\mathbf{T}_{c\to b}$, such that the total scene can be reconstructed up to a global scale factor.

During the physics optimization, s is updated to enable the simulated ragdoll to follow the reconstructed kinematics under gravity and contact constraints. Specifically, floating objects (which correspond to an overly large background scale) are penalized because they lead to a falling motion that is inconsistent with the kinematic reconstruction. Similarly, ground penetrations (which correspond to an overly small background scale) are also penalized because they lead to an inconsistent "bounce" from ground reaction forces.

Differentiable Ragdoll Simulation. We construct an articulated body dynamics model of a ragdoll using standard Newtonian dynamics [137, 166]:

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{S}\boldsymbol{\tau} + \mathbf{J}_c(\mathbf{q})^T\mathbf{f} - \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}})), \tag{6.8}$$

where $\mathbf{q} = [\mathbf{G}_{o\to w}, \mathbf{Q}] \in \mathbb{R}^{6+3B}$ contains the generalized coordinates of the ragdoll. $\mathbf{G}_{o\to w}$ is the root SE(3) transformation in Eq. (6.7) and $\mathbf{Q} \in \mathbb{R}^{3B}$ are joint angles produced by Eq. (6.3). $\mathbf{M}$ is the generalized mass matrix, $\mathbf{J}_c$ is the contact Jacobian computed by forward kinematics, $\mathbf{f}$ represents contact forces, $\mathbf{c}$ includes Coriolis force and gravity, and $\boldsymbol{\tau} \in \mathbb{R}^{3B}$ represents the joint torque actuation, which is mapped to the generalized coordinates using a selection matrix $\mathbf{S}$ [166]. Intuitively, Eq. (6.8) is the generalization of Newton's "F=MA" for rotating rigid bodies under contact. We *differentiably* simulate ragdoll rigid body dynamics with environmental contact using Warp [153, 294]. Warp performs semi-implicit Euler integration to compute

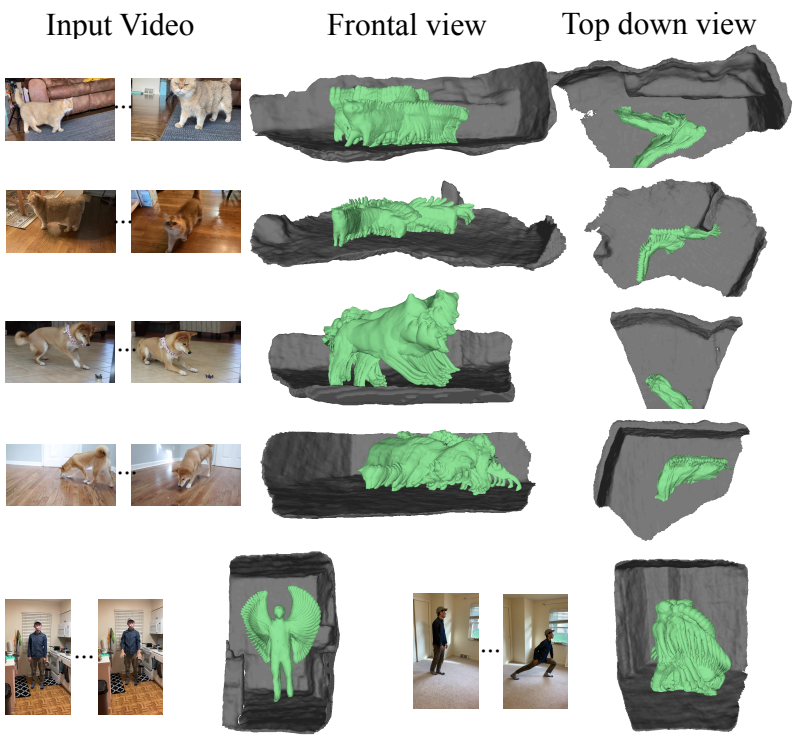

Figure 6.4: **Reconstruction given monocular videos.** We show physically-plausible reconstructions of the articulated shapes (green) and the surrounding environment (gray). Please see the supplement website for video results.

the updated system state $(\mathbf{q}, \dot{\mathbf{q}})$, which is differentaible. To ensure differentiability through contact, Warp uses the frictional contact model that approximates Coulomb friction with a linear step function [66]. Additionally, it incorporates the contact damping force formulation to provide better smoothness in contact dynamics [293]. We refer readers to [294] for details.

Control. Rather than directly optimizing for time-varying joint torque profiles $\boldsymbol{\tau}_t$, we optimize for gain parameters of a Proportional Derivative (PD) Controller [168], and the time-varying target motion. Given target joint angles $\mathbf{Q}^t$, currently simulated joint angles $\mathbf{Q}^s$, and their derivatives at every frame, the PD controller computes joint torques to reach the target:

$$\boldsymbol{\tau}_t = \mathbf{K}_p(\mathbf{Q}^t - \mathbf{Q}^s) + \mathbf{K}_d(\dot{\mathbf{Q}}^t - \dot{\mathbf{Q}}^s), \tag{6.9}$$

where $\mathbf{K}_p \in \mathbb{R}^{3B}$ and $\mathbf{K}_d \in \mathbb{R}^{3B}$ are PD gains. During optimization (described in the

Figure 6.5: **Differentiable physics simulation helps 3D tracking.** We draw trajectories for points on the left-rear foot of the cat over time (yellow lines). Feet undergoing walking motion are difficult to track using visual cues, due to similar textures and occlusion by the other body parts. PPR uses dynamics priors via physics simulation to bias the solution towards avoiding sudden changes of velocity, and tracks the left-rear foot undergoing occlusion.

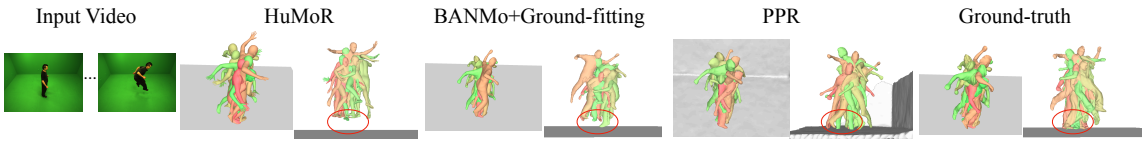

Figure 6.6: **Qualitative comparison on `AMA-bouncing`.** We visualize the reconstructed meshes in world coordinates, colored by their timestamp (red: early; green: late). From the input viewpoint (left), the reconstruction of PPR looks comparable to the baselines. From a novel viewpoint (as if viewing the scene from the right), PPR produces more physically plausible poses. HuMoR [205] reconstructs a floating person with physically-implausible ground contact. BANMo with ground plane fitting (following NeuMan [102]) estimates a rough scale of the person, but produces inaccurate foot contact (note the feet make contact with the ground only once over the video). PPR jointly optimizes 3D pose and relative scale under dynamics and contact constraints, producing accurate foot contact and upright human poses. Please find more visual comparisons in the supplement.

next section), both the gains $(\mathbf{K}_p, \mathbf{K}_d)$ and the targets $(\mathbf{Q}^t, \dot{\mathbf{Q}}^t)$ are updated. $\mathbf{Q}^t$ is initialized as the most recent kinematic reconstruction.

6.3.4 Optimization and Losses

Given monocular videos of an articulated target, we optimize the geometric parameters including the object and scene radiance fields $\mathbf{T}$, kinematic (or motion) parameters $\mathbf{D} = \{\mathbf{G}_o, \mathbf{G}_b, \mathbf{A}, \mathbf{S}\}$, as well as physics parameters $\phi = \{s, \mathbf{M}, \mathbf{K}, \mathbf{Q}^t\}$ as described above. The model is learned by minimizing two types of losses: differentiable rendering losses and differentiable physics losses.

Differentiable Rendering (DR) Losses. Similar to BANMo [308], the DR losses leverage differentiable volume rendering to update both the neural fields $\mathbf{T}$ and the kinematic parameters $\mathbf{D}$. Reconstruction losses are similar to those in existing differentiable rendering pipelines [165, 313], where the goal is the minimize the difference between the rendered images (including object silhouette, color, optical flow, pixel features) and the observed ones:

$$\mathcal{L}_{\text{DR}}(\mathbf{T}, \mathbf{D}) = \mathcal{L}_{\text{sil}} + \mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{OF}} + \mathcal{L}_{\text{feat}} + \mathcal{L}_{\text{Reg}}. \tag{6.10}$$

We refer readers to the supplementary material for regularization terms, and BANMo [308] for the volume rendering equations for each rendered image quantities. To disentangle the object from the background, we use off-the-shelf estimates of object segmentation [115] as supervision to kick-start the optimization. To account for errors in the off-the-shelf segmentation, we set the weight of silhouette term to 0 after several iterations of optimization, while composite rendering of foreground and background itself is capable of disentangling the object and the non-object components by matching the image evidence.

Differentiable Physics (DP) Losses. While image reconstruction losses alone can achieve visually appealing results from the reference viewpoint, the resulting poses can be physically implausible (e.g., Fig. 6.2), particularly for the relative scale and the non-visible body parts. To address this ambiguity, we use a differentiable physics simulator to regularize the solution. The physics term is defined as the difference between the *observed* kinematics $\mathbf{q}$ and a *simulated* trajectory $\mathbf{q}^s$ that is by construction physically-plausible:

$$\begin{aligned} \mathcal{L}_{\text{DP}}(\mathbf{D}, \phi) = \sum_{t=t_0}^{t_0+T} \left\| \mathbf{q}_t(\mathbf{D}) - \mathbf{q}_t^s(\boldsymbol{\phi}) \right\| \quad \text{such that} \\ \mathbf{q}_{t+1}^s = \mathcal{I}\left(\mathbf{q}_t^s, \boldsymbol{\phi}\right). \end{aligned} \tag{6.11}$$

Here, the observed kinematics $\mathbf{q}$ are a function of reconstructed root coordinates in Eq. (6.7) and joint angles in Eq. (6.3), while the simulated trajectory $\mathbf{q}^s$ is a function of physical parameters $\boldsymbol{\phi}$ including scale, body mass and control. Notably, $\mathbf{q}^s$ is *constrained* to be physically plausible since it is the output of a physics simulator $\mathcal{I}$,

which is also *differentiable* and therefore allows one to compute $\frac{\partial \mathcal{I}(\cdot)}{\boldsymbol{\phi}}$.

Coordinate Descent Optimization. In theory, the overall $\text{Loss}(\mathbf{T}, \mathbf{D}, \boldsymbol{\phi}) = \mathcal{L}_{\text{DR}}(\mathbf{T}, \mathbf{D}) + \mathcal{L}_{\text{DP}}(\mathbf{D}, \phi)$ could be directly optimized over all photometric, geometric, and physical parameters [151]. However, differentiable rendering and physical simulation tend to require different sampling strategies [194]. For instance, differentiable rendering is more efficient when sampling pixels uniformly from a dataset to encourage batch diversity, while differentiable physics requires samples of long time intervals that support physical dynamic computations. As a result, joint optimization is rather sample inefficient. Instead, we optimize by repeating the following two steps of coordinate descent:

1. $\min_{\mathbf{T},\mathbf{D}} \text{Loss}(\mathbf{T}, \mathbf{D}, \boldsymbol{\phi})$ [Differentiable Rendering]
2. $\min_{\boldsymbol{\phi}} \text{Loss}(\mathbf{T}, \mathbf{D}, \boldsymbol{\phi})$ [Differentiable Physics]

Step 1 corresponds to a standard differentiable rendering optimization where the kinematics $\mathbf{D}$ are *regularized* to be similar to the most recent simulated trajectory $\mathbf{q}^s(\boldsymbol{\phi})$. Step 2 corresponds to a differentiable physics optimization that solves for physical parameters that closely *targets* (or tracks) the most recent kinematic reconstruction $\mathbf{D}$.

SGD. In practice, each coordinate step is implemented via a fixed number of stochastic gradient descent (SGD) steps, initialized from the values of the variables in the previous descent cycle. Specifically, Step 1 constructs a batch of $N_{px} = 4096$ random pixels for optimization, performing $\mathcal{N}_{\text{DR}} = 10$ iterations of SGD. Because the reconstructed kinematics will be heavily regularized to lie close to the most recent physically-plausible trajectory $\mathbf{q}^s(\boldsymbol{\phi})$, we found faster convergence by initializing $\mathbf{D}$ to $\arg\min_{\mathbf{D}} \mathcal{L}_{\text{DP}}(\mathbf{D}, \boldsymbol{\phi})$, rather than its value from the previous DR cycle. Finally, for SGD optimization of Step 2, we construct random batches of $N_{int} = 30$ random *time intervals* of length 2.4 seconds, performing $\mathcal{N}_{\text{DP}} = 10$ iterations of SGD per cycle. In practice, we perform 50-100 cycles of coordinate descents.

Comparison to Prior Methods. PPR differs from prior work [62, 63, 204, 228] in several ways. First, prior art can be conceptualized as one cycle of coordinate descent, by producing a single kinematic reconstruction (Step 1) to which one fits a ragdoll simulation (Step 2). Second, the simulator often solves for forces assuming known physical parameters (such as generalized mass $\mathbf{M}$ and control parameters

K), while PPR optimizes over such parameters. We ablate such design choices in our experiments. Intuitively, multiple descent cycles allows for different kinematic reconstructions, rather than a single (potentially inaccurate) point estimate of geometry and kinematics. Finally, from a control perspective, one can view Step 2 as an instance of model-predictive control (MPC) with stochastic batch sampling over small time intervals [190].

6.4 Experiments

Dataset. We test PPR on the articulated-mesh-animation (`AMA`) dataset, the `casual-videos` dataset from BANMo, and a newly collected `RGBD-pet` dataset.
`AMA` contains videos of human performance with 3D mesh ground-truth, and we use it to evaluate surface reconstruction accuracy. To test our method, we select 4 videos of `samba` and 4 videos of `bouncing`. Although the videos are calibrated multi-view captures, we treat them as monocular videos and *do not* use the time-synchronization or camera extrinsic parameters.
`Casual-videos` includes monocular videos of the same instance captured from different viewpoints, locations, and times. It contains 11 videos of a cat, 11 videos of a dog, and 10 videos of a human. We manually annotate per-frame binary foot contact labels, and use them to evaluate contact reconstruction accuracy.
`RGBD-pet` dataset contains videos of a cat and a dog, captured by an iPad with RGBD sensor. We use it to evaluate scene reconstruction performance (please see supplement).
Implementation Details. Our differentiable rendering pipeline is implemented with Pytorch. The object neural fields follow BANMo [308], and the background neural fields follow VolSDF [314]. We modify neural blend skinning of BANMo to represent skeletal deformation, and follow CaDeX [126] to represent soft deformation as invertible 3D flow fields. We use Warp [153] for differentiable physics simulation. It implements articulated body dynamics as well as contact physics, and integrates kinematics over time using the semi-implicit Euler scheme. The step size of the simulator is set to $5e^{-4}$s. The simulation operations are automatically differentiated and parallelized at CUDA kernel level. We write wrappers to pass gradients between Warp and Pytorch.

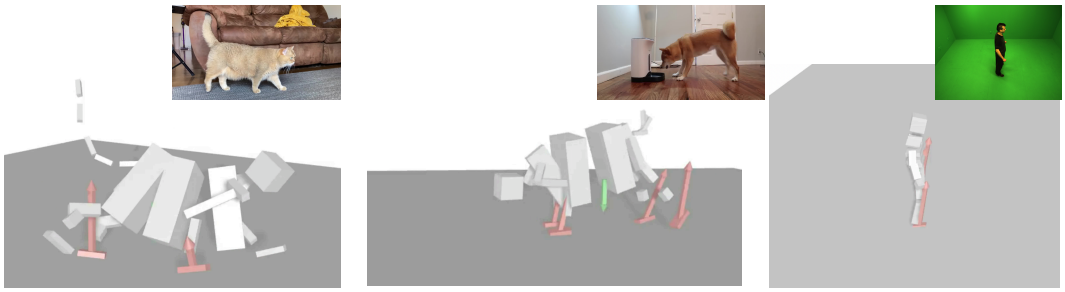

Figure 6.7: **Visualization of ground reaction force.** We show the ground contact returned by the simulator. The body part in contact with the ground is colored in red and the ground reaction forces are marked with red arrows. Gravity is represented by a green arrow. Note the contact modes are aligned with the image evidence.

Hyper-parameters. We use AdamW optimizer and optimize the model for 36k iterations (taking around 18 hours on 2 NVIDIA GeForce RTX 3090 GPUs). The weights of loss terms are tuned to have similar initial magnitudes. We first pre-train a background field [318], and jointly optimize an object field with differentiable composite rendering [173]. The object root poses are initialized with a viewpoint network following BANMo.

Extracting Registered Meshes. To evaluate surface reconstruction accuracy, we extract the canonical mesh by finding the zero-level set of SDF with running marching cubes on a 256^3 grid. To get the shape at a specific time instance, the canonical mesh is forward warped with $\mathcal{W}^{t,\rightarrow}$, which defines an articulation and deformation operation.

6.4.1 Surface Reconstruction

Metrics. To evaluate the reconstruction quality, we report both Chamfer distance and F-scores. Chamfer distance computes the average distance between the ground-truth and the estimated surface points by finding the nearest neighbour matches, but it is sensitive to outliers. F-score at distance thresholds $d \in \{5\text{cm}, 2\text{cm}\}$ [248] provides a more informative quantification of surface reconstruction error at different granularity. To account for the scale ambiguity, we fit a per-video scale factor by aligning the predicted mesh with the ground-truth in the view space.

Table 6.1: **Surface reconstruction evaluation on `AMA`.** 3D Chamfer distance (cm, ↓) and F-score (%, ↑) are averaged over all the frames. The best results are in bold. We align the reconstructed meshes with the ground-truth meshes by a per-sequence scale factor and SE(3) transformation. PPR outperforms HuMoR and BANMo in all metrics. Replacing BANMo's control point deformation with our skeleton deformation significantly improves results on `samba` but made results worse on `bouncing`. Further enforcing dynamics and contact constraints via differentiable physics simulation (PPR-Ours) significantly improves results on both sequences.

Method	`samba`			`bouncing`		
	CD	F@5cm	F@2cm	CD	F@5cm	F@2cm
HuMoR	10.5	65.3	31.7	14.8	49.0	18.9
BANMo	11.8	56.7	27.2	12.8	56.0	25.6
+skel	8.9	68.1	32.0	14.1	51.3	23.9
PPR-Ours	**8.3**	**73.4**	**35.4**	**9.1**	**68.3**	**32.8**

Baselines. We compare against HuMoR and BANMo for human reconstruction. HuMoR [205] learns human motion priors (in the world coordinate) from large-scale motion capture datasets. Given an input video, it performs test-time optimization leveraging OpenPose keypoint detections and the learned humor motion priors. Processing a 170 frame video takes around 2 hours on a Titan-X machine. BANMo [308] is a template-free method for video-based deformable shape reconstruction. It relies on differentiable rendering optimization given optical flow correspondence and DensePose features [171]. Running BANMo on around 1k frames takes 10 hours on two V100 GPUs.

Results. We show qualitative results in Fig. 6.6, and quantitative results in Tab. 6.1. HuMoR produces accurate and consistent reconstructions for videos with common motion (such as the samba sequence). However, human motion prior fails for certain athletic movements (such as the bouncing sequence) due to the lack of those motions in the human MoCap dataset. BANMo reconstructions look reasonable from the reference viewpoint, but the invisible body parts often appear distorted or tilted from a novel viewpoint due to the fundamental depth ambiguity. In contrast, PPR's differentiable rendering module alone improves the reconstruction (CD: 8.9% vs 11.8% for samba) by constraining the body deformation with a skeleton. Our physics-

informed optimization further improves the reconstruction (CD: 8.3% vs 8.9% for samba) by inferring root pose and body motions that satisfy contact and dynamics constraints in a differentiable physics simulator.

6.4.2 Contact Estimation

Evaluation Protocol. To evaluate the physical plausibility of the reconstructions, we follow DiffPhy [62] to design contact metrics, including contact accuracy and foot skate. The contact accuracy is defined as the F-score of contact state estimation averaged over all frames. We further measure the amount of foot skate as the average distance feet move over adjacent contact frames, frames that are marked as in contact with the ground according to the ground-truth. To predict contact state, we mark a foot to be in contact with the ground if its distance to the ground is smaller than 5% of the body height.

Results. We show quantitative evaluation in Tab. 6.2. Our method with physics-informed optimization achieves the highest accuracy in contact estimation. BANMo with ground fitting solves a rough relative scale between the background and the object, and we found it to produce worse results than PPR (Contact: 68.6 v 93.1 for casual-cat). We posit that scale fitting suffers from inaccurate kinematic reconstructions, while PPR jointly improves both via differentiable physics simulation. In terms of foot skate, although PPR works better than BANMo, it still produces more foot skate than HuMoR, especially for the highly-dynamic bouncing sequence. Enforcing a stronger physics prior for PPR (e.g., simulating longer time intervals) may produce less foot skates. However, doing so makes the motion more challenging to track by a controller. Besides contact estimation, PPR also produces plausible ground reaction force estimation as shown in Fig. 6.7.

6.4.3 Ablation Study

We ablate design choices and show the results in Tab. 6.3.

Ground Prior vs Differentiable Physics. One commonly used approach to determine the object scale is to force the reconstructions to be above the ground plane except for a supporting region touching the ground [102]. However, this is sensitive to the accuracy of the visual reconstruction, often resulting inaccurate

Table 6.2: **Foot contact estimation on `casual-cat` and `AMA`.** Foot contact F-score (%, ↑) and skating (cm, ↓) are averaged over all frames. The best results are in bold. *To compare with BANMo that only produces camera-space reconstructions, we take a step further to reconstruct the background and use ground prior to find the scale of camera motion following NeuMan [102], and decouple it from BANMo results to produce world-space reconstructions. By enforcing physics constraints, PPR-Ours outperforms the baselines in contact estimation. It produces more foot skate than HuMoR, but less foot skate than BANMo.

Method	`casual-cat`		`samba`		`bouncing`	
	Contact	Skate	Contact	Skate	Contact	Skate
HuMoR	N.A.		44.6	**0.4**	54.8	**2.6**
BANMo*	68.6	2.8	24.5	1.6	1.3	8.3
PPR-Ours	**93.1**	**2.1**	**67.4**	1.2	**85.4**	7.4

ground contact with tilted body poses, as illustrated in Fig. 6.2 (c). As a result, replacing differentiable physics simulation with ground prior makes contact estimation accuracy much worse (67.4% vs 26.3%).

