# Towards Reconstructing Non-rigidity from Single Camera

Chaoyang Wang

CMU-RI-TR-23-25

May 10, 2023


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

**Thesis Committee:**
Simon Lucey *chair*
Laszlo A. Jeni
Fernando De La Torre
Katerina Fragkiadaki
Hongdong Li , *Australian National University*

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

# Abstract

In this document, we study how to infer 3D from images captured by a single camera, without assuming the target scenes / objects being static. The non-static setting makes our problem ill-posed and challenging to solve, but is vital in practical applications where target-of-interest is non-static. To solve ill-posed problems, the current trend in the field is to learn inference models *e.g*. neural networks on datasets with labeled groundtruth. Instead, we attempt a data-less approach without requiring datasets with 3D annotations. This poor man's approach is beneficial to tasks which lack well annotated datasets.

Our works are grouped into two parts.

(i) We first introduce our series of works on non-rigid structure from motion (NR-SfM) and its application to learn 3D landmark detectors with only 2D landmark annotations. Our general framework is a two stage approach – we design a novel NR-SfM module to reconstruct shape and camera poses from input 2D landmarks, and then these are used to teach a neural network to detect 3D landmarks from image inputs. We propose techniques to make the NR-SfM module scalable to large datasets and robust to missing data. We also propose a new loss to let the 3D landmark detector learn more efficiently from the NR-SfM module.

(ii) We then present works on reconstructing dense dynamic scenes. Dense reconstruction is challenging for NR-SfM algorithms mainly due to the difficulty in getting reliable long-term correspondences for every pixel. On the other hand, being able to reconstruct every pixel of the scene is necessary for applications like novel view synthesis. Therefore, we investigate solutions without the need for long-term correspondences. As a preliminary exploration, we first take a data-driven approach, by collecting videos from Internet and train a depth estimation network. Despite the simplicity of this approach, it lacks geometric reasoning and consequently limited in its generalizability. We then explore an analysis-by-synthesis approach, where we leverage recent advances in differentiable neural rendering and represent dynamic scenes using deformable neural radiance fields (D-NeRF). Prior D-NeRF-based methods only use photometric loss for optimization, which we find is not sufficient to recover rapid object motions. We present a new method for D-NeRFs that can directly use optical flow as supervision. We overcome the major challenge with respect to the computational inefficiency of enforcing the flow constraints

to the deformation field used by D-NeRFs. We present results on novel view synthesis with rapid object motion, and demonstrate significant improvements over baselines without flow supervision.

# Acknowledgments

I would like to express my deepest gratitude to my supervisor, Dr. Simon Lucey, for his invaluable guidance and unwavering support throughout my research. His vast expertise and insightful feedback have been instrumental in shaping this thesis and helping me develop as a researcher.

I would also like to express the same level of gratitude to my co-advisor, Dr. Laszlo A. Jeni, for his invaluable support during the last two years of my PhD life. His expertise, constructive criticism, and helpful advice have been invaluable, and I am grateful for his assistance in finding job opportunities after graduation.

I would like to thank all the members of my thesis committee, Fernando De La Torre, Katerina Fragkiadaki and Hongdong Li, for their constructive comments and valuable suggestions, which have significantly contributed to the quality of this work.

I would like to extend my thanks to all the mentors and collaborators throughout my whole research journey. This includes Lijuan Wang, Yasuyuki Matsushita, Frank K. Soong, and Yichen Wei from Microsoft Research Asia, who were instrumental in helping me complete my first computer vision project and my first CVPR paper. I would like to thank Prof. Liqing Zhang for his great support and supervision during my study in Shanghai Jiaotong University. I would like to thank Oliver Wang and Federico Perazzi from Adobe for their hospitality during my internship and for contributing to my wonderful experience in Seattle. I would also like to thank Orazio Gallo and Ben Eckart from Nvidia for the memorable experience we had while fighting for the paper submission deadline.