One-cycle vs Coordinate Descent Optimization. Instead of alternating between visual reconstruction and physics-informed optimization, existing works [62, 63, 204, 228] only complete one cycle by first estimating the shape and motion (using DR or feed-forward networks) and then optimizing physics (using DP or trajectory optimization). Compared to using coordinate descent, the reconstruction error of one-cycle optimization significantly increases (8.3cm vs 46.0cm). In terms of contact metrics, the foot contact accuracy dropped (62.3% vs 67.4%), although the skating metric becomes slightly better. This validates the effectiveness of joint vision-physics optimization via coordinate descent: alternating between visual reconstruction and physics-informed optimization improves reconstruction quality while making the solution physically plausible.

Feedback vs Open-loop Control. We ablate the necessity of using feedback control (specifically PD control in Sec. 6.3.3) during differentiable simulation, as some existing works [85, 98] directly optimize open-loop control without position and velocity feedback. We find that open-loop control finds a hard time tracking the target kinematics obtained from DR, and decreases both the contact accuracy and

Table 6.3: **Ablation study on `AMA-samba`.** Best results are in bold. Please see Sec. 6.4.3 for a detailed discussion.

Method	Contact (%, ↑)	Skate (cm, ↓)	CD (cm, ↓)
Full method	**67.4**	1.2	**8.3**
Phys→ground	26.3	1.3	8.9
One-cycle	62.3	**1.1**	46.0
PD→ open-loop	53.6	2.7	12.7
Freeze K/M	50.4	1.2	9.0

reconstruction quality. The gain of moving from open-loop to feedback control indicates that incorporating prior knowledge in controller design improves reconstruction results and physical plausibility.

Optimizing Mass and PD Gains. We further investigate the effect of optimizing the mass of each body part, as well as the PD gains for each joint of the ragdoll. We found freezing **K** and **M** decreases contact estimation accuracy and reconstruction quality (even worse than without DP). This suggests jointly inferring the internal parameters of the ragdoll is important for physics-informed optimization.

Part IV

Perceiving Dynamic Scenes

Chapter 7

3D Scene Flow Estimation through Optical Expansion

7.1 Introduction

Estimating 3D motion is crucial for autonomous robots to move safely in a dynamic world. For example, collision avoidance and motion planning in a dynamic requirement hinge on such inferences [34, 154, 155, 167]. Many robotic platforms make use of stereo cameras or time-of-flight sensors for which metric distances are accessible. Here, 3D motion can be determined by searching for correspondence over frames, or registration between 3D point clouds. Such active sensing and fixed-baseline stereo methods struggle to capture far-away objects due to limited baselines and sparse sensor readings. In this work, we analyze the problem of dynamic 3D perception with monocular cameras, which do not suffer from baselines or sparse readings.

Challenges However, estimating 3D motion from monocular cameras is fundamentally ill-posed without making assumptions about the scene rigidity: given a particular 2D flow vector, there is an infinite pair of 3D points along two degrees of freedom (obtained by back-projecting two rays for the source and target pixel - see Fig. 7.3) that project to the same 2D flow. Intuitively, a close-by object that moves slowly will generate the same 2D flow as a far-away object that moves fast.

Prior work Nevertheless, there have been numerous attempts at monocular dynamic scene reconstruction using multi-body SfM and non-rigid SfM [124, 332]. A recent approach [30] attempts to solve the monocular scene flow problem in its generality.

Figure 7.1: Optical flow vs optical expansion. From left to right: overlaid two consecutive frames, color-coded optical flow fields and optical expansion map, where white indicates larger expansion or motion towards the camera. Notice it is difficult to directly read the 3D motion of the hawk from optical flow. However, it is easy to tell the hawk is approaching the caemra from optical expansion.

Because such tasks are under-constrained, these methods need to rely on strong prior assumptions, either in the form of prior 3D geometry (typically learned from data-driven scenes) or prior 3D motion (typically rigid-body priors) that are difficult to apply to "in-the-wild" footage. Instead, we derive a simple but direct geometric relationship between 3D motion and 2D correspondence that allows us to extract up-to-scale 3D motion.

Why optical expansion? Human perception informs us that change in the perceived size of an object is an important cue to determine its motion in depth [224, 244]. Indeed, optical expansion is also a well-known cue for biological navigation, time-to-contact prediction, and looming estimation [70]. Inspired by these observations, we propose to augment 2D optical flow measurements with 2D optical expansion measurements: for each pixel in a reference frame, we estimate both a 2D offset and a relative scale change (u, v, s), as shown in Fig.7.2. We show that such measurements can be robustly extracted from an image pair, and importantly, resolve *half* of the fundamental ambiguity in 3D motion estimation. Because optical expansion is a local pixelwise measurement, we demonstrate that it can be easily incorporated into existing approaches for self-learning of optical flow, increasing the accuracy.

3D motion from expansion Specifically, under a scaled orthographic camera projection model, optical expansion is directly equivalent to the motion-in-depth of the non-rotating scene element that projects to the corresponding source and target pixel. This eliminates one degree-of-freedom. When combined with camera intrinsics and optical flow, optical expansion reveals the true direction of the 3D motion, but

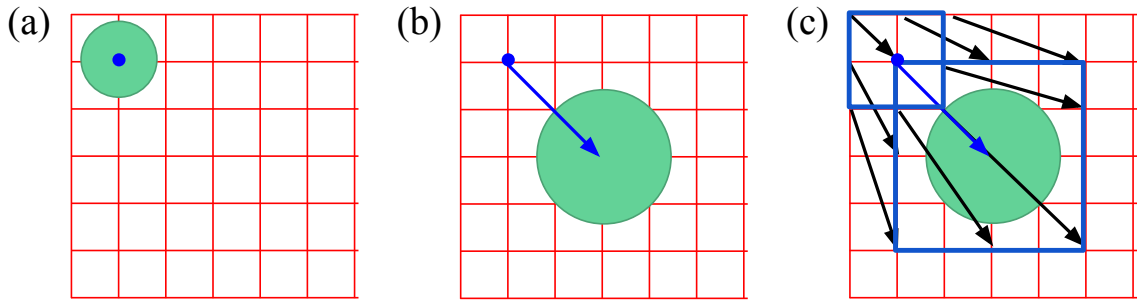

Figure 7.2: Optical expansion vs optical flow. (a): reference image, where we are interested in the position and scale change of the pixel marked as blue; (b): optical flow providing the position change (u, v); (c): optical expansion providing the scale change s. Intuitively, optical expansion can be measured as the square root of the ratio between the two areas covered by the rectangles.

not its magnitude. Fig. 7.3 demonstrates that we now know if an object is moving closer toward or away from the camera, but there is still an overall scale ambiguity, which can also be resolved by specifying the depth of *one* of the point pairs along its back-projected ray.

Method To estimate per-pixel optical expansion, we propose an architecture based on local affine transformations. The relative scale ground-truth is obtained from the existing optical flow and 3D scene flow training datasets. We also present a self-supervised approach to learn optical expansion from photometric information of the input images.

Contributions We summarize our contribution as follows. (1) We theoretically derive the effectiveness of optical expansion to reduce the ambiguities inherent to monocular scene flow. (2) We propose a neural architecture for normalized scene flow estimation that encodes strong geometry knowledge and leads to better interpretability and generalization. (3) We demonstrate the effectiveness of optical expansion across a variety of benchmark tasks, establishing new SOTA results for optical scene flow, LiDAR scene flow, time-to-collision estimation - while being significantly faster than prior methods, and improving results for self-supervised optical flow. (4) We apply dense optical expansion to two-frame depth estimation and show improvements over triangulation-based methods in numerically-unstable regions near the epipole.

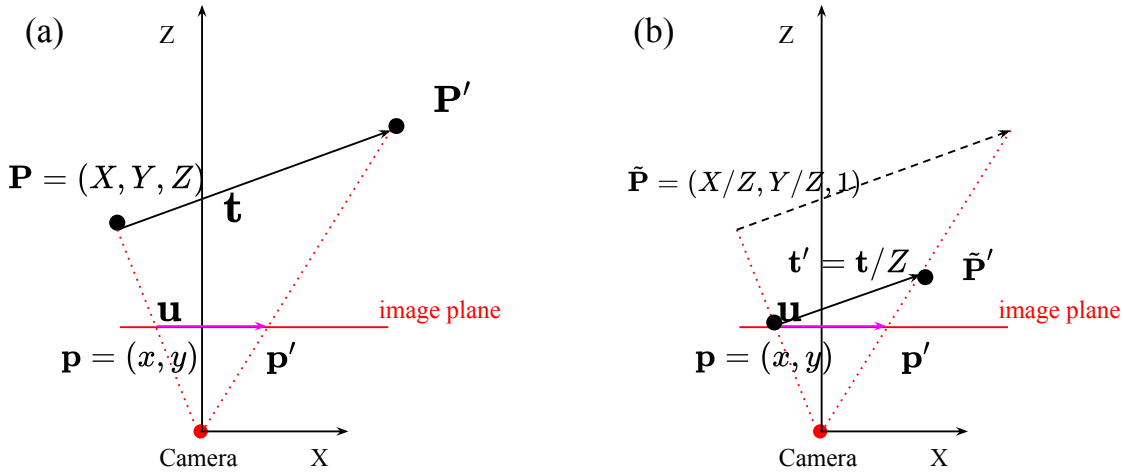

Figure 7.3: (a): Projection from scene flow $\mathbf{t}$ to optical flow $\mathbf{u}$. (b): Projection from scene flow $\mathbf{t}$ to normalized scene flow $\hat{\mathbf{t}} = \mathbf{t}/Z$. Normalized scene flow is a 3-dimensional vector that extends standard optical flow to capture changes in depth. Notice when projecting a 4DoF scene flow vector to the image plane (given the reference 2D point), 2DoF can be recoverd by optical flow and 2DoF are missing: 1) the depth Z, which is not recoverable; 2) and the motion-in-depth, which can be recovered from optical expansion.

7.2 Related Work

Visual correspondence Visual correspondence dates back to the early work in human visual perception and 3D reconstruction, where it was found point correspondence needs to be solved to perceive depth and 3D structures can be recovered from their projected points [156, 261]. Affine correspondence defines a 2×2 affine transform between the neighborhood of point correspondences, to encodes higher-order information about the scene geometry [13, 14, 200]. Similar to affine correspondence, we extract local information of point correspondences to encode rich geometric information about motion-in-depth, but not rotation or shear.

Scale estimation The concept of scale changes of visual features is well studied in the context of feature descriptor and matching [16, 147, 188] as well as dense optical flow [196, 275, 295]. In these approaches, scale is often treated as a discrete auxiliary variable for producing better descriptors and feature matches, but not estimated as a continuous quantity at a fine scale. Some other approaches either directly estimate the intrinsic scales by Laplacian filtering [169] or compute the scale changes from

the divergence of optic flow fields [35, 277], but give sub-accurate results. Instead, our method produces continuous dense optical expansion reliably in a data-driven fashion. Moreover, the relationship between relative scale and depth changes has been explored for 3D reconstruction [55, 191, 217, 320] as well as collision avoidance in robotics [83, 154, 167]. However, prior methods often focus on object-level scale changes and sparse interest points. Our work extends the concept of relative scale and depth changes to the dense, low-level correspondence tasks of 3D scene flow estimation.

Monocular dynamic reconstruction Prior work on monocular 3D motion estimation casts the task as a sub-problem of monocular scene reconstruction, attempting to jointly recover both motion and depth [124, 197, 212, 264]. Because of the ill-posed nature of this problem, they either rely on strong motion priors such as multi-rigid body [197, 264] and as rigid as possible [124], or strong shape priors such as low rank and union-of-subspaces [74, 332]. Those handcrafted priors hallucinate good reconstructions when their assumptions are met, but in other cases not applicable. On the other hand, when scene elements are piece-wise rigid, we can reconstruct up-to-scale local motions with planar homographies. However, homography estimation is sensitive to noise in 2D correspondences, requiring the use of strong priors to regularize the problem [124]. In this work, we propose a simpler representation of local motion, e.g. optical expansion, which can be reliably estimated from real-world imagery because fewer degrees of freedom are needed to be inferred.

7.3 Approach

In this section, we first establish the relationship between optical expansion and motion-in-depth under scaled orthographic projection model. Then we derive a direct relationship between motion-in-depth, normalized 3D motion, and scene flow. Finally we propose a neural architecture of learning optical expansion and normalized 3D flow.

7.3.1 Optical expansion

Here we explicitly derive the relationship between optical expansion, which describes the change of the perceived size of objects, and motion-in-depth. We begin

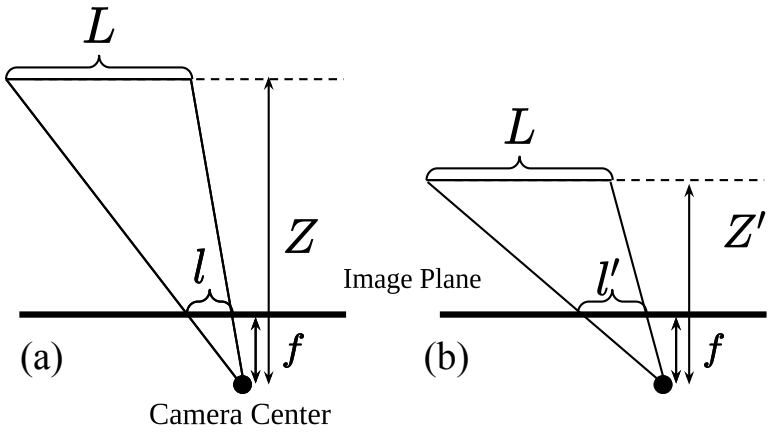

Figure 7.4: We visualize a moving object under scaled orthographic projection across two timesteps (a) and (b). Given a definition of optical expansion $s = l'/l$ and motion-in-depth $\tau = Z'/Z$, Eq. 7.1 derives that $s = 1/\tau$.

with a simple pinhole camera model that projects a 3D point $P = (X, Y, Z)$ into an image position (x, y):

$$\mathbf{p} = (x, y) = \frac{f}{Z}(X, Y),$$

where f is the focal length. Under a *scaled orthographic camera* model, the projection of all points on an object can be computed with an orthographic projection onto a fronto-parallel plane followed by a perspective projection of the plane [79]. Such an approximation is reasonable if the variation in depth of an object is small compared to its distance from the camera. With the physical length of an object being L and its orientation σ defined as the angle between the surface normal and the camera z-axis, the projected length of the object is then given by

$$l = \frac{f\bar{L}}{Z} = \frac{fL\cos\sigma}{Z},$$

where $\bar{L} = L\cos\sigma$ accounts for the foreshortening. Assume the scene is locally rigid and one of the rigid pieces changes its depth across two frames from Z to Z', while keeping its physical size L and orientation σ unchanged. Define the optical expansion s to be the ratio of its projected lengths l'/l, and define the motion-in-depth τ to be the ratio of depths Z'/Z. We can now derive that $s = 1/\tau$ assuming *1) a scaled*

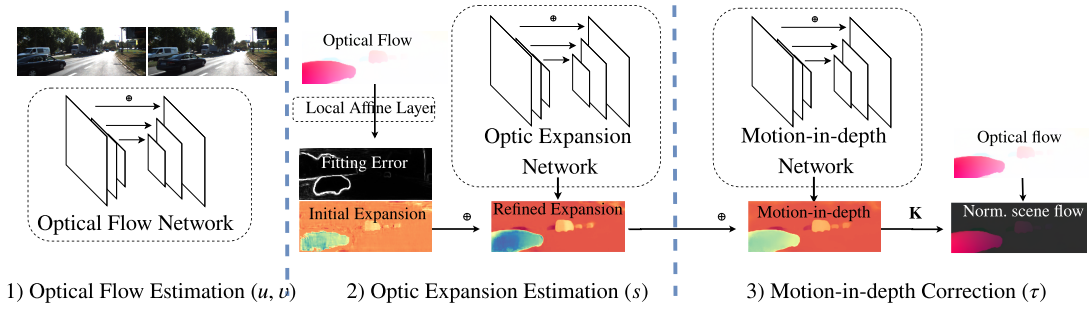

Figure 7.5: Network architecture for estimating normalized scene flow. 1) Given two consecutive images, we first predict dense optical flow fields using an existing flow network. 2) Then we estimate the initial optical expansion with a local affine transform layer, which is refined by a U-Net architecture taking affine fitting error and image appearance features as guidance [113]. 3) To correct for errors from the scaled-orthographic projection and rotation assumptions, we predict the difference between optical expansion and motion-in-depth with another U-Net. Finally, a dense normalized scene flow field is computed using Eq. 7.2 by combining (u, v, τ) with camera intrinsics $\mathbf{K}$.

orthographic camera model and *2) the scene elements are not rotating relative to the camera* (Fig. 7.4):

$$l = \frac{f\bar{L}}{Z}, l' = \frac{f\bar{L}}{Z'} \quad \Rightarrow \quad s = \frac{l'}{l} = \frac{Z}{Z'} = \frac{1}{\tau} \tag{7.1}$$

7.3.2 Normalized scene flow

In the last section, we showed that motion-in-depth τ can be computed from optical expansion s for a scaled orthographic camera model. In this section, we show that motion-in-depth τ can be combined with camera intrinsics K to compute a normalized 3D scene flow vector.

Given camera intrinsics $\mathbf{K}$, for a 3D point changing its position from $\mathbf{P}$ to $\mathbf{P}'$, we have

$$\mathbf{P} = \lambda \mathbf{K}^{-1}\tilde{\mathbf{p}}, \quad \mathbf{P}' = \lambda' \mathbf{K}^{-1}\tilde{\mathbf{p}}',$$

where $\tilde{\mathbf{p}}$ and $\tilde{\mathbf{p}}'$ are homogeneous 2D coordinates with the last coordinate being 1,

and λ, λ' are scale factors. Because the last row of the intrinsic matrix $\mathbf{K}$ is (0,0,1), scale factors are directly equal to the depth of each point: $\lambda = Z$ and $\lambda' = Z'$.

Following prior work [161], we model scene flow as 3D motion vectors relative to the camera, which factorizes out the camera motion. The scene flow $\mathbf{t}$ is then computed as:

$$\begin{aligned}
\mathbf{t} &= \mathbf{P}' - \mathbf{P} \\
&= \mathbf{K}^{-1}(Z'\tilde{\mathbf{p}}' - Z\tilde{\mathbf{p}}) \\
&= Z\mathbf{K}^{-1}\Big[\tau(\tilde{\mathbf{u}} + \tilde{\mathbf{p}}) - \tilde{\mathbf{p}}\Big] \quad \text{where} \quad \tilde{\mathbf{u}} = \tilde{\mathbf{p}}' - \tilde{\mathbf{p}} \\
&= Z\mathbf{K}^{-1}\Big[(\tau - 1)\tilde{\mathbf{p}} + \tau\tilde{\mathbf{u}}\Big] \\
&= Z\hat{\mathbf{t}} \quad \text{where} \quad \hat{\mathbf{t}} = \mathbf{K}^{-1}\Big[(\tau - 1)\tilde{\mathbf{p}} + \tau\tilde{\mathbf{u}}\Big]
\end{aligned} \tag{7.2}$$

We denote $\hat{\mathbf{t}}$ as the "normalized scene flow", which is a vector pointing in the direction of the true 3D scene flow. It can be "upgraded" from 2D flow $\tilde{\mathbf{u}}$ knowing motion-in-depth τ and camera intrinsics $\mathbf{K}$. When augmented with the true depth of the point in either frame Z or Z' (following an analogous derivation to the above), normalized scene flow can be further "upgraded" to the true 3D scene flow.

7.3.3 Learning normalized scene flow

In this section we introduce a network architecture for optical expansion and normalized scene flow estimation, and describe ways of learning optical expansion, either in a supervised fashion, or with self-supervised learning.

Network We separate the task of estimating normalized scene flow into three sequential steps: (1) optical flow estimation, where the (u, v) component is predicted from an image pair, (2) optical expansion estimation, where the optical expansion component s is estimated conditioned on the optical flow, and (3) motion-in-depth estimation τ, where the optical expansion is refined to produce correct outputs for a full perspective camera model. Finally, normalized scene flow can be computed given camera intrinsics. We design an end-to-end-trainable architecture for the above steps, as shown in Fig. 7.5. An ablation study in Sec. 7.5 discusses different design choices that affect the performance.

Local affine layer To extract dense optical expansion over two frames, we propose a local affine layer that directly computes the expansion of local 3x3 patches over two frames, as described in the three following steps:

1) Fit local affine motion models. Given a dense optical flow field $\mathbf{u}$ over a reference frame and a target frame, we fit a local affine transformation $\mathbf{A} \in \mathbb{R}^{2\times2}$ [13] for each pixel $\mathbf{x}_c = (x_c, y_c)$ over its 3x3 neighborhood $\mathcal{N}(\mathbf{x}_c)$ in the reference image by solving the following linear system:

$$(\mathbf{x}' - \mathbf{x}'_c) = \mathbf{A}(\mathbf{x} - \mathbf{x}_c), \quad \mathbf{x} \in \mathcal{N}(\mathbf{x}_c), \tag{7.3}$$

where $\mathbf{x}' = \mathbf{x} + \mathbf{u}(\mathbf{x})$ is the correspondence of $\mathbf{x}$.

2) Extract expansion. We compute optical expansion of a pixel as the ratio of the areas between the deformed vs original 3x3 grid: $s = \sqrt{|det(\mathbf{A})|}$.

3) Compute fitting errors. We compute the residual L_2 error of the least-squares fit from Eq. 7.3 (indicating the confidence of the affine fit) and pass this in as an additional channel to the optical refinement network.

Crucially, we implement the above steps as *dense, pixel-wise, and differential* computations as Pytorch layers that efficiently run on a GPU with negligible compute overhead.

Learning expansion (supervised) To train the optical expansion network that predicts s, one challenge is to construct the optical expansion ground-truth. The common solution of searching over a multi-scale image pyramid is infeasible because it gives sparse and inaccurate results. Instead, we extract expansion from the local patches of optic flow fields [35, 277]. Specifically, for each pixel with optical flow ground-truth, we fit an affine transform over its 7x7 neighborhood and extract the scale component, similar to the local affine layer. Pixels with a high fitting error are discarded. In practice, we found optical expansion ground-truth can be reliably computed for training, given the high-quality optical flow datasets available [11, 33, 50, 120, 158, 161].

Learning expansion (self-supervised) Since real world ground-truth data of optical flow are costly to obtain, here we describe a self-supervised alternative of learning the expansion network. Previous work on self-supervised optical flow [139, 159, 206] obtain supervision from photometric consistency, where losses are computed

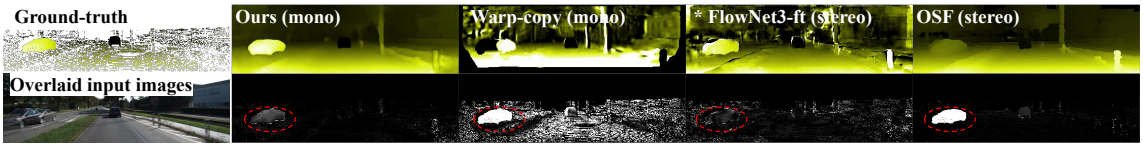

Figure 7.6: Results for image "000105" in the KITTI val set. Top: Motion-in-depth between two frames where bright indicates points moving towards the camera; bottom: error maps of motion-in-depth. Our method predicts more accurate motion-in-depth than the baselines.

by comparing the difference between the intensity values of either the reference and target pixel, or a $K \times K$ patch around the reference pixel and their correspondences. In both cases, the motion of pixels is not explicitly constrained. Our key distinction is to use the predicted optical expansion to expand or contract the reference patches when constructing the loss. The benefits are two-fold: for one, it extracts the supervision signal to train the optical expansion model; for another it puts explicit constraints to the local motion patterns of optical flow, and thus guides the learning.

Learning motion-in-depth To train the motion-in-depth network that predicts τ, we use existing 3D scene flow datasets, from which the ground-truth motion-in-depth can be computed as the ratio between the depth of corresponding points over two frames,

$$\tau^*(\mathbf{x}) = \frac{Z'^*(\mathbf{x} + \mathbf{u}^*(\mathbf{x}))}{Z^*(\mathbf{x})},$$

where Z^* and Z'^* are the ground-truth depth in the reference and target frame respectively, and $\mathbf{u}^*$ is the ground-truth optical flow.

Losses We empirically find that supervised learning of optical expansion yields better performance than self-supervised learning (245 vs 336 in log-L_1 error, as in Tab. 7.5 and Tab. 7.6), and therefore supervised learning is used throughout experiments in Sec. 4.1-4.3. Here, multi-task L_1 losses are used to train the optical expansion (s) and motion-in-depth (τ) networks jointly:

$$L = \sum_{\mathbf{x}} |\sigma_s(\mathbf{x}) - \log s^*(\mathbf{x})| + |\sigma_\tau(\mathbf{x}) - \log \tau^*(\mathbf{x})|,$$

where σ_s and σ_τ are the predicted log-scale expansion and motion-in-depth, and s^*

and τ^* are ground-truth labels. The loss is summed over pixels with valid labels. We experimented with stagewise training, but found joint end-to-end training simpler while performant.

7.4 Experiments

We first evaluate our method on 3D scene perception tasks including optical scene flow, LiDAR scene flow and time-to-collision estimation. We then show results of self-supervised training of optical expansion models. Finally, we conclude with qualitative results that apply optical expansion for rigid depth estimation during forward or backward translational camera movements that are difficult for traditional structure-from-motion.

Setup We freeze the pre-trained optical flow network [301] and train our normalized scene flow model on Driving, Monkaa, and KITTI-15 [158, 161]. For KITTI, we split the original 200 images with ground-truth into train and validation sets. Specifically, for optical scene flow, we select every 5 images for validation, and add the rest 160 images for training; while for LiDAR scene flow, we follow the split of MeteorNet [142] and use the first 100 out of 142 images with LiDAR point clouds for training, and the rest for validation. We follow a two-stage training protocol [88], and empirically choose a larger learning rate of 0.01. Pre-training on Driving and Monkaa takes 60k iterations, and fine-tuning on KITTI takes 30k iterations.

7.4.1 Optical scene flow

We first compare to baselines on KITTI-15 validation set, where the standard metrics of scene flow and error of log motion-in-depth (MiD) are used [161].

Our solution Since our expansion network only provides motion-in-depth over two frames, to generate the full scene flow vector, we use an off-the-shelf monocular depth estimation network MonoDepth2 [72] to predict d_1, which is the disparity of frame one. To predict d_2, the disparity of frame one pixels that have moved to frame two, we simply divide d_1 by the predicted motion-in-depth.

Validation performance To compute d_2, Schuster et al. [225] warp the second-frame disparity maps to the first frame using forward flow, without dealing with occlusions; we consider a stronger baseline that further copies the disparity of out-of-frame pixels

Table 7.1: Scene flow estimation on KITTI-15 validation set. D1, D2, Fl, and SF measure the percentage error of disparity, optical flow and overall scene flow prediction respectively. MiD measures the L1 error of log motion-in-depth $\log(\boldsymbol{D}_2/\boldsymbol{D}_1)$ scaled by 10,000X. Monocular methods are listed at the top, while stereo-based methods are listed below. Methods with † use validation data to train. Our method outperforms the monocular baselines by a large margin, and beats the stereo baselines in terms of MiD.

Method	D1	Fl	D2	SF	MiD
Delta [90]	14.51	6.00	78.87	83.26	2237
Warp+copy [225]	14.51	6.00	27.73	31.16	623
Ours	14.51	6.00	**16.71**	**19.65**	**75**
FlowNet3 [90]	6.95	32.41	20.89	37.09	537
†FlowNet3-ft [90]	1.51	7.36	4.70	9.60	217
PRSM [268]	4.05	8.32	7.52	10.36	124
OSF [162]	3.98	8.95	7.69	10.16	115

from the first frame, denoted by "Warp+copy". Following FlowNet3 [90], we also train a refinement network to hallucinate the disparities of the regions occluded in the second frame. As for monocular scene flow, shown in the first group of Tab. 7.1, our method outperforms baselines by a large margin.

We further consider baselines using stereo cameras to estimate the metric depth at both frames: PRSM [268] and OSF [162] are stereo-based methods that break down the image into rigid pieces and jointly optimize their depth and 3D motion. To evaluate MiD, we simply divide their predicted d_2 by d_1. As a result, our method achieves the lowest error in terms of MiD, reducing the error by 10X for monocular baselines, and outperforming the stereo baseline by a large margin (115 v.s. 75). A visual example is shown in Fig. 7.6. This demonstrates the effectiveness of modeling relative scale change via optical expansion.

Test performance (fg objects) We then evaluate our method on scene flow prediction for foreground objects on KITTI-15 benchmark, as shown in Tab. 7.2. We first compare against Mono-SF, the only monocular scene flow method on the benchmark. It formulates monocular scene flow estimation as an optimization problem, and use probabilistic predictions of a monocular depth network as one energy term.

Table 7.2: Scene flow estimation on KITTI-15 benchmark foreground pixels. All metrics are errors in perceptage shown for the foreground pixels. The best among the same group are bolded, and the best among all are underlined. Monocular methods are listed at the top, while stereo-based methods are listed below.

Method	D1	D2	Fl	SF	time (s)
Mono-SF [30]	**26.94**	32.70	19.64	39.57	41
Ours-mono	27.90	**31.59**	**8.66**	**36.67**	**0.2**
PRSM [268]	10.52	15.11	13.40	20.79	300
DRISF [152]	4.49	9.73	10.40	15.94	**0.75**
Ours-stereo	**3.46**	**8.54**	**8.66**	**13.44**	2

⋆ The expansion and motion-in-depth networks take 15ms for KITTI-sized images on a TITAN Xp GPU, giving a total run time of 200ms together with flow. Also notice that both PRSM and Mono-SF run on a single-core CPU, and could possibly be parallelized for a better speed.

Notice although our disparity error D1 is similar to Mono-SF, we obtain better D2 and SF metrics, which indicates that our prediction of normalized scene flow is more accurate.

Our method of estimating motion-in-depth and d_2 is also applicable to stereo scene flow, where we directly take GANet [321], the SOTA method on D1 metric, to predict the disparity of the first frame d_1. To obtain d_2, we divide the d_1 with estimated motion-in-depth as before. As a result, we obtained the SOTA accuracy on foreground depth change D2 and scene flow SF, which further demonstrates the effectiveness of our method for upgrading optical flow to 3D scene flow. In comparison, we effectively reason about relative depth change at a low cost (15ms), instead of explicitly computing the disparity at frame two. This gives us improved accuracy, spatial consistency and reduced latency.

7.4.2 LiDAR scene flow

Given two consecutive LiDAR scans of the scene, the LiDAR scene flow task is defined as estimating the 3D motions of the point clouds. Prior work either register two point clouds by optimization [48], or train a network to directly predict the 3D

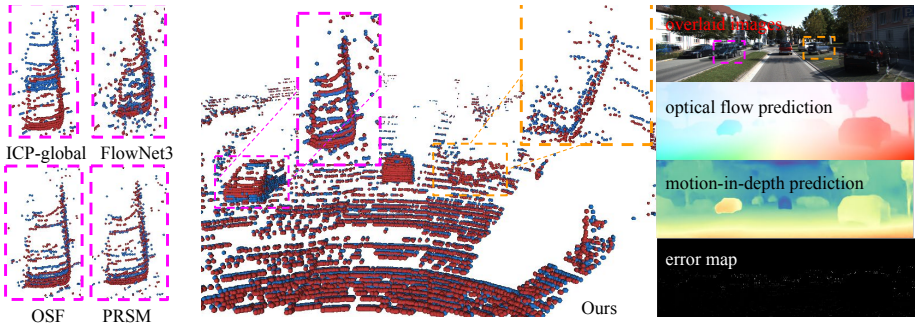

Figure 7.7: LiDAR scene flow result on KITTI-15 val set frame "000124". Red (2nd frame) and blue (translated first frame) points are supposed to overlap for perfect 3D flow estimation. Our method predicts more accurate 3D flow than global-ICP and FlowNet3 on the front vehicles. OSF and PRSM produce motion fields with a similar quality as ours, but use stereo images and are much slower.

motion [75, 141, 142].

Our solution Practically, LiDAR scans are usually paired with monocular cameras. Therefore, we use such monocular images to predict optical flow and expansion and convert them to normalized scene flow by Eq. 7.2. To obtain 3D scene flow for the point clouds, we project them onto the image plane and use LiDAR depth to "upgrade" the normalized scene flow to full 3D scene flow.

Evaluation protocol We compare with prior work on 42 KITTI validation images, using the evaluation protocol from MeteorNet [142]: raw LiDAR points are projected onto the image plane and the ground-truth 3D flow is constructed from disparity and flow annotations. Methods are scored by 3D end-point-error (EPE, L2 distance between vectors).

Baselines Among all the point-based methods, FlowNet3D and MeteorNet are finetuned on the same set of KITTI images as ours, and the numbers are taken from their paper. HPLFlowNet is trained on FlythingThings [158] and we modify their code to run on raw point clouds. ICP-global finds a single rigid transformation that best describes the motion of all the scene points, and does not deal with non-rigid elements. We further consider stereo scene flow methods [90, 162, 268], where the projected LiDAR depth and $\frac{d_2}{d_1}$ are used to determine the depth-wise flow displacements.

Results As in Tab. 7.3, our method trained on synthetic dataset already performs better than all the point-based methods as well as FlowNet3. After fine-tuning

Table 7.3: Evaluation LiDAR scene flow on KITTI-15.

Method	input	EPE (m)
ICP-global	points × 2	0.727
HPLFlowNet [75]	points × 2	0.590
FlowNet3D-ft [141]	points × 2	0.287
MeteorNet-ft [142]	points × 2	0.251
FlowNet3 [90]	points + stereo × 2	0.878
†FlowNet3-ft [90]	points + stereo × 2	0.551
OSF [162]	points + stereo × 2	0.137
PRSM [268]	points + stereo × 2	0.116
Ours	points + mono × 2	0.119
w/o ft	points + mono × 2	0.184

on KITTI, it out-performs all the stereo-based methods, except for PRSM, which takes 100X more inference time. Compared to point-based methods where exact 3D correspondence may not exist in the sparse scan, our method estimates normalized scene flow on a much denser pixel grid, which leads to higher precision. A visual example is shown in Fig. 7.7.

7.4.3 Time-to-collision estimation

Modelling time-to-collision (TTC) is important for robots to avoid collisions and plan the trajectory [34, 58, 154, 155, 167]. Indeed, knowing the motion-in-depth directly tells us the time a point takes to collide with the image plane by

$$T_c \quad = \quad \frac{Z}{Z - Z'} T \quad = \quad \frac{T}{1 - \tau},$$

assuming a constant velocity, where T is the sampling interval of the camera and τ is motion-in-depth [83]. We convert the motion-in-depth estimations to time-to-collision and compare our method with the baselines in Tab. 7.4.

We treat TTC prediction as a binary classification task where we predict whether the TTC is less than {1s, 2s, 5s} for each pixel [154]. The sampling interval is set as 0.1s and only the points with positive TTC ground-truth are evaluated. We compute

Table 7.4: Percentage errors of time-to-contact estimation on KITTI.

Method	Err-1s	Err-2s	Err-5s	Input
FlowNet3 [90]	22.87	21.49	15.97	stereo
†FlowNet3-ft [90]	11.97	13.86	12.43	stereo
OSF [162]	6.94	7.78	8.74	stereo
PRSM [268]	5.91	5.72	6.10	stereo
Ours	**4.21**	**4.07**	**4.51**	mono

the accuracy over 40 KITTI validation images as used in optical scene flow evaluation.

We find that OSF and PRSM perform reasonably well on TTC estimation, which is consistent with their high accuracy on motion-in-depth estimation. Our monocular method outperforms all the baselines for all time intervals, indicating it makes better predictions on possible future collisions.

7.4.4 Self-learning of optical expansion

We explore the task of self-supervised learning of optical flow and expansion. Our network is trained on 6800 images from KITTI depth estimation dataset [260] for 20k iterations, where the sequences that appear in KITTI-15 scene flow training set are excluded. Then we evaluate 40 validation KITTI-15 images as used in optical scene flow.

As for baselines, "Brightness" compares the difference between the intensity values of the reference and target pixel, and "Census" compares between intensity values of a $K \times K$ patch around the reference pixel and their correspondences. Both methods do not provide supervision signals for optical expansion. Our scale-aware loss provides supervision for optical expansion, and combined with census loss, gives the best performance, as shown in Tab.7.5.

7.4.5 Rigid depth estimation

Structure-from-motion jointly estimates camera poses and 3D point locations of a rigid scene given point correspondences [79]. However, for two frames undergoing a forward or backward translational camera motion, the triangulation error for pixels

Table 7.5: Results of self-supervised flow estimation on KITTI-15.

Method	Fl EPE	Exp. log-L_1
Brightness [206]	9.472	*N.A.*
Census [159]	7.000	*N.A.*
Ours-Scale	7.380	336
Ours-Census+Scale	**6.564**	348

near the focus of expansion (FoE), or epipole, is usually high due to the limited baseline and small triangulation angle [17, 59]. Here we describe a method of computing depth from optical expansion, which is not sensitive to small baseline.

Here we consider the case where camera motion is a given translation $\mathbf{t}_c = (t_{cx}, t_{cy}, t_{cz})$, and compare the depth estimation solutions using triangulation and motion-in-depth. For triangulation, assuming an identity camera intrinsics, we have depth

$$Z = \frac{x - \text{FoE}_x}{u} t_{cz} = \frac{y - \text{FoE}_y}{v} t_{cz},$$

where FoE = $(\frac{t_{cx}}{t_{cz}}, \frac{t_{cy}}{t_{cz}})$ and (u, v) is the displacement of reference point (x, y) [143]. Notice when only lateral movement exists, the above is equivalent to $Z = t_{cx}/u$. Motion-in-depth τ also tells us the depth via time-to-contact,

$$Z = \frac{1}{1 - \tau} t_{cz}.$$

Assuming the errors according to triangulation and time-to-contact are $\epsilon_{||\mathbf{u}||}$ and ϵ_τ respectively, we have

$$\epsilon_{Z_1} \sim \frac{1}{||u||^2}, \quad \epsilon_{Z_2} \sim \frac{1}{(1-\tau)^2},$$

which indicates large error occurs when flow is smaller for the triangulation solution, and large error occurs when optical expansion is close to 1 for the time-to-contact solution. Interestingly, it is always the case that for points near FoE where displacements are small, the optical expansion is either greater than one (moving forward) or

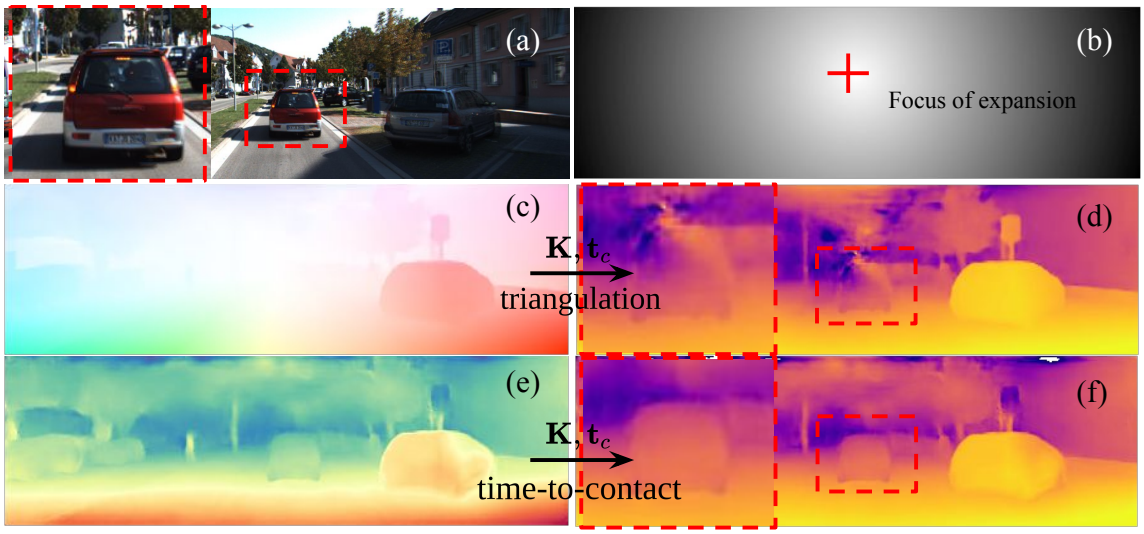

Figure 7.8: Rigid depth estimation with optical flow vs motion-in-depth. (a): overlaid input frames, where the pixel motion is relative small for the marked region near the focus of expansion. (b): distance to the focus of expansion from image coordinates, given by $||\mathbf{p} - \text{FoE}||$. (c): flow correspondences visualized by the Middlebury color wheel. (d): depth estimation by triangulation of flow correspondences, where the estimation for the marked region near the focus of expansion is corrupted due to small displacements. (e): motion-in-depth estimation. (f): depth reconstruction by time-to-contact, where the depth estimation near the focus of expansion is more robust than the triangulation method.

smaller than one (moving backward) [167], giving robust signals for reconstructing points near the FoE as shown in Fig. 7.8.

7.4.6 Two-view SfM with normalized scene flow

Problem setup In Sec. 4.5, we discussed using motion-in-depth τ to estimate depth Z for a static scene when the camera pose is given. Here, we solve for the camera motion $(\mathbf{R}_c, \mathbf{t}_c)$ and the 3D shape $\mathbf{P} = (X, Y, Z)$ jointly given normalized scene flow $\hat{\mathbf{t}}$ from two views of a static scene. This can be formulated as a optimization

problem, and we want to minimize the 3D re-projection error

$$
\begin{aligned}
L(\mathbf{R}_c, \mathbf{t}_c, \mathbf{Z}) &= \sum_k w_k d(\mathbf{R}_c \mathbf{P}_k + \mathbf{t}_c, \mathbf{P}_k + \mathbf{t}_k)^2 \\
&= \sum_k w_k ||\mathbf{R}_c \mathbf{P}_k + \mathbf{t}_c - (\mathbf{P}_k + \mathbf{t}_k)||^2 \\
&= \sum_k w_k ||\mathbf{R}_c (Z_k \tilde{\mathbf{P}}_k) + \mathbf{t}_c - (Z_k \tilde{\mathbf{P}}_k + Z_k \hat{\mathbf{t}}_k)||^2 \\
&= \sum_k w_k ||Z_k (\mathbf{R}_c \tilde{\mathbf{P}}_k - \tilde{\mathbf{P}}_k - \hat{\mathbf{t}}_k) + \mathbf{t}_c||^2,
\end{aligned}
$$

where k is the index of each point, w is the weight assigned to each point, and $\tilde{\mathbf{P}} = \mathbf{K}^{-1}\tilde{\mathbf{p}}$ are normalized 3D coordinates.

Solution Empirically, we find coordinate descent gives a robust solution to the above optimization problem. We alternate between pose $(\mathbf{R}_c, \mathbf{t}_c)$, and scales $\{Z_1, \ldots, Z_K\}$ as follows. Fixing the scale Z_k, the problem becomes finding a rigid transformation between two registered point sets. The solution can be described as: 1) align the center of two point clouds, 2) solve for rotation using SVD, and 3) solve for translation [236]. Fixing $(\mathbf{R}_c, \mathbf{t}_c)$, scales $\{Z_1, \ldots, Z_K\}$ can be obtained by solving a least square problem.

Results We initialize the depth to one for all 3D points $Z_k = 1, k \in \{1 \ldots, N\}$, and perform steepest descent for each sub-problem for five iterations. Qualitative results on KITTI and Blackbird [8] are shown in Fig. 7.9 and Fig. 7.11.

7.5 Ablation

Setup To demonstrate the advantage of our method over alternatives for estimating optical expansion, we perform an extensive set of diagnostics. For all experiments, we train the network for 20k iterations on Driving and Monkaa with a batch size of 8, and test on the 40 KITTI validation images used in optical scene flow experiments. We also test on the sintel training set, which compared to KITTI, has more dynamic objects and a much smaller range of optical expansion, since depth does not change much over frames.