I am grateful to all the current and past members of CI2CV Lab and CUBE Lab as well as my co-authors for their stimulating discussions, encouragement, and support. I would like to thank Chen-Hsuan Lin, Ming-Fang Chang, Gengshan Yang, Ziyan Wang, Rui Zhu, Nathaniel Chodosh, Xueqian Li, Hamed Kiani, Jhony Kaesemodel Pontes, Chen Kong, Mosamkumar Dabhi, Calvin Murdock, Lachlan MacDonald, Sameera Ramasinghe, Jose Miguel Buenaposada, Ashton Fagg, Shubham Agrawal, Anuj Pahuja, Kushal Vyas, Brendan Miller, Hunter Goforth, Zoltan A. Milacski, Heng Yu, Aarush Gupta, Dinesh Reddy. Their willingness to share their expertise and ideas has been an inspiration to me.

Finally, I would like to thank my parents as much as I can for their unconditional

love and support, which have sustained me throughout this challenging journey. Their belief in me has been a constant source of motivation and inspiration.

# Contents

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

# List of Figures

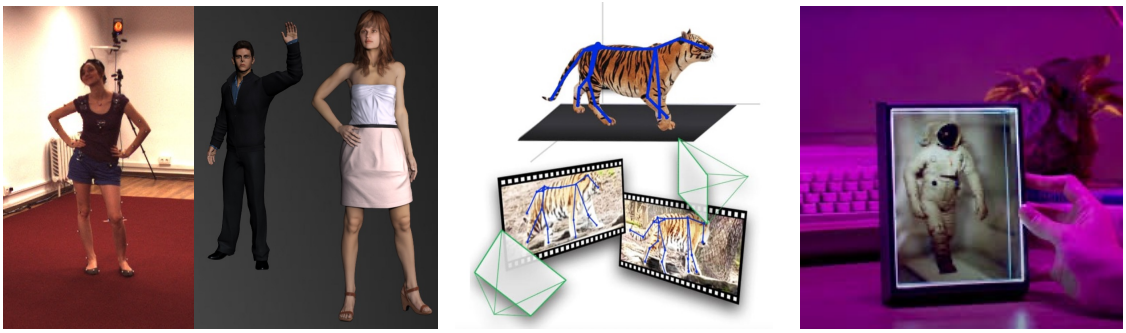
# List of Tables

# Chapter 1

# Introduction

Images and videos are ubiquitous media for capturing the dynamic world around us. Extracting 3D information from these 2D media has led to numerous interesting applications in fields such as robotics, computer graphics, virtual reality, and augmented reality. For instance, in robotics, 3D reconstruction based on multiple images or videos can provide valuable depth information, enabling robots to navigate environments more effectively and avoid obstacles. In computer graphics, 3D reconstruction can be used to create visually stunning 3D models of objects and scenes for use in video games, movies, and other visual media. Moreover, in virtual reality, 3D reconstruction can be utilized to create immersive environments that can be explored in a virtual space, while in augmented reality, it can be used to superimpose digital objects onto the real world. Capturing human motions from 2D media can be utilized to control virtual characters in real-time, opening up new possibilities for gaming, animation, and other interactive media. Furthermore, 3D reconstruction can be used to understand animal behaviors and to reconstruct light fields, allowing the display of 4D content on holographic displays.