Comparison to expansion-based options We first remove the residual prediction structure and directly learn to regress the optical expansion from the initial prediction

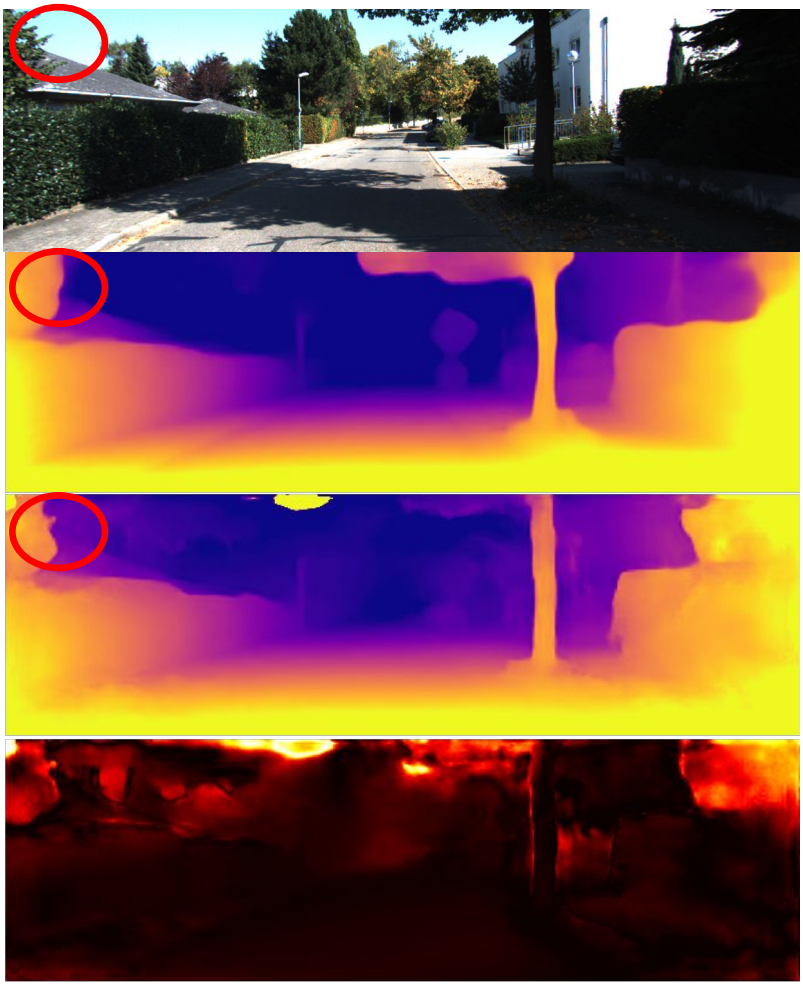

Figure 7.9: Results on KITTI sequence "10-03-0034" (not in the training sequences) frame 840, where the scene motion can be described by a rigid transform. From top to bottom: reference image, depth prediction from MonoDepth2 [72], our depth prediction and fitting error (uncertainty). Our method jointly estimates camera pose together with scene geometry, and predicts sharper boundaries.

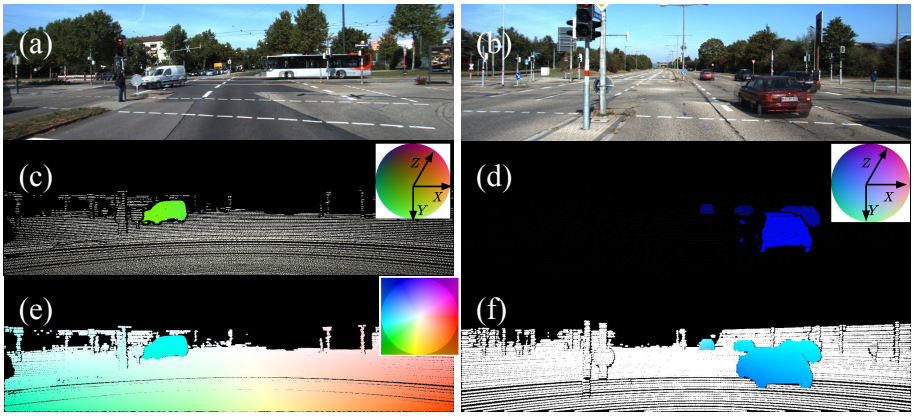

Figure 7.10: Normalized scene flow v.s. optical flow. (a)-(b): overlaid images of two consecutive frames. (c)-(d): visualization of normalized scene flow using negative and positive color hemispheres. Notice that normalized scene flow provides information about depth change, where in (c) the front car moving left-inwards is colored green and in (d) the car moving left-outwards is colored blue. (e)-(f): visualization of optical flow using the Middlebury color wheel [11]. In comparison, optical flow is not sensitive to change of depth, where left-moving cars are all bluish, no matter moving towards or away from the camera.

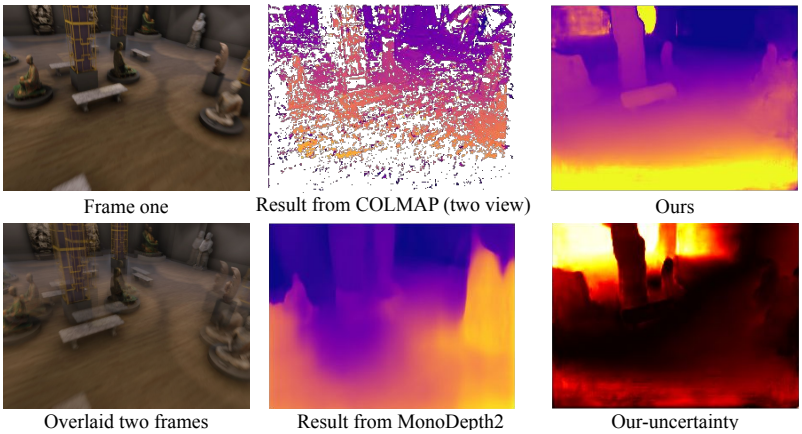

Figure 7.11: Results on Blackbird dataset [8], which is a unmanned aerial vehicle dataset for aggressive indoor flight. Prior method of estimating the scene geometry either uses sparse point correspondences, for example COLMAP [220], or rely on strong data prior, for example MonoDepth2 [72]. Our method produces dense and reliable depth without strong priors by making use of dense optical expansion.

Table 7.6: Ablation study on optical expansion estimation.

Method	KITTI log-L_1	Sintel log-L_1
Ours	**245**	**78**
w/o residual	255	83
affine→flow	383	116
affine→warp	450	174
Raw affine transform	363	131
Matching over scales	541	145

and find the performance drops slightly. Then we investigate the effectiveness of input features. Replacing initial expansion to flow predictions as inputs increases the error by 50.2% on KITTI and 39.8 on Sintel, which shows the initial scale extracted from local affine transform is crucial for estimating the optical expansion. We then replace initial expansion with the reference and warped target image features (by flow) as inputs, and find the error rises by 76.5% on KITTI and 109.6% on Sintel, which indicates it is difficult to learn optical expansion directly from image features. To demonstrate the improvement from the optical expansion network, we evaluate the raw scale component extracted from the local affine transforms, which increases the error by 42.4% on KITTI and 57.8% on Sintel.

Matching over scales We consider a scale matching network baseline that searches for scale over a pyramid of images [196, 275, 295]. At the quarter feature resolution, we discretize the $s \in [0.5, 2]$ into $S = 9$ intervals in log space, and construct a pyramid by scaling the reference image features. Then a 3D cost volume of size $(H/4, W/4, S)$ is constructed by taking the dot product between reference feature and target feature pyramid warped by the optical flow prediction. The cost volume is further processed by 3D convolutions and soft-argmin regression following prior work on stereo matching [112, 305]. However, this approach faces a hard time predicting the optical expansion correctly. We posit the signals in raw images feature is not strong enough for the matching network to directly reason about expansions.

7.6 Discussion

We explore problems of 3D perception using monocular cameras and propose to estimate optical expansion, which provides rich information about relative depth change. We design a neural architecture for optical expansion and normalized scene flow, associated with a set of supervised or self-supervised learning strategies. As a result, significant improvements over prior art on multiple 3D perception tasks are achieved, including LiDAR scene flow, optical scene flow, and time-to-collision estimation. For future work, we think dense optical expansion is a valuable low-level cue for motion segmentation and robot collision avoidance. Moreover, the geometric relationship between optical expansion and normalized scene flow is currently established assuming a scaled orthographic camera model and non-rotating scene elements. Extending it to a perspective camera model with rotating scene elements would be interesting. Finally, background rigidity is a powerful prior for depth and motion estimation, and incorporating it with our local estimates would further improve the performance.

Chapter 8

Learning to Segment Rigid Motions from Two Frames

8.1 Introduction

Autonomous agents such as self-driving cars need to be able to navigate safely in dynamic environments. Static environments are far easier to process because one can make use of geometric constraints (SFM/SLAM) to infer scene structure [52]. Dynamic environments require the fundamental ability to both segment moving obstacles and estimate their depth and speed [15]. Popular solutions include object detection or semantic segmentation [134]. While one can build accurate detectors for many categories of objects that are able to move, "being able to move" is not equivalent to "moving". For example, there is a profound difference between a parked car and an ever-so-slightly moving car (that is pulling out of parked location), in terms of the appropriate response needed from a nearby autonomous agent. Secondly, class-specific detectors rely heavily on appearance cues and categories present in a training set. Consider a trash can that falls on the street; current *closed-world* detectors will likely not be able to model all types of moving debris. This poses severe implications for safety in the open-world that a truly autonomous agent must operate [19].

Problem formulation: We follow historic work on motion-based perceptual grouping [91, 227, 258, 278, 298] and segment moving objects without relying on appearance cues. Specifically, we focus on segmenting *rigid* bodies from *two frames*. We focus

(a) Two frame overlay (b) Naive flow-triangulated reconstruction (c) Sampson err. (white indicates moving)

Figure 8.1: Collinear motion ambiguity. (a) The input scene contains a dynamic object (the car in the lower left) moving parallel to camera translational direction. (b) One can triangulate motion correspondences assuming overall *rigidity* that places the moving car at an elevated height, which illustrates both (1) the commonality of this degenerate case [319] in urban navigation, and (2) one solution of using structural scene priors that do not allow for floating objects above the ground. (c) Due to such ambiguities, the 2D motion of the moving car is *consistent* with the camera egomotion, leaving it indistinguishable under classic motion segmentation metrics such as Sampson error [79].

on two-frame because it is the minimal set of inputs to study the problem of motion segmentation, and in practice, perception-for-autonomy needs to respond immediately to dynamic scenes, e.g., an animal that appears from behind an occlusion. We focus on rigid body and its *3D* motion parameterizations because it's directly relevant for an autonomous agent acting in a 3D world. While dynamic scenes often contain nonrigid objects such as people, we expect that deformable objects may be modeled as a rigid body over short time scales, or decomposed into rigidly-moving parts [5, 23].

Challenges: Earlier work on rigid motion segmentation often makes use of geometric constraints arising from epipolar geometry and rigid transformations. However, there are several fundamental difficulties that plague geometric motion segmentation. First, epipolar constraints fail when camera motion is close to zero [298]. Second, points moving along epipolar lines cannot be distinguished from the rigid background [319], which we discuss at length in Sec. 8.3.1. Third, geometric criteria are often not robust enough to noisy motion correspondences and camera egomotion estimates, which can lead to catastrophic failures in practice.

Method: We theoretically analyze ambiguities in 3D rigid motion segmentation, and resolve such ambiguities by exploiting recent techniques for upgrading 2D motion observation to 3D with optical expansion [302] and monocular depth cues [198]. To deal with noisy motion correspondences and degenerate scene motion, we design a convolutional architecture that segments the rigid background and an arbitrary number of rigid bodies from a given motion field. Finally, we parameterize the 3D

motion of individual rigid bodies by fitting 3D rigid transformations.

Contributions: (1) We provide a geometric analysis for ambiguities in 3D rigid motion segmentation from 2D motion fields, and introduce solutions to deal with such ambiguities. (2) We propose a geometry-aware architecture for 3D rigid motion segmentation from two RGB frames, which is generalizable to novel appearance, resilient to different motion types and robust to noisy motion observations. (3) Our method achieves state-of-the-art (SOTA) performance of rigid motion segmentation on KITTI/Sintel. The inferred rigidity masks significantly improve the performance of downstream depth and scene flow estimation tasks.

8.2 Related Work

Geometric Motion Segmentation: The problem of clustering motion correspondences into groups that follow a similar 3D motion model has been extensively studied in the past [254, 256, 258, 263, 264, 298, 319]. However, prior methods either focus on theoretical analysis with noisy-free data, or assume relatively simple scenes where long-term motion trajectories can be obtained by point tracking algorithms. Some recent work [20, 22, 286] tackles more complex scenarios with two-frame optical flow inputs, where geometric constraints, such as motion angle and plane plus parallax (P+P) [218] are considered as cues of "moving versus static". However, such geometric constraints are sensitive to noise in optical flow even under a robust estimation framework [57]. Moreover, as we shall see in Sec. 8.3.1, the prior two-frame solutions do not deal with several degenerate cases, including co-planar/co-linear motion [319] and camera motion degeneracy [256]. We address these problems by encoding geometric constraints into a modular neural network.

Learning-Based Video Object Segmentation: Segmenting salient objects from videos historically stems from the problem of image salient object detection [177, 186], where existing methods often rely either on appearance features or on salient motions from 2D optical flow [96, 148, 251, 252, 312, 329]. Oftentimes, optical flow is interpreted as a color image [96, 329], where geometric information, such as camera egomotion, is ignored. Close to our methodology, Motion Angle Network (MoA-Net) [21], analytically reduces the effect of camera rotation and uses the "rectified" flow angle as input features to a binary segmentation network. Our approach further

incorporates 3D flow and depth cues and segments multiple 3D rigid motions.

Instance Scene Flow: Scene flow is the problem of resolving dense 3D scene motion from an ego-camera [161, 262], which is challenging due to the lack of visual evidence to find correspondence matches, for example, when occlusion occurs. Prior approaches often utilize scene rigidity priors to resolve such ambiguities, such as piecewise rigidity prior [161, 268] and semantic rigidity prior [18, 152]. However, it is risky to segment the scene purely relying on semantics – an object that is able to move is not the same as an object that is moving. Furthermore, such high-level cues do not generalize to an open-world, where algorithms are required to be robust to never-before-seen categories [19]. Instead, we exploit *motion rigidity* for scene flow estimation, which decomposes the scene into multiple rigidly moving segments while preserving the completeness of individual rigid bodies.

8.3 Approach

In this section, we first analyze degeneracies in motion segmentation that arise when dynamic motion is indistinguishable from the camera motion, and what information is required to resolve the ambiguities. We then design a neural architecture for rigid instance motion segmentation that builds on this geometric analysis, producing a pipeline for two-frame rigid motion segmentation.

8.3.1 Two-Frame 3D Motion Segmentation

Problem setup: Given two-frame 2D motion correspondences written in homogenous coordinates $(\tilde{\mathbf{p}}_\mathbf{0}, \tilde{\mathbf{p}}_\mathbf{1})$ with depths (Z_0, Z_1) observed by camera intrinsics $(\mathbf{K_0}, \mathbf{K_1})$, the corresponding 3D points in the camera coordinate system of each frame is given by $\mathbf{P_0} = Z_0 {\mathbf{K_0}}^{-1} \tilde{\mathbf{p}}_\mathbf{0}$ and $\mathbf{P_1} = Z_1 {\mathbf{K_1}}^{-1} \tilde{\mathbf{p}}_\mathbf{1}$. We wish to detect points whose 3D motion cannot be described by camera motion $\mathbf{R_c} \in \mathbf{SO}(3)$, $\mathbf{T_c} \in \mathbb{R}^3$:

$$(\mathbf{R_c P_1} + \mathbf{T_c}) - \mathbf{P_0} \neq \mathbf{0}, \qquad \text{(transform of coordinates)} \tag{8.1}$$

To gain more geometric insights, we re-arrange Eq. (8.1) into

$$\begin{aligned} &\mathbf{T_{sf}} = \mathbf{K_0}^{-1}(Z_1\mathbf{H_R}\tilde{\mathbf{p}}_\mathbf{1} - Z_0\tilde{\mathbf{p}}_\mathbf{0}) \neq -\mathbf{T_c}, \\ &(\text{“rectified” 3D scene flow} \neq \text{negative camera translation}) \end{aligned} \tag{8.2}$$

where $\mathbf{T_{sf}} = \mathbf{R_c}\mathbf{P_1} - \mathbf{P_0}$ is the “rectified” 3D scene flow, with the motion induced by camera rotation $\mathbf{R_c}$ removed through “rectifying” the second frame coordinate system to have the same orientation as the first frame; and $\mathbf{H_R} = \mathbf{K_0}\mathbf{R_c}\mathbf{K_1}^{-1}$ is the rotational homography that “rectifies” the second image plane into the same orientation as the first image plane, removing the effect of camera rotation from the 2D motion fields. Eq. (8.2) states that the rectified 3D scene flow of a moving point $\mathbf{P}$ will not equal the negative camera translation. However, assuming camera intrinsics and motion are known, there are still two crucial degrees of freedom that are undetermined: depth Z_0 and Z_1.

Coplanar motion degeneracy: Solving for Z_0 and Z_1 equates to estimating the depth and 3D scene flow, which itself is challenging [161]. To remove such dependencies, classic geometric motion segmentation segments points whose *2D motion* is inconsistent with the camera motion, measured either by Sampson distance to the epipolar line [79, 254] or plane plus parallax (P+P) [218] representations that factor out camera rotation, allowing one to evaluate the angular deviation of the 2D motion to the epipole [20, 91]. However, *is 2D motion sufficient to segment points moving in 3D?* The answer is no (Fig. 8.1). Formally, 3D points that translate within the epipolar plane defined by the camera translation vector $\mathbf{T}_c$ will project to the epipolar line, making them ”appear” as stationary points, as shown in Fig. 8.2 Case (II).

To detect such co-planar motion, we make use of optical expansion cues that upgrade 2D flow to 3D as suggested by recent work [302]. Optical expansion, measured by the scale change of overlapping image patches, approximates the relative depth $\tau = \frac{Z_1}{Z_0}$ for non-rotating scene elements under scaled orthographic projection [302]. We derive a 3D motion angle criterion that does not require depth, but removes the ambiguity of points moving within the epipolar plane. Normalizing Eq. (8.2) by depth

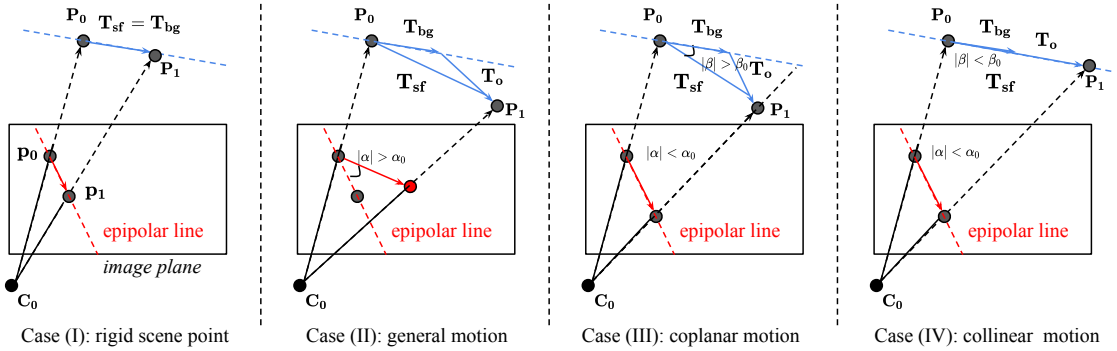

Figure 8.2: When can a moving scene point **P** be identified from a moving camera? Rectified 3D scene flow of **P** (that assumes camera rotation has been removed) can be written as the sum of rigid background motion (induced by the camera) and independent object motion $\mathbf{T_{sf}} = \mathbf{T_{bg}} + \mathbf{T_o}$, where $\mathbf{T_{bg}} = -\mathbf{T_c}$. Case (I): Assuming a rigid scene point **P** with zero independent motion $\mathbf{T_o} = \mathbf{0}$, the 2D motion projected by $\mathbf{T_{sf}}$ must lie on the epipolar line. Case (II): In other words, if the 2D motion deviates from the epipolar line, $|\alpha| > \alpha_0$, **P** must be moving, analagous to Sampson error [79]. Case (III): *However, the inverse does not hold.* If 2D flow is consistent with the background motion ($|\alpha| < \alpha_0$), **P** might still be moving in the epipolar plane. However, if the *angular direction* of 3D flow $\mathbf{T_{sf}}$ – computable from optical expansion [302] – differs from $\mathbf{T_{bg}}$ ($|\beta| > \beta_0$), **P** must be moving. Case (IV): If the 3D flow direction is consistent with background motion ($|\beta| < \beta_0$), **P** could still be moving in the direction of $\mathbf{T_{bg}}$, making it unrecoverable without knowing the scale (or relative depth).

Z_0, we have

$$\tilde{\mathbf{T}}_{\mathbf{sf}} = \mathbf{K_0}^{-1}(\tau \mathbf{H_R} \tilde{\mathbf{p}}_\mathbf{1} - \tilde{\mathbf{p}}_\mathbf{0}) \nsim -\mathbf{T_c}, \quad \text{(rectified 3D flow direction} \neq \text{neg. camera translation direction)} \tag{8.3}$$

where $\tilde{\mathbf{T}}_{\mathbf{sf}} = \frac{\mathbf{T_{sf}}}{Z_0}$ is the rectified and normalized 3D flow and $\nsim$ indicates two vectors are different in their directions. Eq. (8.3) states that a point is moving if the direction of its rectified 3D scene flow is not consistent with the direction of the camera translation, as shown in Fig. 8.2 Case (III).