To recover accurate 3D from imaginary data recorded by a single camera, is however challenging. Geometric-based vision methods are typically only reliable for static scenes, as the equations derived from trigonometry are severely underconstrained for dynamic scenes. To address this issue, researchers have turned to machine learning models that can learn priors from data. One of the most successful applications of such models is in human 3D body pose estimation, which has been made possible by the construction of large-scale motion capture datasets such as PanopticStudio [86] and realistic synthetic



(a) motion capture for virtual character control (b) animal behavior analysis (c) light field display

Figure 1.1: Extracting 3D information of dynamic scenes and objects yields interesting applications, such as (a) motion capture for virtual character control [83], (b) animal behavior study [37], (c) reconstructing light field for holographic display [2].

datasets like *e.g*. RenderPeople [3]. Constructing those datasets, however can be costly. For example, PanopticStudio is captured by hundreds of calibrated cameras and is a joint effort by tens of researchers and annotators. Whether this kind of success can be easily replicated to other tasks is questionable, at least not with a limited budget. In this document, we attempt another direction. We investigate how to infer 3D from 2D observations without the need for 3D annotations, using an unsupervised learning paradigm. This approach has the potential to expand our training source to include images and videos collected from the internet. But IT also presents theoretical challenges. Nonetheless, we will present our positive findings on overcoming such difficulties, within the context of 3D pose estimation and dynamic scene reconstruction from videos.

This document is divided into two parts, based on the scope of the problems addressed.

## 1.1 Learning 3D pose estimation from 2D annotations

The first part of this document focuses on describing a pipeline for learning 3D pose estimation from 2D annotations in still images. The motivation for this task is that datasets with solely 2D landmark annotations are abundant and easier to acquire. By using these annotations, our method can be applied to a wider range of objects, without being limited by the availability of 3D models, kinematic priors, or sequential/multi-view footage.

The pipeline described in this document consists of two modules, as illustrated in

Fig. 1.2. The first module is a non-rigid structure from motion (NR-SfM) method, which reconstructs camera poses and shape representations from 2D landmark annotations in the training set. This module serves as a teacher to train our second module, which is a 3D landmark detector that directly regresses 3D landmark positions from image inputs. In the following sections, we will provide detailed explanations of each of these modules.

In the first module of our pipeline, we introduce a conceptually simple non-rigid structure from motion (NR-SfM) method based on autoencoders, which we call the Procustean Autoencoder for Unsupervised Lifting (PAUL). This method is unique in two ways: (i) it learns the 3D autoencoder weights solely from 2D projected measurements, and (ii) it is Procrustean in that it jointly resolves the unknown rigid pose for each shape instance. We demonstrate state-of-the-art performance across several benchmarks and compare favorably to recent innovations such as Deep NRSfM [94] and C3PDO [143].

One practical issue for many deep NR-SfM methods is how to deal with missing data and perspective projection. Addressing these issue is crucial for learning from in-the-wild datasets. We mathematically derive a generalized strategy for improving a learning-based NRSfM methods [94] to tackle the above issues.

Finally, we propose to learn a 3D landmark detector by distilling knowledge from NR-SfM. The challenge for using NR-SfM as teacher is that they often make poor depth reconstruction when the 2D projections have strong ambiguity. Directly using those wrong depth as hard target would negatively impact the student. Instead, we propose a novel loss that ties depth prediction to the cost function used in NR-SfM. This gives the student pose estimator freedom to reduce depth error by associating with image features. Validated on H3.6M dataset, our learned 3D pose estimation network achieves more accurate reconstruction compared to NRSfM methods.

## 1.2 Dense reconstruction of dynamic scenes

### 1.2.1 Learning depth estimation from videos

We first present a fully data-driven method to compute depth from diverse monocular video sequences that contain large amounts of non-rigid objects, e.g., people. In order to learn reconstruction cues for non-rigid scenes, we introduce a new dataset consisting of stereo videos scraped in-the-wild. This dataset has a wide variety of scene types, and features