Collinear motion degeneracy: However, there is still a remaining ambiguity that cannot be resolved. If point **P** moves in the opposite direction of the camera translation, both classic criteria and Eq. (8.3) would fail, as shown in Fig. 8.2 Case (IV). Such ambiguity remains even given multiple frames [319], but is common in many real-world applications, e.g., two cars passing each other (Fig. 8.1). To identify moving points in such cases, one could use depth Z_0 to recover the metric scale of normalized rectified scene flow $\tilde{\mathbf{T}}_{\mathbf{sf}}$, and compare it with camera translation $\mathbf{T_c}$. However, in a monocular setup, we neither know the scale of $\mathbf{T_c}$ nor trust the overall scale of Z_0 [79]. Instead, we derive a depth contrast criterion, inspired by an observation that *dynamic* scene points triangulated from flow assuming overall rigidity will appear "abnormal" in the 3D reconstruction, such as the floating car in Fig. 8.1 (b). To do so, we contrast the flow-derived depth Z_0^{flow} with a depth prior Z_0^{prior},

$$Z_0^{flow} \neq \gamma Z_0^{prior}, \quad \text{(flow-triangulated depth} \neq \text{depth prior)} \tag{8.4}$$

where Z_0^{flow} can be computed efficiently using midpoint triangulation [79], depth prior Z_0^{prior} can be represented by a data-driven monocular depth network, and the scale factor γ that globally aligns Z_0^{prior} to Z_0^{flow} can be determined by the ratio of their medians or robust least squares [241].

Egomotion degeneracy: Furthermore, when the camera translation is small, $\mathbf{T_c}$ is notoriously difficult to estimate due to small motion parallax. In such cases, rigid-background motion (and objects that deviate from it) is easier to model with a rotational homography model [254, 255].

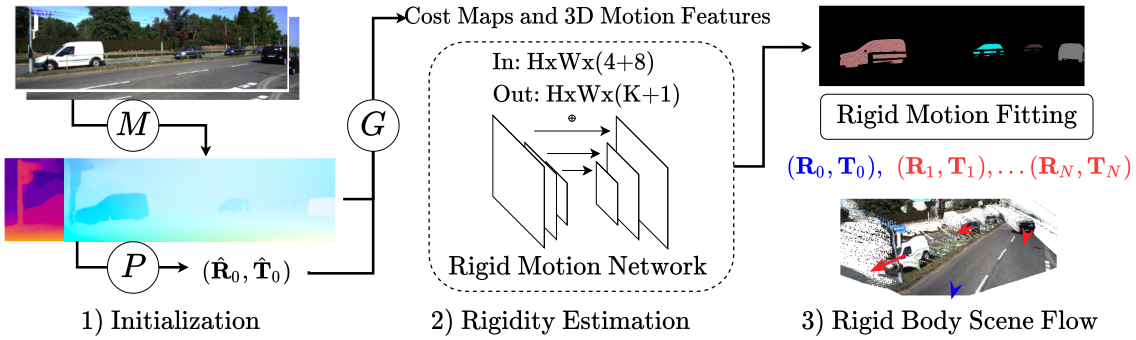

Figure 8.3: We detect and estimate rigid motions in three steps: First, depth and optical flow are computed using off-the-shelf networks (M) and camera motion is estimated by epipolar geometry (P) given two frames. Then, rigidity cost maps and rectified scene flow are computed (G) and fed into a two-stream network that produces the segmentation masks of a rigid background and an arbitrary number of rigidly moving instances. Finally, we fit rigid transformations for the background and each rigid instance to update their depth and 3D scene flow.

8.3.2 Learning to Segment Rigid Motions

We now operationalize our motion analysis into a deep network for rigid motion segmentation (Fig. 8.3). At its heart, *our network learns to transform motion measurements (noisy 3D scene flow) into pixel-level masks of rigid background and instances.* To do so, we construct motion cost maps designed to address the motion degeneracies described earlier. Given such input maps and raw scene flow measurements, we use a two-stream network architecture that separately regresses the rigid background and rigid instance masks.

Motion estimation: First, we extract the camera and relative scene motion given two frames. We apply existing methods to estimate optical flow, optical expansion and monocular depth [198, 302]. To estimate camera motion, we fit and decompose essential matrices from flow maps using the five-point algorithm with a differentiable and parallel RANSAC [28].

Rigidity cost-maps inputs: We construct rigidity cost maps tailored to camera-object motion configurations analyzed in Sec. 8.3.1, including (1) an epipolar cost for general configurations, computed as per-pixel Sampson error [79]; (2) a homography cost to deal with small camera translations, implemented as per-pixel symmetric transfer error [51] with regard to the rotational homography, $\mathbf{H_R} = \mathbf{K_0 R_c K_1}^{-1}$; (3) a

3D P+P cost to detect coplanar motions, computed as

$$c_{\text{3D}} = ||\tilde{\mathbf{T}}_{\mathbf{sf}}|| \cdot |\sin\beta|, \tag{8.5}$$

where $\beta = |\angle(\tilde{\mathbf{T}}_{\mathbf{sf}}, -\mathbf{T}_{\mathbf{c}})|$ is the measured angle between normalized scene flow $\tilde{\mathbf{T}}_{\mathbf{sf}}$ (computed by Eq. 8.3) and negative camera translation $-\mathbf{T}_{\mathbf{c}}$; and (4) a depth contrast cost to deal with colinear motion ambituity, computed as

$$c_{\text{depth}} = |\log(\frac{Z_0^{\text{flow}}}{\gamma Z_0^{\text{prior}}})|. \tag{8.6}$$

Please refer to the supplement for visuals and more details.

Rectified scene flow inputs: Besides rigidity cost-maps, we find it helpful to also input raw scene flow measurements, represented as an 8-channel feature map, containing the first frame 3D scene points $\mathbf{P_0}$, rectified motion fields $\mathbf{T_{sf}}$, and uncertainty estimations of flow and optical expansion (σ_1, σ_2). To compute $\mathbf{P_0}$, we back-project the first frame pixel coordinates given monocular depth inputs; to compute $\mathbf{T_{sf}}$, we upgrade optical flow using optical expansion τ,

$$\tilde{\mathbf{T}}_{\mathbf{sf}} = \mathbf{K_0}^{-1}(\tau \mathbf{H_R} \tilde{\mathbf{p}}_{\mathbf{1}} - \tilde{\mathbf{p}}_{\mathbf{0}}), \tag{8.7}$$

where the second coordinate frame is rectified by rotational homography $\mathbf{H_R} = \mathbf{K_0}\mathbf{R_c}\mathbf{K_1}^{-1}$ to remove the effect of camera rotation. Finally, the uncertainty of optical flow and optical expansion are computed as out-of-range confidence score and Gaussian variance respectively [89, 301]. Such rectified scene flow inputs are more effective than 2D optical flow, as empirically tested in ablation study (Tab. 8.4).

Architecture: We use a two-stream architecture: (1) a lightweight U-Net [210] architecture to predict binary labels for pixels belonging to the (rigid) background and (2) a CenterNet [331] architecture to predict pixel instance masks. Inspired by the single-shot segmentation head proposed in PolarMask [289], stream (2) outputs a heatmap representing object centers and a regression map of $K = 36$ polar distances at evenly distributed angles. Intuitively, stream (2) generates coarse instance-level masks that are refined by pixel-accurate background masks from stream (1). Specifically, pixels where rigid background and instance predictions disagree are not used for rigid

body fitting below, and marked as incorrect during evaluation.

Losses: The overall architecture is end-to-end differentiable and can be trained with standard loss functions,

$$L = \alpha_1 L_{\text{binary}} + \alpha_2 L_{\text{center}} + \alpha_3 L_{\text{polar}} \tag{8.8}$$

where L_{binary} is binary cross-entropy loss with label balancing, L_{center} is the focal loss [136, 331] and L_{polar} is the polar loss defined in PolarMask [289]. Given ground-truth contours of M objects, we convert them to polar coordinates quantized as K rays uniformly emitted from their mass-centers. Then the polar loss is computed as

$$L_{polar} = \frac{1}{KM} \sum_{i=1}^{M} \sum_{k=1}^{K} |d_{i,k} - d^*_{i,k}|, \tag{8.9}$$

where $d^*_{i,k}$ is the ground-truth distance of the k-th ray to the mass-center. Weights are balanced as $\alpha_1 = 1^{-4}$, $\alpha_2 = 1^{-3}$ and $\alpha_3 = 1^{-7}$ through grid search.

Rigid body scene flow: Given segmentations of rigid bodies, our goal is to parameterize 3D scene flow as 3D geometry and transformations of rigid bodies by fitting flow and depth observations. We provide details in Alg. 1. To find the best fit of rotations and up-to-scale translations, we estimate and decompose essential matrices over flow correspondences with RANSAC [79]. To obain a more reliable 3D reconstruction than back-projected monocular depth, we triangulate flow using rigid motion estimations for each rigid body, and determine their scales by aligning each triangulated depth map to monocular depth inputs with RANSAC [241]. Given the above parameterization, the second frame coordinates are computed as

$$\mathbf{P_1} = \textstyle\sum_{i=0}^{N} \mathbf{S}_i(\mathbf{R_i P_0} + \mathbf{T_i}), \tag{8.10}$$

where $\mathbf{S}_i$ is a one-hot rigid motion segmentation vector.

8.4 Experiments

Our method is quantitatively compared with state-of-the-art rigidity estimation algorithms on KITTI and Sintel in Sec. 8.4.1, and then applied to the depth and

Algorithm 1 Rigid body scene flow (monocular)

Input: Rigid body segmentation maps $\{\mathbf{S}_0, \ldots, \mathbf{S}_N\}$, flow correspondence $(\mathbf{p}, \mathbf{p}')$, first frame depth map Z_{prior}, intrinsics $(\mathbf{K}, \mathbf{K}')$.

Output: Rigid transformation $\{(\mathbf{R}_0, \mathbf{T}_0) \ldots, (\mathbf{R}_N, \mathbf{T}_N)\}$, first frame scene points $\{\mathbf{P}_0, \ldots, \mathbf{P}_N\}$.

Normalize coordinates $\tilde{\mathbf{p}} \leftarrow \mathbf{K}^{-1}[\mathbf{p}, 1]^T, \quad \tilde{\mathbf{p}}' \leftarrow \mathbf{K}'^{-1}[\mathbf{p}', 1]^T$
For $i = 0 \cdots N$ ▷ $i = 0$ indicates the rigid background
  $(\tilde{\mathbf{p}}_i, \tilde{\mathbf{p}}_i') \leftarrow \{(\tilde{\mathbf{p}}, \tilde{\mathbf{p}}'), \mathbf{S}_i(\mathbf{p}) = 1\}$ ▷ points on the current body
  Fit essential matrix $\mathbf{E}_i$ over $(\tilde{\mathbf{p}}_i, \tilde{\mathbf{p}}_i')$ with 5-pt+RANSAC.
  Decompose $\mathbf{E}_i$ and select the best $(\mathbf{R}_i, \mathbf{T}_i)$ by cheirality check [19].
  Triangulate 3D points $\mathbf{P}_i$ from $(\tilde{\mathbf{p}}_i, \tilde{\mathbf{p}}_i')$ and $(\mathbf{R}_i, \mathbf{T}_i)$.
  Align $\mathbf{P}_i^{(3)}$ to Z_{prior} by scale s_i with RANSAC. ▷ scale ambiguity
    $\mathbf{T}_i \leftarrow s_i \mathbf{T}_i, \quad \mathbf{P}_i \leftarrow s_i \mathbf{P}_i$
  if $c_{\text{hom}} < 4$ ▷ when parallax motion is small: supp.mat. Eq. (2)
  then $\mathbf{T}_i \leftarrow \mathbf{0}, \quad \mathbf{P}_i \leftarrow Z_{prior}\tilde{\mathbf{p}}_i$ ▷ rely on depth prior

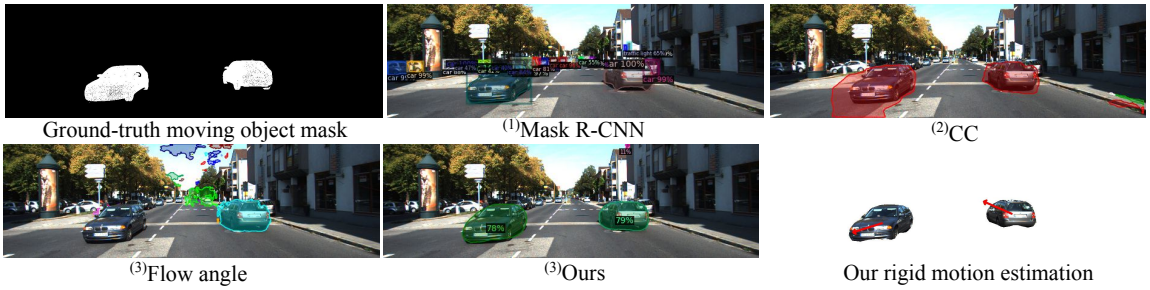

Figure 8.4: Comparison on KITTI-SF, image 137. The prefix of each method indicates the test-time inputs: (1)Single frame. (2)Multi-frame with appearance features. (3)Multi-frame without appearance. Our best appearance-based segmentation baseline, (1)Mask R-CNN [81] detects all the moving vehicles, but also reports parked cars in the background. (2)CC [199] correctly detects moving cars but also reports the edge of the road as moving objects. (3)Geometric segmentation algorithm [20, 286] fails on the approaching car due to the colinear motion ambiguity, and reports false positives at the background due to the noisy flow estimation. In contrast, (3)our method correctly segments both the moving vehicles and the rigid background. Rigid motions are estimated within each mask and applied to depth and scene flow estimation.

scene flow estimation tasks in Sec. 8.4.2- 8.4.3. In Sec. 8.4.4 we conclude with an ablation study.

Dataset: We use KITTI-SF (sceneflow) and Sintel for quantitative analysis. KITTI-SF [64, 161] features an urban driving scene with multiple rigidly moving vehicles. Sintel [33] is a synthetic movie dataset that features a highly dynamic environment. It contains viewpoints and objects (such as dragons) that are rare in existing datasets. KITTI provides ground-truth segmentation masks for the rigid background and moving car instances, where the remaining dynamic objects (such as pedestrians) are manually removed. For Sintel, computing rigid instances masks is an ill-posed problem since most objects are nonrigid. Instead, we obtain ground-truth rigid background segmentation from MR-Flow [286]. Both datasets also provide ground-truth depth and scene flow as well as camera intrinsics.

Implementation: We use MiDaS [198], a state-of-the-art monocular depth estimator to acquire imprecise, up-to-scale depth of the first frame as inputs. The remaining networks are trained without target domain data: optical flow and optical expansion networks are trained using FlyingChairs, SceneFlow, VIPER, and HD1K [50, 120, 158, 208]; the rigid motion segmentation network is trained on SceneFlow [158].

8.4.1 Two-frame Rigid Motion Segmentation

Metrics: Following prior works, we compute background IoU [150, 199] for rigid background segmentation and object F-measure [46] for rigid instance segmentation. Only the rigid background segmentation metric is reported on Sintel due to the lack of rigid bodies ground-truth rigid motion segmentation masks.

Baselines: We group baselines according to test inputs.

(1)**Single frame methods**. Mask R-CNN with ResNeXt-101+FPN backone is the most accurate model on MSCOCO provided by Detectron2 [81, 135, 285, 290]; U^2Net [195] is a state-of-the-art salient object detector; and MR-Flow-S [286] is a semantic rigidity estimation network fine-tuned separately on KITTI and Sintel.

(2)**Multi-frame with appearance features**. FusionSeg [96] is a two-stream architecture that fuses the appearance and optical flow features, and we provide SOTA optical flow on KITTI and Sintel as motion input. COSNet [148] and MATNet [329]

Table 8.1: Rigidity estimation on KITTI (K) and Sintel (S) without fine-tuning. (1)Single frame. (2)Multi-frame with appearance features. (3)Multi-frame without appearance. The best result under each metric (IoU in %) is in bold. *:For methods only estimating background masks, we use connected components to obtain object masks. ‡:Methods trained on target dataset. MR-Flow-S (K) is trained on KITTI, and MR-Flow-S (S) is trained on Sintel.

	Method	K: obj ↑	K: bg ↑	S: bg ↑
(1)	Mask R-CNN [285]	88.20	96.42	81.98
	U^2 (Saliency) [195]	64.80*	93.34	82.01
	MR-Flow-S (K) [286]	75.59*	94.70‡	76.11
	MR-Flow-S (S) [286]	11.11*	84.72	92.64‡
(2)	FSEG [96]	85.08*	96.27	80.22
	MAT-Net [329]	68.40*	93.08	77.95
	COSNet [148]	66.67*	93.03	80.86
	CC [199]	50.87*	85.50	✗
	RTN [150]	34.29*	84.44	64.86
(3)	FSEG-Motion [96]	61.29	89.41	78.25
	CC-Motion [199]	42.99	74.06	✗
	Flow angle [20, 286]	25.83	85.52	74.23
	Ours	**90.71**	**97.05**	**86.72**

are SOTA video objection segmentation methods on DAVIS [186]. CC [199] combines "flow-egomotion consensus score" (similar to our epipolar costs) with a foreground probability regressed from five consecutive frames, which is then thresholded to obtain the background mask. RTN [150] uses a CNN to predict rigid background masks given two RGBD images. For Sintel, we use the ground-truth depth as input; for KITTI, since the ground-truth depth is sparse, we use MonoDepth2 [72] instead.

(3)**Two-frame without appearance**. We separately evaluate the motion stream of FSEG and the flow-egomotion consensus results of CC. Following prior work [20, 286], we implement a classic motion segmentation pipeline that combines the motion angle and motion residual criteria.

Besides CC, RTN, and the classic pipeline, all baselines are trained or pre-trained on large-scale manually annotated segmentation datasets that contain common objects and scenes, while ours is not.

Performance analysis: We show qualitative comparison in Fig. 8.4 and report results in Tab. 8.1. On KITTI, our method outperforms the most accurate baseline,

Table 8.2: Monocular depth and scene flow results on KITTI (K) and Sintel (S). D1: first frame disparity (inverse depth) error. SF: scene flow error (%). The best result is in bold, and underlined if not trained on the target domain data. On Sintel, we evaluate on 330 frame pairs with average flow magnitude greater than 5 pixel.

Method	K: D1 ↓	K:SF ↓	S: D1 ↓	S:SF ↓
CC [199]	36.20	51.80	✗	✗
SSM [87]	31.25	47.05	✗	✗
Mono-SF [30]	**16.72**	**21.60**	✗	✗
MiDaS+OE [302]	37.27	44.87	49.89	55.43
MiDaS+Mask	17.33	22.47	39.60	47.40
MiDaS+Ours	16.98	22.19	**38.29**	**46.05**

Mask R-CNN, in terms of both rigid instance segmentation and background segmentation. Although Mask R-CNN is trained on common scenes (including driving), it cannot tell whether an object is moving or static, similar to other single-frame methods. Therefore, our method compares favorably to Mask R-CNN on rigid motion segmentation task. On Sintel, our method outperforms all the baselines except MR-Flow-S (S), which uses the first half of all Sintel sequences for training. If we compare to MR-Flow-S (K), which is not fine-tuned on Sintel, our method is better. Finally, among the motion-based segmentation methods, our method is the best on both datasets, because of our robustness to degenerate motion configurations as well as noisy flow inputs.

8.4.2 Monocular Scene Flow

We then apply the estimated rigid motion masks to two-frame depth and scene flow estimation on KITTI and Sintel. Following Alg. 1, we estimate 3D scene flow by fitting rigid transformations to initial depth and optical flow estimations.

Metrics: Following the convention of KITTI [161], we arrange Sintel as pairs of adjacent frames, and report the average depth and scene flow estimation performance on KITTI and Sintel. We report disparity error on both frames (D1, D2), optical flow error (Fl) and scene flow error (SF). To remove the overall scale ambiguity, we take an extra step to align the overall scale of the predictions to the ground-truth with their medians [198, 269].

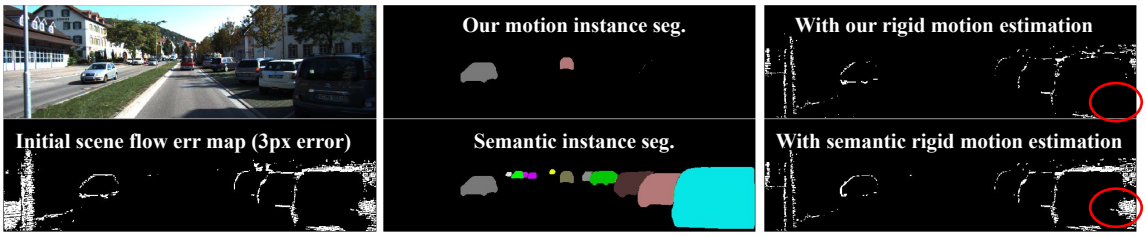

Figure 8.5: Rigidity vs semantic-based segmentation for instance scene flow. Given instance segmentation masks, scene flow can be optimized by fitting rigid body transforms within each mask. While semantic segmentation fails to improve scene flow estimation on the parked cars (in red circle), our rigid motion mask groups the parked car together with the rigid background and successfully reduces the scene flow error.

Baselines: We compare against state-of-the-art monocular scene flow baselines. **CC** [199] and **SSM** [87] are representative methods for self-supervised monocular depth and scene flow estimation that does not make use of segmentation priors at inference time. **Mono-SF** [30] trains a monocular depth network with KITTI ground-truth, and solve an optimization problem given semantic instance segmentation provided by Mask R-CNN. The above three methods are trained on KITTI and the results are taken from their papers. **OE** (optical expansion) [302] learns to predict relative depth from dense optical expansion, which together with optical flow, directly yields 3D motion. It is trained on the synthetic SceneFlow dataset, and we use MiDaS to provide the scale. We also implement a baseline (**MiDaS+Mask**) that predicts instance segmentation masks by Mask R-CNN, and follows the same rigid body fitting procedure as ours.

Performance analysis: We report results on KITTI-SF and Sintel in Tab. 8.2. First, it is noted our method reduces the disparity error of MiDaS by more than 50% on KITTI, and 20% on Sintel. Compared to OE, which uses the same monocular depth input as ours, we are better in all metrics. (SF: 22.19% vs 44.87%), which demonstrates the effectiveness of our rigid motion mask. Our method also outperforms the other methods that do not use segmentation priors (CC and SSM). Compared to Mono-SF, which is trained with ground-truth KITTI depth maps, and uses a semantic segmentation prior, our method is slightly worse on KITTI. Compared to Midas-Mask, our method is strictly better on both KITTI and Sintel, indicating the

Table 8.3: Stereo scene flow results on KITTI benchmark. D1 and D2: first and second frame disparity error. Fl: optical flow error. SF: scene flow error. Metrics are errors in percentage and top results are in bold. *First frame disparity is not refined by our method.

Method	D1* ↓	D2 ↓	Fl ↓	SF ↓
PRSM [268]	4.27	6.79	6.68	8.97
OpticalExp [302]	1.81	4.25	6.30	8.12
DRISF [152]	2.55	4.04	4.73	6.31
Ours Mask R-CNN	1.89	3.42	4.26	5.61
Ours Rigid Mask	1.89	**3.23**	**3.50**	**4.89**

benefit of using our rigid motion masks versus appearance-based masks.

8.4.3 Stereo Scene Flow

Our method is also able to take advantage of reliable depth sensors, such as stereo cameras, to produce better two-frame rigid motion segmentation and scene flow estimations. To take advantage of stereo inputs, we make two algorithmic changes. First, we triangulate stereo disparities as the depth input to the segmentation network. Second, we refine each rigid body transformation by solving a perspective-n-point problem (via LM optimization):

$$\min_{(\mathbf{R}_i,\mathbf{T}_i)} \sum_j ||\mathbf{p}_{i,j}{}' - \pi_{\mathbf{K}}(\mathbf{R}_i\mathbf{P}_{i,j} + \mathbf{T}_i)||^2, \tag{8.11}$$

where $\pi(\cdot)$ is a projection function, $\mathbf{P}_{i,j}$ is the j-th point from the i-th rigid body, and $\mathbf{p}'_{i,j}$ is the second frame flow correspondence. We use the results of Alg. 1 as initial values and update rigid body transformations for 20 iterations.

Implementation: We use off-the-shelf networks that are fine-tuned on KITTI-SF to estimate stereo disparity and optical flow [302, 321]. We also fine-tune our rigid motion segmentation network by mixing KITTI-SF and SceneFlow datasets. The performance is reported on the KITTI benchmark.

Baselines: Our method segments rigid motions based on two-frame rigidity and fits rigid body transformations over depth-flow correspondences, which is used to update the second frame depth and flow estimations. Among the baselines, **OE** [302] uses the same architecture (as in the monocular setup) fine-tuned on KITTI to upgrade optical

Table 8.4: Diagnostics of rigid body motion segmentation on KITTI-SF. Dignostics in the second group are sequential.

Method	K: obj ↑	K: bg ↑	S: bg ↑
Reference	**89.53**	**97.22**	**84.63**
(1)w/o cost maps	88.66	96.59	76.81
(2)w/o uncertainty	85.09	95.72	77.25
(3)w/o monocular depth	84.46	94.84	76.14
(4)w/o expansion (MoA [21])	81.28	95.50	77.00
(5)w/o learning [20, 286]	25.83	85.52	74.23

flow to 3D scene flow. Same as ours, GA-Net stereo and VCN optical flow are used as inputs. **PRSM** [268] segments an image into superpixels, and fits rigid motions to estimate piece-wise rigid scene flow. Given semantic instance segmentation [81], depth, and optical flow, **DRISF** [152] casts scene flow estimation as an energy minimization problem and finds the best rigid transformation for each *semantic* instance. The key difference between our method and DRISF is that we use rigid motion segmentation masks.

Performance analysis: As reported in Tab. 8.3, our method demonstrates state-of-the-art performance on KITTI scene flow benchmark (SF: 4.89 vs 6.31). If we replace the segmentation masks with semantic instance segmentation, i.e., Mask R-CNN, the performance drops noticeably (SF: 4.89% to 5.61%). As illustrated in Fig. 8.5, our method successfully groups the static objects (e.g. parked cars) with the rigid background, which effectively improves scene flow accuracy by optimizing the whole background as one rigid body, while semantic instance segmentation methods fail to do so.

8.4.4 Diagnostics

We ablate critical components of our approach and retrain networks. Results are shown in Tab. 8.4. We validate the design choices of using (1)explicitly computed rigidity cost-maps inputs, (2)uncertainty estimation inputs, (3)monocular depth inputs, (4)optical expansion that upgrades 2D optical flow to 3D, and (5) our rigid motion segmentation network. (1)Removing rigidity cost-maps leads to a slight drop of accuracy on KITTI, and a significant drop on Sintel (84.63% to 76.81%). This

indicates the cost map features are crucial for Sintel, possibly due to complex camera and object motion configurations, in which cases explicit geometric priors are helpful. (2)Removing uncertainty inputs leads to a noticeable drop of performance on KITTI (88.66% to 85.09%). We posit uncertainty estimation contains rich information about motion distribution, and is therefore useful for segmentation. (3)Further removing monocular depth inputs leads to an accuracy drop on all metrics, especially on KITTI, which shows the importance of using depth cues to deal with collinear motions in autonomous driving scenes. (4)After further removing optical expansion, our method degrades to MoA-Net [21]. The performance drops noticeably on KITTI rigid instance segmentation metric (84.46% to 81.28%), which indicates optical expansion is useful for segmenting foreground objects. (5)Lastly, if we directly apply the rigidity cost maps with manually-tuned thresholds to decide the background region without the neural architecture and learning, the method becomes worse in all metrics due to the loss of robustness to noisy inputs and degenerate motion.

8.5 Conclusion

We investigate the problem of two-frame rigid body motion segmentation in an open environment. We analyze the degenerate cases in geometric motion segmentation and introduce novel criteria and inputs to resolve such ambiguities. We further propose a modular neural architecture that is robust to noisy observations as well as different motion types, which demonstrates state-of-the-art performance on rigid motion segmentation, depth and scene flow estimation tasks.

Part V

Conclusion and Future works

In this thesis, we explore building 4D models of generic deformable objects and dynamic scenes from monocular videos. We start by reconstructing deformable objects from a single video and gradually move to multiple videos, different instances, and dynamic scenes. Finally, we show that offline-reconstructed 4D models can be distilled into efficient neural architectures for real-time inference. Beyond what we have shown, more work needs to be done to improve the quality, robustness, and efficiency. We summarize the lessons we learned and point out potential future directions.

1. **Rotation registration**. In the works we have discussed, the most common cause of optimization instability is object rotation (a.k.a. viewpoint) errors. This coincides with the classic analysis of shape and motion ambiguities [3, 245], which is magnified in the nonrigid case. Even with a carefully designed optimization routine, we find it is important to properly initialize the camera rotation (e.g., less than 60 degrees of error) to ensure convergence. In practice, we train viewpoint estimators to initialize rotation for humans and quadruped animals and use sparse manual annotation for the other categories. Although this works well for common cases, it would be interesting to develop a method to register arbitrary objects with as few manual annotations as possible. One approach is to train a generic viewpoint estimator [323] to initiate the camera rotations. Another approach is to improve the modeling such that the optimization is robust to rotation errors.
2. **Fine-level reconstruction**. Our method reaches state-of-the-art reconstruction quality for cats, dogs, and humans in terms of mid-level reconstruction, but fine details are still missing, such as face and hands. This is expected as for one, we use low-resolution (256x256) images of color, flow, and features etc. to keep storage manageable; and second, although important for social interaction, those body parts only account for a small portion of the input pixels, which might not contain enough information to recover the details. Besides scaling up the input resolution, one thought is to leverage existing models trained for specific parts [104]. A seemingly more scalable approach is to fine-tune a pre-trained full-body model with domain-specific data, such as imagery of hands and faces.
3. **Scaling up.** We have shown the possibility of distilling offline-optimized 4D

models into efficient feed-forward architectures. Compared with our optimization-based methods, feed-forward models are hundreds of times faster but less accurate. To improve the quality of feedforward models, one approach is to perform test-time optimization [119], which finds a middle ground between the two paradigms. Another approach is to scale up the training data, which urges a better design of the optimization pipeline and requires more system efforts.

4. **Unified models.** We have shown the results of reconstructing individual categories of humans, cats, dogs, and cars. However, there are outliers instances that appear on the semantic boundary of different categories. For instance, a human wearing a cat costume could be classified as a cat or a human. This argues for a unified representation where the model could interpolate attributes of different categories to borrow shared structures, which would potentially improve robustness at test time.

5. **Physical models.** In PPR [310], we couple differentiable rendering with differentiable physics simulation and build a pipeline to estimate physics quantities from monocular videos. Our choice of the physical model (e.g., rigid body dynamics and contact) allows for efficient forward simulation, but at the same time, limits our methods to specific scenarios (terrestrial creatures making contact with the ground). Although it seems promising to apply PPR to other scenarios (e.g., soft bodies, marine animals) by replacing the forward physics model, it remains an open problem how to extend the forward simulation to handle generic dynamics and contact, as they are governed by different types of physics equations.

Outlook. Beyond geometry and motion, there are higher-order quantities that govern the generation process of the world we observe every day, such as forces, the intent of agents, and even neural activities. Building models that faithfully represent the structures and dynamics of the world is a grand goal that would revolutionize the way we live and work. Although solving these latent structures is challenging due to their high-dimensional and unobservable nature, analysis-by-synthesis with differentiable engines and data-driven priors seems a promising way to constrain the problem, which leverages large scale less-structured data and a deep understanding of their internal mechanism.

Appendix A

Notations

A.1 Notations for Chapter. 2

A summary of the notations is listed in Tab. A.1.

A.2 Notations for Chapter. 4

We refer readers to a list of notations in Tab. A.2 and a list of learnable parameters in Tab. A.3.

Table A.1: Table of notations used in this work.

Symbol	Description
	Measurements $\mathbf{Y}^*$
I_t	Input RGB image at time t
S_t	Input or measured object silhouette image at time t
$\mathbf{u}_t^+$	Input or measured forward optical flow map from time t to $t+1$
$\mathbf{u}_t^-$	Input or measured backward optical flow map from time t to $t-1$
	Renderings Y
$\hat{I}_t$	Rendered color image of the object at time t
$\hat{S}_t$	Rendered object silhouette image at time t
$\hat{\mathbf{u}}_t^+$	Rendered forward optical flow map of the object from time t to $t+1$
$\hat{\mathbf{u}}_t^-$	Rendered backward optical flow map of the object from time t to $t-1$
	Variables
f_t	Focal length of the camera at time t
$\mathbf{K_t}$	Intrinsic matrix of a simple pinhole camera (with zero skew and square pixel) at time t
$\mathbf{R_{0,t}}$	Object root body rotation matrix $\in SO(3)$ at time t
$\mathbf{T_{0,t}}$	Object root body translation vector at time t
$\mathbf{G_{0,t}}$	Object root body transformation at time t, $\mathbf{G_{0,t}} = \begin{pmatrix}[c\|c]\mathbf{R_0} & \mathbf{T_0}\end{pmatrix}_\mathbf{t}$
$\mathbf{R_{1\ldots B,t}}$	Bone rotations from the rest pose to time t
$\mathbf{T_{1\ldots B,t}}$	Bone rotations from the rest pose to time t
$\mathbf{G_{1\ldots B,t}}$	Bone transformations from the rest pose to time t, $\mathbf{G_{i,t}} = \begin{pmatrix}[c\|c]\mathbf{R_i} & \mathbf{T_i}\end{pmatrix}_\mathbf{t}, i \in \{1\ldots,B\}$
$\mathbf{D_t}$	Union of camera and bone transformations $\{\mathbf{G_{0,t}}, \ldots, \mathbf{G_{B,t}}\}$
$\mathbf{P_t}$	Projection matrix of the camera at time t, $\mathbf{P_t} = \mathbf{K_t}\mathbf{G_{0,t}}$
$\Delta\mathbf{V_t}$	Vertex motion from the rest shape to time t
	Parameters X
(p_x, p_y)	Principal point of the camera
$\bar{\mathbf{V}}_i$	Position of the i-th vertex of the mesh in the rest pose (or mean shape)
$\bar{\mathbf{C}}_i$	Color of the i-th vertex of the mesh
$\mathbf{S}$	Union of all mesh parameters, $\mathbf{S} = \{\bar{\mathbf{V}}, \bar{\mathbf{C}}, \mathbf{F}\}$
$\mathbf{J}_b$	Position of the center of the b-th bone (or Gaussian component)
$\mathbf{Q}_b$	Precision matrix of b-th bone (or Gaussian component)
$\mathbf{W}$	Skinning weights matrix, $\mathbf{W}$=$\{\mathbf{J}, \mathbf{Q}\}$
ϕ_w	Weights of the convolutional camera and pose network
$\mathbf{n}^*$	Normal vector of the symmetry plane in the canonical frame
	Constants
T	Number of frames in the input video
M	Number of faces in the mesh
N	Number of vertices in the mesh
B	Number of bones for LBS
$\mathbf{F}$	Faces of the mesh
$\boldsymbol{\beta}$	Weights between the losses
$\mathbf{H}$	Householder transformation matrix describing reflection about the y-z plane
$\mathbf{n}_0$	Unit vector towards to the x axis
	Others
S0	Training stage 0: optimize for $\{\phi_w(f_t, \mathbf{G_{0,t}}), p_x, p_y, \mathbf{n}^*, \bar{\mathbf{V}}, \bar{\mathbf{C}}\}$
S1-3	Training stage 1 to 3: optimize for $\{\phi_w(f_t, \mathbf{G_{0\ldots B,t}}), p_x, p_y, \mathbf{n}^*, \bar{\mathbf{V}}, \bar{\mathbf{C}}, \mathbf{J}, \mathbf{Q}\}$

Table A.2: 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 $\mathbf{x} = (\text{x,y})$
$\mathbf{X}^t$	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 A.3: 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
$\mathbf{\Lambda}^0 \in \mathbb{R}^{3\times3}$	Scale of the bones in the zero-configuration (diagonal matrix).
$\mathbf{V}^0 \in \mathbb{R}^{3\times3}$	Orientation of the bones in the zero-configuration.
$\mathbf{C}^0 \in \mathbb{R}^3$	Center of the bones in the zero-configuration.
$\mathbf{MLP}_\Delta$	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_s	Temperature scalar for canonical feature matching
β	Scale parameter that controls the solidness of the object surface
$\mathbf{\Pi} \in \mathbb{R}^{3\times3}$	Intrinsic matrix of the pinhole camera model

Bibliography

[1] Carnegie-mellon mocap database. http://mocap.cs.cmu.edu. 1.1

[2] Robust vision challenge. www.robustvision.net, 2020. 3.4.2

[3] Gilad Adiv. Inherent ambiguities in recovering 3-d motion and structure from a noisy flow field. *PAMI*, 11(5):477–489, 1989. 1

[4] 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. 4.1

[5] Gerald J Agin and Thomas O Binford. Computer description of curved objects. *IEEE Transactions on Computers*, (4):439–449, 1976. 8.1

[6] Antonio Agudo, Melcior Pijoan, and Francesc Moreno-Noguer. Image collection pop-up: 3d reconstruction and clustering of rigid and non-rigid categories. In *CVPR*, pages 2607–2615, 2018. 2.1, 2.2

[7] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. In *ACM SIGGRAPH 2005 Papers*, pages 408–416. 2005. 5.1

[8] Amado Antonini, Winter Guerra, Varun Murali, Thomas Sayre-McCord, and Sertac Karaman. The blackbird dataset: A large-scale dataset for uav perception in aggressive flight. In *2018 International Symposium on Experimental Robotics (ISER)*, 2018. 7.4.6, 7.11

[9] 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. 2.2, 4.2, 5.2, 6.2

[10] Simon Baker, Takeo Kanade, et al. Shape-from-silhouette across time part i: Theory and algorithms. *IJCV*, 62(3):221–247, 2005. 2.1

[11] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 92(1):1–31, 2011. 7.3.3, 7.10

[12] Praneet C Bala, Benjamin R Eisenreich, Seng Bum Michael Yoo, Benjamin Y

Hayden, Hyun Soo Park, and Jan Zimmermann. Openmonkeystudio: Automated markerless pose estimation in freely moving macaques. *BioRxiv*, pages 2020–01, 2020. 1.1

[13] Daniel Barath. Recovering affine features from orientation-and scale-invariant ones. In *ACCV*, 2018. 7.2, 7.3.3

[14] Daniel Barath and Zuzana Kukelova. Homography from two orientation-and scale-covariant features. In *ICCV*, 2019. 7.2

[15] Ioan Andrei Bârsan, Peidong Liu, Marc Pollefeys, and Andreas Geiger. Robust dense mapping for large-scale dynamic environments. In *ICRA*, 2018. 8.1

[16] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *ECCV*, 2006. 7.2

[17] Christian Beder and Richard Steffen. Determining an initial image pair for fixing the scale of a 3d reconstruction from an image sequence. In *Joint Pattern Recognition Symposium*, pages 657–666. Springer, 2006. 7.4.5

[18] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger. Bounding boxes, segmentations and object coordinates: How important is recognition for 3d scene flow estimation in autonomous driving scenarios? In *ICCV*, 2017. 8.2

[19] Abhijit Bendale and Terrance Boult. Towards open world recognition. In *CVPR*, 2015. 8.1, 8.2

[20] Pia Bideau and Erik Learned-Miller. It's moving! a probabilistic model for causal motion segmentation in moving camera videos. In *ECCV*, 2016. 8.2, 8.3.1, 8.4, 8.1, 8.4.1, 8.4

[21] Pia Bideau, Rakesh R Menon, and Erik Learned-Miller. Moa-net: self-supervised motion segmentation. In *ECCVW*, 2018. 8.2, 8.4, 8.4.4

[22] Pia Bideau, Aruni RoyChowdhury, Rakesh R Menon, and Erik Learned-Miller. The best of both worlds: Combining cnns and geometric constraints for hierarchichal motion segmentation. In *CVPR*, 2018. 8.2

[23] Irving Biederman. Geon theory as an account of shape recognition in mind and brain. *The Irish Journal of Psychology*, 14(3):314–327, 1993. 8.1

[24] Benjamin Biggs, Thomas Roddick, Andrew Fitzgibbon, and Roberto Cipolla. Creatures great and smal: Recovering the shape and motion of animals from video. In *ACCV*, pages 3–19. Springer, 2018. 2.1, 2.1, 2.4.1, 2.3, 2.4, 3.4.1, 3.4.2, 5.1

[25] 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. 2.1, 2.2, 4.2, 5.2, 6.2

[26] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. Faust: Dataset and evaluation for 3d mesh registration. In *CVPR*, pages 3794–3801, 2014. 5.1

[27] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J. Black. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *ECCV*, 2016. 2.1, 3.3.3

[28] Eric Brachmann and Carsten Rother. Neural- Guided RANSAC: Learning where to sample model hypotheses. In *ICCV*, 2019. 8.3.2

[29] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. In *CVPR*, 2000. 2.1, 3.1, 3.2, 4.1, 4.2, 6.2

[30] Fabian Brickwedde, Steffen Abraham, and Rudolf Mester. Mono-SF: Multi-view geometry meets single-view depth for monocular scene flow estimation of dynamic traffic scenes. In *CVPR*, 2019. 7.1, 7.2, 8.2, 8.4.2

[31] Andrés Bruhn, Joachim Weickert, and Christoph Schnörr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *IJCV*, 61(3): 211–231, 2005. 1.3

[32] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *ECCV*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, October 2012. 3.2

[33] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012. 7.3.3, 8.4

[34] Jeffrey Byrne and Camillo J Taylor. Expansion segmentation for visual collision detection and estimation. In *ICRA*, 2009. 7.1, 7.4.3

[35] Ted Camus, David Coombs, Martin Herman, and Tsai-Hong Hong. Real-time single-workstation obstacle avoidance using only wide-field flow divergence. In *Proceedings of 13th International Conference on Pattern Recognition*, volume 3, pages 323–330, 1996. 7.2, 7.3.3

[36] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *TPAMI*, 2019. 2.4.2

[37] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 1.3

[38] Thomas J Cashman and Andrew W Fitzgibbon. What shape are dolphins?

building 3d morphable models from 2d images. *PAMI*, 35(1):232–244, 2012. 2.1, 2.1

[39] Eric Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *arXiv*, 2020. 5.2

[40] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *arXiv*, 2021. 5.2

[41] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, pages 5799–5809, 2021. 5.3.1

[42] Xu Chen, Yufeng Zheng, Michael J Black, Otmar Hilliges, and Andreas Geiger. Snarf: Differentiable forward skinning for animating non-rigid neural implicit shapes. 2021. 4.1, 4.1, 4.3.2

[43] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. In *NeurIPS*. 2016. 2.1

[44] Yuchao Dai, Hongdong Li, and Mingyi He. A simple prior-free method for non-rigid structure-from-motion factorization. *IJCV*, 107(2):101–122, 2014. 1.2, 2.1, 2.2, 2.3.2, 6.1

[45] Trevor Darrell and Alex P Pentland. Cooperative robust estimation using layers of support. *TPAMI*, 17(5):474–487, 1995. 3.2

[46] Achal Dave, Pavel Tokmakov, and Deva Ramanan. Towards segmenting everything that moves. In *ICCVW*, 2019. 8.4.1

[47] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009. 2.1, 2.3.4

[48] Ayush Dewan, Tim Caselitz, Gian Diego Tipaldi, and Wolfram Burgard. Rigid scene flow for 3d lidar scans. In *IROS*, 2016. 7.4.2

[49] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 5.3.2, 6.3.2

[50] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 7.3.3, 8.4

[51] Elan Dubrofsky. Homography estimation. *Diplomová práce. Vancouver: Univerzita Britské Kolumbie*, 2009. 8.3.2

[52] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping:

part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006. 8.1

[53] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *ICCV*, pages 10786–10796, 2021. 5.4.1, 6.3.1

[54] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *ECCV*, pages 834–849. Springer, 2014. 1.2, 3.1

[55] Andreas Ess, Bastian Leibe, and Luc Van Gool. Depth and appearance for mobile scene analysis. In *ICCV*, 2007. 7.2

[56] Jean Feydy, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouve, and Gabriel Peyré. Interpolating between optimal transport and mmd using sinkhorn divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2681–2690, 2019. 4.4, 5.3.4

[57] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 8.2

[58] Pete Florence, John Carter, and Russ Tedrake. Integrated perception and control at high speed: Evaluating collision avoidance maneuvers without maps. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2016. 7.4.3

[59] Wolfgang Förstner. Uncertainty and projective geometry. In *Handbook of Geometric Computing*, pages 493–534. Springer, 2005. 7.4.5

[60] C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax - a differentiable physics engine for large scale rigid body simulation, 2021. URL `http://github.com/google/brax`. 6.2

[61] Ravi Garg, Anastasios Roussos, and Lourdes Agapito. A variational approach to video registration with subspace constraints. *IJCV*, 2013. 2.4.1, 3.4.2

[62] Erik Gärtner, Mykhaylo Andriluka, Erwin Coumans, and Cristian Sminchisescu. Differentiable dynamics for articulated 3d human motion reconstruction. In *CVPR*, pages 13190–13200, 2022. 6.1, 6.2, 6.3.4, 6.4.2, 6.4.3

[63] Erik Gärtner, Mykhaylo Andriluka, Hongyi Xu, and Cristian Sminchisescu. Trajectory optimization for physics-based reconstruction of 3d human pose from monocular video. In *CVPR*, pages 13106–13115, 2022. 6.3.4, 6.4.3

[64] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 8.4

[65] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, pages 3354–3361. IEEE, 2012. 1.3.4, 3.2

[66] Moritz Geilinger, David Hahn, Jonas Zehnder, Moritz Bächer, Bernhard

Thomaszewski, and Stelian Coros. Add: Analytically differentiable dynamics for multi-body systems with frictional contact. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020. 6.3.3

[67] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *CVPR*, pages 4857–4866, 2020. 2.3.2

[68] Justin Johnson Georgia Gkioxari, Jitendra Malik. Mesh r-cnn. *ICCV*, 2019. 2.4.2

[69] Partha Ghosh, Medhi SM Sajjadi, Antonio Vergari, and Michael Black. From variational to deterministic autoencoders. In *ICLR*, 2020. 5.4.3

[70] James J Gibson. *The ecological approach to visual perception: classic edition*. Psychology Press, 2014. 7.1

[71] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *ICCV*, pages 9785–9795, 2019. 2.1

[72] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. In *ICCV*, 2019. 7.4.1, 7.9, 7.11, 8.4.1

[73] Shubham Goel, Angjoo Kanazawa, and Jitendra Malik. Shape and viewpoints without keypoints. In *ECCV*, 2020. 2.1, 2.1, 2.2, 2.3.2, 3.2, 4.2, 5.2, 6.2

[74] Paulo FU Gotardo and Aleix M Martinez. Non-rigid structure from motion with complementary rank-3 spaces. In *CVPR*, 2011. 2.1, 2.2, 3.2, 4.2, 6.2, 7.2

[75] Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang. HPLFlowNet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds. In *CVPR*, 2019. 7.4.2, 7.3

[76] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *CVPR*, pages 7297–7306, 2018. 1.3, 3.1, 3.2, 3.3.2

[77] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, volume 2, pages 1735–1742. IEEE, 2006. 3.3.3

[78] Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. Point2mesh: A self-prior for deformable meshes. *arXiv preprint arXiv:2005.11084*, 2020. 2.3.4

[79] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 6.3.3, 7.3.1, 7.4.5, 8.1, 8.3.1, 8.2, 8.3.1, 8.3.1, 8.3.2, 8.3.2

[80] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 2.3.4, 2.2

[81] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 8.4, 8.4.1, 8.4.3

[82] Eric Heiden, David Millard, Erwin Coumans, Yizhou Sheng, and Gaurav S Sukhatme. NeuralSim: Augmenting differentiable simulators with neural networks. In *ICRA*, 2021. URL https://github.com/google-research/tiny-differentiable-simulator. 6.2

[83] Berthold KP Horn, Yajun Fang, and Ichiro Masaki. Time to contact relative to a planar surface. In *IEEE Intelligent Vehicles Symposium*, 2007. 7.2, 7.4.3

[84] Taylor A Howell, Simon Le Cleac'h, J Zico Kolter, Mac Schwager, and Zachary Manchester. Dojo: A differentiable simulator for robotics. *arXiv preprint arXiv:2203.00806*, 2022. 6.2

[85] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable programming for physical simulation. *ICLR*, 2020. 6.2, 6.4.3

[86] Jingwei Huang, Hao Su, and Leonidas Guibas. Robust watertight manifold surface generation method for shapenet models. *arXiv preprint arXiv:1802.01698*, 2018. 2.3.4

[87] Junhwa Hur and Stefan Roth. Self-supervised monocular scene flow estimation. In *CVPR*, 2020. 8.2, 8.4.2

[88] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017. 7.4

[89] Eddy Ilg, Ozgun Cicek, Silvio Galesso, Aaron Klein, Osama Makansi, Frank Hutter, and Thomas Brox. Uncertainty estimates and multi-hypotheses networks for optical flow. In *ECCV*, 2018. 8.3.2

[90] Eddy Ilg, Tonmoy Saikia, Margret Keuper, and Thomas Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *ECCV*, 2018. 7.1, 7.4.1, 7.4.2, 7.3, 7.4

[91] Michal Irani and P Anandan. A unified approach to moving object detection in 2d and 3d scenes. *PAMI*, 1998. 8.1, 8.3.1

[92] Alec Jacobson and Olga Sorkine. Stretchable and twistable bones for skeletal shape deformation. In *SIGGRAPH Asia*, pages 1–8, 2011. 5.3.1

[93] Alec Jacobson, Zhigang Deng, Ladislav Kavan, and JP Lewis. Skinning: Real-time shape deformation. In *ACM SIGGRAPH 2014 Courses*, 2014. 4.3.2

[94] Yasamin Jafarian and Hyun Soo Park. Learning high fidelity depths of dressed humans by watching social media dance videos. In *CVPR*, pages 12753–12762, June 2021. 4.2

[95] Ramesh Jain and H-H Nagel. On the analysis of accumulative difference pictures from image sequences of real world scenes. *TPAMI*, (2):206–214, 1979. 5.3.3

[96] Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos. In *CVPR*, 2017. 8.2, 8.4.1, 8.1

[97] Wonbong Jang and Lourdes Agapito. Codenerf: Disentangled neural radiance fields for object categories. In *ICCV*, pages 12949–12958, October 2021. 5.3.1

[98] Krishna Murthy Jatavallabhula, Miles Macklin, Florian Golemo, Vikram Voleti, Linda Petrini, Martin Weiss, Breandan Considine, Jérôme Parent-Lévesque, Kevin Xie, Kenny Erleben, et al. gradsim: Differentiable simulation for system identification and visuomotor control. *ICLR*, 2021. 6.4.3

[99] Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-calibrating neural radiance fields. In *ICCV*, 2021. 4.2

[100] Allan Jepson and Michael J Black. Mixture models for optical flow computation. In *CVPR*, pages 760–761. IEEE, 1993. 3.2

[101] Boyi Jiang, Yang Hong, Hujun Bao, and Juyong Zhang. Selfrecon: Self reconstruction your digital avatar from monocular video. In *CVPR*, 2022. 5.2

[102] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video. In *ECCV*, pages 402–418. Springer, 2022. 6.6, 6.4.3, 6.2

[103] Hanbyul Joo, Tomas Simon, Xulong Li, Hao Liu, Lei Tan, Lin Gui, Sean Banerjee, Timothy Godisart, Bart Nabbe, Iain Matthews, et al. Panoptic studio: A massively multiview system for social interaction capture. *TPAMI*, 41(1):190–204, 2017. 1.1, 3.1, 3.2

[104] Hanbyul Joo, Tomas Simon, and Yaser Sheikh. Total capture: A 3d deformation model for tracking faces, hands, and bodies. In *CVPR*, pages 8320–8329, 2018. 2

[105] Navami Kairanda, Edith Tretschk, Mohamed Elgharib, Christian Theobalt, and Vladislav Golyanik. f-sft: Shape-from-template with a physics-based deformation model. In *CVPR*, pages 3948–3958, 2022. 6.2

[106] Takeo Kanade, Peter Rander, and P.J. Narayanan. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE Multimedia*, 1997. 1.1

[107] Angjoo Kanazawa, Michael J Black, David W Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018. 1.1

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

ECCV, 2018. 2.1, 2.1, 2.2, 2.3, 2.3.2, 2.3.3, 2.4.1, 3.2, 3.3, 3.3.3, 4.2, 5.2, 6.1, 6.2

[109] Dongho Kang, Simon Zimmermann, and Stelian Coros. Animal gaits on quadrupedal robots using motion matching and model-based control. In *IROS*, pages 8500–8507. IEEE, 2021. 1.1

[110] Dongho Kang, Flavio De Vincenti, Naomi C Adami, and Stelian Coros. Animal motions on legged robots using nonlinear model predictive control. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11955–11962. IEEE, 2022. 1.1

[111] Ladislav Kavan, Steven Collins, Jiří Žára, and Carol O'Sullivan. Skinning with dual quaternions. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, pages 39–46, 2007. 5.3.2

[112] 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. 3.3.2, 3.3.3, 4.3.3, 7.5

[113] Sameh Khamis, Sean Fanello, Christoph Rhemann, Adarsh Kowdle, Julien Valentin, and Shahram Izadi. Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. In *ECCV*, 2018. 7.5

[114] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2.2

[115] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. PointRend: Image segmentation as rendering. 2019. 6.3.4

[116] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *CVPR*, 2020. 1.3.1, 2.3.4, 2.4.1, 4.4.1, 5.4.1

[117] Muhammed Kocabas, Nikos Athanasiou, and Michael J. Black. Vibe: Video inference for human body pose and shape estimation. In *CVPR*, June 2020. 1.1, 2.1, 2.2, 2.4, 2.4.2, 3.1, 3.2, 3.1, 3.4.1, 4.2, 5.2, 6.2

[118] Filippos Kokkinos and Iasonas Kokkinos. To the point: Correspondence-driven monocular 3d category reconstruction. In *NeurIPS*, 2021. 4.2, 5.2, 6.2

[119] Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In *ICCV*, 2019. 3

[120] Daniel Kondermann, Rahul Nair, Katrin Honauer, Karsten Krispin, Jonas Andrulis, Alexander Brock, Burkhard Gussefeld, Mohsen Rahimimoghaddam, Sabine Hofmann, Claus Brenner, et al. The hci benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving. In *CVPRW*, 2016. 7.3.3, 8.4

[121] Chen Kong and Simon Lucey. Deep non-rigid structure from motion. In *ICCV*, 2019. 2.1, 2.2, 3.2, 4.2, 6.2

[122] Nilesh Kulkarni, Abhinav Gupta, David F Fouhey, and Shubham Tulsiani. Articulation-aware canonical surface mapping. In *CVPR*, pages 452–461, 2020. 2.1, 2.1, 2.2, 2.3.2, 2.3.2, 2.4.1, 2.3, 2.4, 3.1, 3.2, 3.3.1, 3.3.2, 3.3.2, 3.3.3, 3.4, 3.4.4, 4.2, 4.3.4

[123] Suryansh Kumar. Non-rigid structure from motion: Prior-free factorization method revisited. In *WACV*, 2020. 4.2, 6.2

[124] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Monocular dense 3d reconstruction of a complex dynamic scene from two perspective frames. In *ICCV*, 2017. 7.1, 7.2

[125] Binh Huy Le and Zhigang Deng. Robust and accurate skeletal rigging from mesh sequences. *ACM Transactions on Graphics (TOG)*, 33(4):1–10, 2014. 5.3.1

[126] Jiahui Lei and Kostas Daniilidis. Cadex: Learning canonical deformation coordinate space for dynamic surface representation via neural homeomorphism. In *CVPR*, 2022. URL https://cis.upenn.edu/~leijh/projects/cadex. 5.3.2, 5.3.2, 6.3.2, 6.4

[127] John P Lewis, Matt Cordner, and Nickson Fong. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 165–172, 2000. 2.3.2, 3.3.1

[128] Ruilong Li, Kyle Olszewski, Yuliang Xiu, Shunsuke Saito, Zeng Huang, and Hao Li. Volumetric human teleportation. In *ACM SIGGRAPH 2020 Real-Time Live!*, pages 1–1. 2020. 5.2

[129] Ruilong Li, Julian Tanke, Minh Vo, Michael Zollhofer, Jurgen Gall, Angjoo Kanazawa, and Christoph Lassner. Tava: Template-free animatable volumetric actors. 2022. 5.2

[130] 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. 2.1, 2.2, 3.2, 4.2, 5.2, 6.2

[131] 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. 2.1, 2.1, 2.2, 2.3.2, 2.4.1, 2.3, 2.4, 3.2, 4.2, 5.2, 6.2

[132] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *CVPR*, 2021. 4.1, 4.2,

4.3.1, 4.3.4, 4.4.1, 4.4.3, 5.3.2, 6.3.2

[133] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *ICCV*, 2021. 4.2

[134] Shih-Chieh Lin, Yunqi Zhang, Chang-Hong Hsu, Matt Skach, Md E Haque, Lingjia Tang, and Jason Mars. The architectural implications of autonomous driving: Constraints and acceleration. In *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 751–766, 2018. 8.1

[135] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 2.1, 3.4.1, 8.4.1

[136] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 8.3.2

[137] C Karen Liu and Sumit Jain. A quick tutorial on multibody dynamics. *Online tutorial, June*, page 7, 2012. 6.3.3

[138] 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. 4.2

[139] Pengpeng Liu, Michael Lyu, Irwin King, and Jia Xu. Selflow: Self-supervised learning of optical flow. In *CVPR*, 2019. 7.3.3

[140] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *ICCV*, 2019. 1.3, 2.3, 2.3.1, 3.3, 3.3.1, 3.3.3

[141] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3D: Learning scene flow in 3d point clouds. In *CVPR*, 2019. 7.4.2, 7.3

[142] Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. MeteorNet: Deep learning on dynamic 3d point cloud sequences. In *ICCV*, 2019. 7.4, 7.4.2, 7.3

[143] Hugh Christopher Longuet-Higgins and Kvetoslav Prazdny. The interpretation of a moving retinal image. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 208(1173):385–397, 1980. 7.4.5

[144] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *SIGGRAPH Asia*, 2015. 1.1, 1.2, 2.1, 2.1, 2.2, 3.1, 3.2, 4.2, 5.1, 5.2, 5.3.1, 6.2

[145] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH*, 21(4):163–169, 1987. 5.3.4, 6.3.3

[146] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 3.4

[147] DG Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999. 7.2

[148] Xiankai Lu, Wenguan Wang, Chao Ma, Jianbing Shen, Ling Shao, and Fatih Porikli. See more, know more: Unsupervised video object segmentation with co-attention siamese networks. In *CVPR*, 2019. 8.2, 8.4.1, 8.1

[149] Diogo C Luvizon, Hedi Tabia, and David Picard. Human pose regression by combining indirect part detection and contextual information. *Computers & Graphics*, 2019. 4.3.3

[150] Zhaoyang Lv, Kihwan Kim, Alejandro Troccoli, Deqing Sun, James Rehg, and Jan Kautz. Learning rigidity in dynamic scenes with a moving camera for 3d motion field estimation. In *ECCV*, 2018. 8.4.1, 8.1, 8.4.1

[151] Pingchuan Ma, Tao Du, Joshua B Tenenbaum, Wojciech Matusik, and Chuang Gan. Risp: Rendering-invariant state predictor with differentiable simulation and rendering for cross-domain parameter estimation. *arXiv preprint arXiv:2205.05678*, 2022. 6.3.4

[152] Wei-Chiu Ma, Shenlong Wang, Rui Hu, Yuwen Xiong, and Raquel Urtasun. Deep rigid instance scene flow. In *CVPR*, 2019. 7.2, 8.2, 8.3, 8.4.3

[153] Miles Macklin. Warp: A high-performance python framework for gpu simulation and graphics. `https://github.com/nvidia/warp`, March 2022. NVIDIA GPU Technology Conference (GTC). 1.3, 6.2, 6.3.3, 6.4

[154] Aashi Manglik, Xinshuo Weng, Eshed Ohn-Bar, and Kris M Kitani. Future near-collision prediction from monocular video: Feasibility, dataset, and challenges. *IROS*, 2019. 7.1, 7.2, 7.4.3

[155] Thiago Marinho, Massinissa Amrouche, Venanzio Cichella, Dušan Stipanović, and Naira Hovakimyan. Guaranteed collision avoidance based on line-of-sight angle and time-to-collision. In *2018 Annual American Control Conference (ACC)*, 2018. 7.1, 7.4.3

[156] David Marr and Tomaso Poggio. A computational theory of human stereo vision. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 1979. 7.2

[157] 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. 4.2, 4.3.1, 5.3.1, 6.3.1

[158] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. URL `http://lmb.informatik.`

uni-freiburg.de/Publications/2016/MIFDB16. arXiv:1512.02134. 7.3.3, 7.4, 7.4.2, 8.4

[159] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, 2018. 7.3.3, 7.5

[160] 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. 4.2

[161] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *CVPR*, 2015. 1.3.4, 7.3.2, 7.3.3, 7.4, 7.4.1, 8.2, 8.3.1, 8.4, 8.4.2

[162] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018. 7.1, 7.4.1, 7.4.2, 7.3, 7.4

[163] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 5.2

[164] Mariem Mezghanni, Théo Bodrito, Malika Boulkenafed, and Maks Ovsjanikov. Physical simulation layer for accurate 3d modeling. In *CVPR*, pages 13514–13523, 2022. 6.2

[165] 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. 1.3, 3.1, 3.3.2, 3.3.4, 4.1, 4.2, 4.3.1, 4.3.1, 4.3.4, 4.4, 4.4.1, 5.2, 6.3.4

[166] Michael Mistry, Jonas Buchli, and Stefan Schaal. Inverse dynamics control of floating base systems using orthogonal decomposition. In *2010 IEEE International Conference on Robotics and Automation*, pages 3406–3412, 2010. doi: 10.1109/ROBOT.2010.5509646. 6.3.3, 6.3.3

[167] Tomoyuki Mori and Sebastian Scherer. First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles. In *ICRA*, 2013. 7.1, 7.2, 7.4.3, 7.4.5

[168] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 2017. 6.3.2, 6.3.3

[169] Amaury Negre, Christophe Braillon, James L Crowley, and Christian Laugier. Real-time time-to-collision from variation of intrinsic scale. In *Experimental Robotics*, pages 75–84. Springer, 2008. 7.2

[170] Natalia Neverova, David Novotny, Vasil Khalidov, Marc Szafraniec, Patrick Labatut, and Andrea Vedaldi. Continuous surface embeddings. In *NeurIPS*,

2020. 3.2, 3.3.2, 3.1, 3.2, 3.3, 3.4.1, 4.3.1, 4.3.3, 4.3, 4.3.4, 4.3.5, 5.3.1

[171] Natalia Neverova, Artsiom Sanakoyeu, Patrick Labatut, David Novotny, and Andrea Vedaldi. Discovering relationships between object categories via universal canonical maps. In *CVPR*, 2021. 1.3, 4.3.3, 4.3.5, 5.4.1, 6.4.1

[172] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136. IEEE, 2011. 3.1, 3.2

[173] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *CVPR*, pages 11453–11464, 2021. 5.2, 5.3.1, 5.3.3, 6.3.1, 6.4

[174] Atsuhiro Noguchi, Umar Iqbal, Jonathan Tremblay, Tatsuya Harada, and Orazio Gallo. Watch it move: Unsupervised discovery of 3D joints for re-posing of articulated objects. *CVPR*, 2021. 5.3.1

[175] Atsuhiro Noguchi, Xiao Sun, Stephen Lin, and Tatsuya Harada. Neural articulated radiance field. In *ICCV*, 2021. 4.2, 5.2

[176] David Novotny, Nikhila Ravi, Benjamin Graham, Natalia Neverova, and Andrea Vedaldi. C3dpo: Canonical 3d pose networks for non-rigid structure from motion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7688–7697, 2019. 1.2, 2.2

[177] Peter Ochs, Jitendra Malik, and Thomas Brox. Segmentation of moving objects by long term video analysis. *PAMI*, 2013. 8.2

[178] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 1.3

[179] 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. 4.1, 4.2, 4.3.1, 4.4, 4.4.1, 4.4.2, 4.4.3, 5.1, 5.3.2, 5.3.4, 6.1, 6.3.2

[180] 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. 4.2, 5.1, 5.3.1

[181] Sang Il Park and Jessica K Hodgins. Capturing and animating skin deformation in human motion. *ACM Transactions on Graphics (TOG)*, 25(3):881–889, 2006.

1.1

[182] 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. 2.1, 2.2, 2.4, 2.4.2, 4.2, 5.2, 6.2

[183] 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. 4.2

[184] 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. 4.2

[185] Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Lee, Jie Tan, and Sergey Levine. Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*, 2020. 1.1

[186] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 8.2, 8.4.1

[187] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, pages 724–732, 2016. 2.3.4, 2.4.1, 3.4.1

[188] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007. 7.2

[189] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017. 5.4.1

[190] Michael Posa, Cecilia Cantu, and Russ Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2014. 6.3.4

[191] True Price, Johannes L Schönberger, Zhen Wei, Marc Pollefeys, and Jan-Michael Frahm. Augmenting crowd-sourced 3d reconstructions using semantic detections. In *CVPR*, 2018. 7.2

[192] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *CVPR*, 2020. 4.1, 4.2, 4.4.3, 5.1, 6.2

[193] Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C Lin. Efficient differentiable simulation of articulated bodies. In *ICLR*, pages 8661–8671. PMLR, 2021. 6.2

[194] Yi-Ling Qiao, Alexander Gao, and Ming C. Lin. Neuphysics: Editable neural geometry and physics from monocular videos. In *NeurIPS*, 2022. 6.3.4

[195] Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar Zaiane, and Martin Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. *Pattern Recognition*, 2020. 8.4.1, 8.1

[196] Weichao Qiu, Xinggang Wang, Xiang Bai, Alan Yuille, and Zhuowen Tu. Scale-space sift flow. In *WACV*, 2014. 7.2, 7.5

[197] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *CVPR*, pages 4058–4066, 2016. 7.2

[198] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *arXiv:1907.01341*, 2019. 8.1, 8.3.2, 8.4, 8.4.2

[199] Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *CVPR*, 2019. 8.4, 8.4.1, 8.1, 8.4.1, 8.2, 8.4.2

[200] Carolina Raposo and Joao P Barreto. Theory and practice of structure-from-motion using affine correspondences. In *CVPR*, 2016. 7.2

[201] Alex Rav-Acha, Pushmeet Kohli, Carsten Rother, and Andrew Fitzgibbon. Unwrap mosaics: A new representation for video editing. In *SIGGRAPH*, pages 1–11, 2008. 3.2

[202] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 2.4.2

[203] Bernhard Reinert, Tobias Ritschel, and Hans-Peter Seidel. Animated 3d creatures from single-view video by skeletal sketching. In *Proceedings of Graphics Interface 2016*, GI 2016, pages 133–141. Canadian Human-Computer Communications Society / Société canadienne du dialogue humain-machine, 2016. ISBN 978-0-9947868-1-4. doi: 10.20380/GI2016.17. 2.1, 2.2

[204] Davis Rempe, Leonidas J Guibas, Aaron Hertzmann, Bryan Russell, Ruben Villegas, and Jimei Yang. Contact and human dynamics from monocular video. In *ECCV*, pages 71–87. Springer, 2020. 6.1, 6.2, 6.3.4, 6.4.3

[205] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar,

and Leonidas J. Guibas. Humor: 3d human motion model for robust pose estimation. In *ICCV*, 2021. 5.4.1, 6.6, 6.4.1

[206] Zhe Ren, Junchi Yan, Bingbing Ni, Bin Liu, Xiaokang Yang, and Hongyuan Zha. Unsupervised deep learning for optical flow estimation. In *AAAI*, 2017. 7.3.3, 7.5

[207] Zhile Ren, Orazio Gallo, Deqing Sun, Ming-Hsuan Yang, Erik B Sudderth, and Jan Kautz. A fusion approach for multi-frame optical flow estimation. In *WACV*, pages 2077–2086. IEEE, 2019. 3.2

[208] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *ICCV*, pages 2213–2222, 2017. 8.4

[209] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelović, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Neighbourhood consensus networks. In *NeurIPS*, 2018. 2.1

[210] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 3.3.2, 8.3.2

[211] Nadine Rueegg, Silvia Zuffi, Konrad Schindler, and Michael J. Black. BARC: Learning to regress 3D dog shape from images by exploiting breed information. In *CVPR*, pages 3876–3884, 2022. 5.2, 5.4.2

[212] Chris Russell, Rui Yu, and Lourdes Agapito. Video pop-up: Monocular 3d reconstruction of dynamic scenes. In *ECCV*. Springer, 2014. 7.2

[213] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *ICCV*, pages 2304–2314, 2019. 2.2, 3.2, 5.2

[214] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In *CVPR*, 2020. 1.2, 2.1, 2.1, 2.2, 2.4, 3.2, 3.4.1, 5.2

[215] Shunsuke Saito, Jinlong Yang, Qianli Ma, and Michael J. Black. SCANimate: Weakly supervised learning of skinned clothed avatar networks. In *CVPR*, 2021. 4.1, 4.3.2, 5.2

[216] Peter Sand and Seth Teller. Particle video: Long-range motion estimation using point trajectories. In *IJCV*, 2008. 2.1, 2.2, 3.2, 4.2, 6.2

[217] Tomokazu Sato, Tomas Pajdla, and Naokazu Yokoya. Epipolar geometry estimation for wide-baseline omnidirectional street view images. In *ICCVW*, 2011. 7.2

[218] Harpreet S Sawhney. 3d geometry from planar parallax. In *CVPR*, 1994. 8.2, 8.3.1

[219] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1-3):7–42, 2002. 1.3.4

[220] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 7.11

[221] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 1.2, 4.4.4

[222] Johannes Lutz Schönberger, True Price, Torsten Sattler, Jan-Michael Frahm, and Marc Pollefeys. A vote-and-verify strategy for fast spatial verification in image retrieval. In *ACCV*, 2016. 2.4.2

[223] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1.2, 4.4.4

[224] Paul R Schrater, David C Knill, and Eero P Simoncelli. Perceiving visual expansion without optic flow. *Nature*, 410(6830):816, 2001. 7.1

[225] René Schuster, Christian Bailer, Oliver Wasenmüller, and Didier Stricker. Combining stereo disparity and optical flow for basic scene flow. In *Commercial Vehicle Technology 2018*, pages 90–101. Springer, 2018. 7.4.1, 7.1

[226] Yaser Sheikh, Omar Javed, and Takeo Kanade. Background subtraction for freely moving cameras. In *ICCV*, pages 1219–1225. IEEE, 2009. 5.3.3

[227] Jianbo Shi and Jitendra Malik. Motion segmentation and tracking using normalized cuts. In *ICCV*, 1998. 8.1

[228] Soshi Shimada, Vladislav Golyanik, Weipeng Xu, and Christian Theobalt. Physcap: Physically plausible monocular 3d motion capture in real time. *ACM Transactions on Graphics (ToG)*, 39(6):1–16, 2020. 6.1, 6.2, 6.3.4, 6.4.3

[229] Soshi Shimada, Vladislav Golyanik, Weipeng Xu, Patrick Pérez, and Christian Theobalt. Neural monocular 3d human motion capture with physical awareness. *ToG*, 40(4), aug 2021. 6.2

[230] 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. 2.1, 2.1, 2.2, 2.4.1, 2.3, 3.1, 3.2, 3.3, 3.4.2, 4.2, 6.2

[231] 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. 4.3.5

[232] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *SIGGRAPH*. 2006. 4.1

[233] Noah Snavely, Steven M Seitz, and Richard Szeliski. Modeling the world from

internet photo collections. *IJCV*, 2008. 1.2, 3.1, 4.1

[234] Chaoyue Song, Tianyi Chen, Yiwen Chen, Jiacheng Wei, Chuan Sheng Foo, Fayao Liu, and Guosheng Lin. Moda: Modeling deformable 3d objects from casual videos, 2023. 5.3.2

[235] Chonghyuk Song, Gengshan Yang, Kangle Deng, Jun-Yan Zhu, and Deva Ramanan. Total-recon: Deformable scene reconstruction for embodied view synthesis. *arXiv preprint arXiv:2304.12317*, 2023. 1.3.3, 4

[236] Olga Sorkine-Hornung and Michael Rabinovich. Least-squares rigid motion using svd. *Computing*, 1(1), 2017. 7.4.6

[237] David E Stewart and J C Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering*, 39(15):2673–2691, 1996. 6.2

[238] 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. 4.2, 5.2

[239] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew Davison. iMAP: Implicit mapping and positioning in real-time. In *ICCV*, 2021. 4.3.4

[240] Robert W Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. In *ACM SIGGRAPH 2007 papers*. 2007. 2.3.2, 2.3.3, 6.3.2

[241] Deqing Sun, Stefan Roth, and Michael J Black. Secrets of optical flow estimation and their principles. In *CVPR*, 2010. 8.3.1, 8.3.2

[242] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, June 2018. 1.3, 3.2

[243] Narayanan Sundaram, Thomas Brox, and Kurt Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *ECCV*, 2010. 2.1, 2.2, 3.2, 4.2, 6.2

[244] Michael T Swanston and Walter C Gogel. Perceived size and motion in depth from optical expansion. *Perception & psychophysics*, 39(5):309–326, 1986. 7.1

[245] Richard Szeliski and Sing Bing Kang. Shape ambiguities in structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19 (5):506–512, 1997. 1

[246] Jeff Tan, Gengshan Yang, and Deva Ramanan. Distilling neural fields for real-time articulated shape reconstruction. In *CVPR*, pages 4692–4701, 2023. 1.3.2, 5

[247] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil,

Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020. 4.3.2

[248] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *CVPR*, pages 3405–3414, 2019. 4.4.1, 5.4.1, 6.4.1

[249] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 1.3, 2.1, 2.3.4, 3.2, 6.3.1

[250] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in Neural Information Processing Systems*, 34, 2021. 4.4.4, 6.3.1

[251] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. Learning motion patterns in videos. In *CVPR*, 2017. 8.2

[252] Pavel Tokmakov, Cordelia Schmid, and Karteek Alahari. Learning to segment moving objects. *IJCV*, 2019. 8.2

[253] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *IJCV*, 9(2):137–154, 1992. 1.2, 3.1

[254] Philip HS Torr. Geometric motion segmentation and model selection. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 356(1740):1321–1340, 1998. 8.2, 8.3.1, 8.3.1

[255] Philip HS Torr, Andrew Zisserman, and Stephen J Maybank. Robust detection of degenerate configurations while estimating the fundamental matrix. *CVIU*, 1998. 8.3.1

[256] Philip HS Torr, Andrew W Fitzgibbon, and Andrew Zisserman. The problem of degeneracy in structure and motion recovery from uncalibrated image sequences. *IJCV*, 1999. 8.2

[257] 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. 4.2

[258] Roberto Tron and René Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *CVPR*, 2007. 8.1, 8.2

[259] Shubham Tulsiani, Nilesh Kulkarni, and Abhinav Gupta. Implicit mesh reconstruction from unannotated image collections. In *arXiv*, 2020. 2.1, 2.2, 2.3.3, 3.2, 3.3.3, 4.2, 5.2

[260] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *3DV*, 2017. 7.4.4

[261] Shimon Ullman. The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 1979. 7.2

[262] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *ICCV*, 1999. 8.2

[263] René Vidal and Shankar Sastry. Optimal segmentation of dynamic scenes from two perspective views. In *CVPR*, 2003. 8.2

[264] René Vidal, Yi Ma, Stefano Soatto, and Shankar Sastry. Two-view multibody structure from motion. *IJCV*, 2006. 7.2, 8.2

[265] Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. Articulated mesh animation from multi-view silhouettes. *TOG*, 2008. 4.4.1

[266] Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. Articulated mesh animation from multi-view silhouettes. In *SIGGRAPH 2008*. 2008. 2.4.2, 5.4.1

[267] Minh Vo, Yaser Sheikh, and Srinivasa G Narasimhan. Spatiotemporal bundle adjustment for dynamic 3d human reconstruction in the wild. *IEEE TPAMI*, 2020. 4.2, 5.2, 6.2

[268] Christoph Vogel, Konrad Schindler, and Stefan Roth. 3D scene flow estimation with a piecewise rigid scene model. *IJCV*, 2015. 7.1, 7.4.1, 7.2, 7.4.2, 7.3, 7.4, 8.2, 8.3, 8.4.3

[269] Chaoyang Wang, José Miguel Buenaposada, Rui Zhu, and Simon Lucey. Learning depth from monocular videos using direct methods. In *CVPR*, pages 2022–2030, 2018. 8.4.2

[270] Chaoyang Wang, Chen-Hsuan Lin, and Simon Lucey. Deep nrsfm++: Towards 3d reconstruction in the wild. *arXiv preprint arXiv:2001.10090*, 2020. 2.2

[271] Chaoyang Wang, Ben Eckart, Simon Lucey, and Orazio Gallo. Neural trajectory fields for dynamic novel view synthesis. *arXiv preprint arXiv:2105.05994*, 2021. 4.2

[272] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638, September 1994. 3.2

[273] Jiepeng Wang, Peng Wang, Xiaoxiao Long, Christian Theobalt, Taku Komura, Lingjie Liu, and Wenping Wang. Neuris: Neural reconstruction of indoor scenes using normal priors. In *ECCV*, pages 139–155. Springer, 2022. 6.3.1

[274] 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. 4.3.1

[275] Shenlong Wang, Linjie Luo, Ning Zhang, and Jia Li. Autoscaler: scale-attention networks for visual correspondence. *BMVC*, 2016. 7.2, 7.5

[276] 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. 4.2, 5.1

[277] William H Warren, Michael W Morris, and Michael Kalish. Perception of translational heading from optical flow. *Journal of Experimental Psychology: Human Perception and Performance*, 14(4):646, 1988. 7.2, 7.3.3

[278] Joseph Weber and Jitendra Malik. Rigid body segmentation and shape description from dense optical flow under weak perspective. *TPAMI*, 19(2):139–143, 1997. 8.1

[279] Chung-Yi Weng, Brian Curless, Pratul P Srinivasan, Jonathan T Barron, and Ira Kemelmacher-Shlizerman. Humannerf: Free-viewpoint rendering of moving people from monocular video. In *CVPR*, pages 16210–16220, 2022. 6.3.2

[280] Chung-Yi Weng, Brian Curless, Pratul P. Srinivasan, Jonathan T. Barron, and Ira Kemelmacher-Shlizerman. HumanNeRF: Free-viewpoint rendering of moving people from monocular video. In *CVPR*, pages 16210–16220, June 2022. 5.3.1, 5.3.2

[281] Keenon Werling, Dalton Omens, Jeongseok Lee, Ioannis Exarchos, and C Karen Liu. Fast and feature-complete differentiable physics for articulated rigid bodies with contact. *RSS*, 2021. 6.2

[282] Shangzhe Wu, Tomas Jakab, Christian Rupprecht, and Andrea Vedaldi. Dove: Learning deformable 3d objects by watching videos. *arXiv preprint arXiv:2107.10844*, 2021. 3.2, 4.2, 5.2, 6.2

[283] Shangzhe Wu, Ruining Li, Tomas Jakab, Christian Rupprecht, and Andrea Vedaldi. Magicpony: Learning articulated 3d animals in the wild. *arXiv preprint arXiv:2211.12497*, 2022. 5.2

[284] Yuefan Wu, Zeyuan Chen, Shaowei Liu Liu, Zhongzheng Ren, and Shenlong Wang. Casa: Category-agnostic skeletal animal reconstruction. In *NeurIPS*, 2022. 5.3.1

[285] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019. 8.4.1, 8.1

[286] Jonas Wulff, Laura Sevilla-Lara, and Michael J. Black. Optical flow in mostly rigid scenes. In *CVPR*, 2017. 8.2, 8.4, 8.4, 8.4.1, 8.1, 8.4

[287] Donglai Xiang, Hanbyul Joo, and Yaser Sheikh. Monocular total capture: Posing face, body, and hands in the wild. In *CVPR*, 2019. 1.1, 2.2, 4.2, 5.2, 6.2

[288] Christopher Xie, Keunhong Park, Ricardo Martin-Brualla, and Matthew Brown. Fig-nerf: Figure-ground neural radiance fields for 3d object category modelling.

In *3DV*, pages 962–971. IEEE, 2021. 5.3.1

[289] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12193–12202, 2020. 8.3.2, 8.3.2

[290] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 8.4.1

[291] Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J. Black. ICON: Implicit Clothed humans Obtained from Normals. In *CVPR*, pages 13296–13306, June 2022. 5.2, 5.4.1

[292] Hongyi Xu, Eduard Gabriel Bazavan, Andrei Zanfir, William T Freeman, Rahul Sukthankar, and Cristian Sminchisescu. Ghum & ghuml: Generative 3d human shape and articulated pose models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6184–6193, 2020. 6.2

[293] Jie Xu, Tao Chen, Lara Zlokapa, Michael Foshey, Wojciech Matusik, Shinjiro Sueda, and Pulkit Agrawal. An end-to-end differentiable framework for contact-aware robot design. *RSS*, 2021. 6.3.3

[294] Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, and Miles Macklin. Accelerated policy learning with parallel differentiable simulation. *ICLR*, 2022. 6.3.3

[295] Li Xu, Zhenlong Dai, and Jiaya Jia. Scale invariant optical flow. In *ECCV*, 2012. 7.2, 7.5

[296] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. YouTube-VOS: Sequence-to-sequence video object segmentation. In *ECCV*, pages 585–601, 2018. 3.4.3

[297] Weipeng Xu, Avishek Chatterjee, Michael Zollhöfer, Helge Rhodin, Dushyant Mehta, Hans-Peter Seidel, and Christian Theobalt. Monoperfcap: Human performance capture from monocular video. *ACM Trans. Graph.*, 37(2):27:1–27:15, May 2018. ISSN 0730-0301. doi: 10.1145/3181973. URL http://doi.acm.org/10.1145/3181973. 5.4.1

[298] Xun Xu, Loong Fah Cheong, and Zhuwen Li. 3d rigid motion segmentation with mixed and unknown number of models. *PAMI*, 2019. 8.1, 8.2

[299] Yuanlu Xu, Song-Chun Zhu, and Tony Tung. Denserac: Joint 3d pose and shape estimation by dense render-and-compare. In *ICCV*, 2019. 4.3.4

[300] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh.

Rignet: Neural rigging for articulated characters. *ACM Trans. on Graphics*, 39, 2020. 5.3.1

[301] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. In *NeurIPS*, 2019. 1.3, 1.3.1, 1.3.4, 3, 2.1, 2.3.4, 2.3, 3.2, 3.3.2, 3.2, 3.3, 4.4.1, 5.4.1, 6.3.1, 7.4, 8.3.2

[302] Gengshan Yang and Deva Ramanan. Upgrading optical flow to 3d scene flow through optical expansion. In *CVPR*, 2020. 1.3.4, 8.1, 8.3.1, 8.2, 8.3.2, 8.2, 8.4.2, 8.3, 8.4.3

[303] Gengshan Yang and Deva Ramanan. Learning to segment rigid motions from two frames. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1266–1275, 2021. 1.3.4

[304] Gengshan Yang, Peiyun Hu, and Deva Ramanan. Inferring distributions over depth from a single image. In *IROS*, 2019. 1.3.4, 1

[305] Gengshan Yang, Joshua Manela, Michael Happold, and Deva Ramanan. Hierarchical deep stereo matching on high-resolution images. In *CVPR*, 2019. 1.3.4, 2, 7.5

[306] 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. 1.3.1, 1.3.2, 3.2, 3.3, 3.3.1, 3.3.3, 3.1, 3.2, 3.3, 3.4.1, 4.1, 4.1, 4.2, 4.3.2, 5.2, 6.2, 6.3.2

[307] 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. 1.3.1, 4.1, 4.1, 4.2, 4.3.3, 4.3.4, 4.4, 4.4.1, 4.4.2, 5.2, 5.3.1, 6.2

[308] Gengshan Yang, Minh Vo, Natalia Neverova, Deva Ramanan, Andrea Vedaldi, and Hanbyul Joo. BANMo: Building Animatable 3D Neural Models from Many Casual Videos. *CVPR*, 2022. 1.3.1, 1.3.2, 5.1, 5.2, 5.3.2, 5.3.4, 5.4, 5.4.1, 6.2, 6.3.1, 6.3.1, 6.3.2, 6.3.2, 6.3.3, 6.3.4, 6.3.4, 6.4, 6.4.1

[309] Gengshan Yang, Chaoyang Wang, N. Dinesh Reddy, and Deva Ramanan. Reconstructing animatable categories from videos. In *CVPR*, 2023. 1.3.2

[310] Gengshan Yang, Shuo Yang, Ziyang Zhang, and Deva Ramanan. PPR: Physically plausible reconstruction from monocular videos. In *Under Review*, 2023. 1.3.3, 5

[311] Yi Yang and Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, pages 1385–1392. IEEE, 2011. 2.4.1

[312] Zhao Yang, Qiang Wang, Luca Bertinetto, Weiming Hu, Song Bai, and Philip HS

Torr. Anchor diffusion for unsupervised video object segmentation. In *CVPR*, 2019. 8.2

[313] 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. 4.3.4, 4.4, 6.3.4

[314] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *arXiv preprint arXiv:2106.12052*, 2021. 4.3.1, 6.3.1, 6.4

[315] Vickie Ye, Georgios Pavlakos, Jitendra Malik, and Angjoo Kanazawa. Decoupling human and camera motion from videos in the wild. In *CVPR*, June 2023. 1.3.3

[316] Yufei Ye, Shubham Tulsiani, and Abhinav Gupta. Shelf-supervised mesh prediction in the wild. In *CVPR*, 2021. 3.2, 4.2, 5.2, 6.2

[317] Jae Shin Yoon, Zhixuan Yu, Jaesik Park, and Hyun Soo Park. Humbi: A large multiview dataset of human body expressions and benchmark challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):623–640, 2021. 1.1

[318] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *NeurIPS*, 2022. 6.3.1, 6.4

[319] Chang Yuan, Gerard Medioni, Jinman Kang, and Isaac Cohen. Detecting motion regions in the presence of a strong parallax from a moving camera by multiview geometric constraints. In *TPAMI*, 2007. 6.3.3, 8.1, 8.1, 8.2, 8.3.1

[320] Bernhard Zeisl, Torsten Sattler, and Marc Pollefeys. Camera pose voting for large-scale image-based localization. In *ICCV*, 2015. 7.2

[321] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. GA-Net: Guided aggregation net for end-to-end stereo matching. In *CVPR*, 2019. 7.4.1, 8.4.3

[322] 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. 4.3.4

[323] Jason Y. Zhang, Deva Ramanan, and Shubham Tulsiani. RelPose: Predicting probabilistic relative rotation for single objects in the wild. In *ECCV*, 2022. 1

[324] John Z Zhang, Shuo Yang, Gengshan Yang, Arun L Bishop, Deva Ramanan, and Zachary Manchester. Slomo: A general system for legged robot motion imitation from casual videos. *arXiv preprint arXiv:2304.14389*, 2023. 6

[325] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang.

The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018. 2.3.3, 3.3.3

[326] Zerong Zheng, Tao Yu, Yebin Liu, and Qionghai Dai. Pamir: Parametric model-conditioned implicit representation for image-based human reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. doi: 10.1109/TPAMI.2021.3050505. 5.2

[327] Tiancheng Zhi, Christoph Lassner, Tony Tung, Carsten Stoll, Srinivasa G Narasimhan, and Minh Vo. Texmesh: Reconstructing detailed human texture and geometry from rgb-d video. In *ECCV*, 2020. 4.2

[328] Yaofeng Desmond Zhong, Jiequn Han, and Georgia Olympia Brikis. Differentiable physics simulations with contacts: Do they have correct gradients wrt position, velocity and control? In *ICML 2022 2nd AI for Science Workshop*, 2022. 6.2

[329] Tianfei Zhou, Shunzhou Wang, Yi Zhou, Yazhou Yao, Jianwu Li, and Ling Shao. Motion-attentive transition for zero-shot video object segmentation. In *AAAI*, pages 13066–13073, 2020. 2.3.4, 8.2, 8.4.1, 8.1

[330] Tinghui Zhou, Philipp Krahenbuhl, Mathieu Aubry, Qixing Huang, and Alexei A Efros. Learning dense correspondence via 3d-guided cycle consistency. In *CVPR*, 2016. 3.4.1

[331] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. In *arXiv preprint arXiv:1904.07850*, 2019. 8.3.2, 8.3.2

[332] Yingying Zhu, Dong Huang, Fernando De La Torre, and Simon Lucey. Complex non-rigid motion 3d reconstruction by union of subspaces. In *CVPR*, 2014. 7.1, 7.2

[333] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *CVPR*, 2017. 2.1, 2.1, 2.2, 2.4.1, 3.2, 4.2, 5.2, 5.3.1, 6.2

[334] Silvia Zuffi, Angjoo Kanazawa, and Michael J. Black. Lions and tigers and bears: Capturing non-rigid, 3D, articulated shape from images. In *CVPR*, 2018. 1.2, 2.1, 2.2, 3.2, 4.2, 5.2, 6.2

[335] 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. 2.1, 2.2, 2.4.1, 2.3, 4.2, 5.2, 6.2