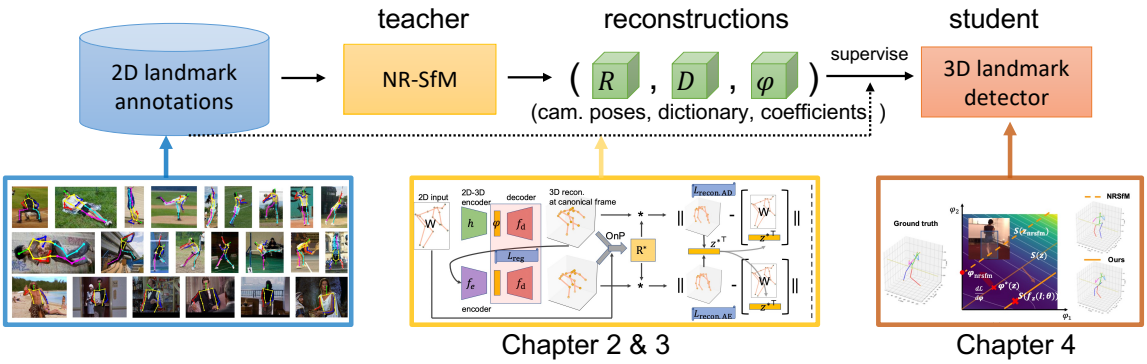



Figure 1.2: Overview of Part I: Pipeline for learning a 3D pose estimator from 2D landmark annotations. We first employ NR-SfM (as a teacher) to reconstruct 3D from 2D landmark annotations. The 3D reconstruction is represented as a combination of camera poses, 3D shape dictionary, and shape coefficients. These are used as supervisions to teach the student 3D landmark detector.

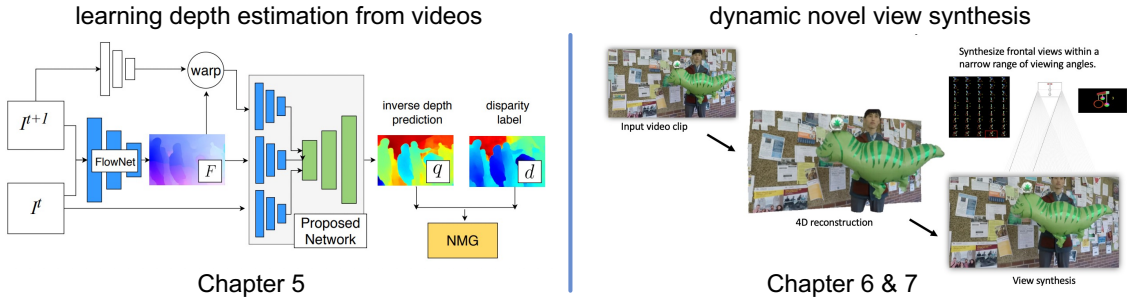



Figure 1.3: Overview of Part II: Dense reconstruction for dyanamic scenes.

large amounts of nonrigid objects, especially people. From this, we compute disparity maps to be used as supervision to train our approach. We propose a loss function that allows us to generate a depth prediction even with unknown camera intrinsics and stereo baselines in the dataset. We validate the use of large amounts of Internet video by evaluating our method on existing video datasets with depth supervision, including SINTEL, and KITTI, and show that our approach generalizes better to natural scenes.

### 1.2.2 Dynamic novel view synthesis using neural radiance fields

We then present our works on novel view synthesis for dynmaic scenes. Recent approaches to render photorealistic views from a limited set of photographs have pushed the boundaries

of our interactions with pictures of static scenes. The ability to recreate moments, that is, time-varying sequences, is perhaps an even more interesting scenario, but it remains largely unsolved.

As a preliminary investigation, we build our method based on time-modulated neural radiance fields to represent dynamic scenes. In order to enforce temporal consistency between frames, we propose neural trajectory fields (NTF), a coordinate-based neural representation for scene motions. NTF learns smooth and stable trajectories over the input sequence for each point in space, which allows us to enforce consistency between any two frames more efficiently compared to other representations such as scene flow fields [116]. This helps improve reconstruction quality, particularly in dynamic regions. We also apply NTF for lidar scene flow integration, which shows more consistent motion estimation compared to scene flow fields.

While using time-modulated NeRF has the drawback of not strictly enforcing temporal consistency. Consistency is enforced as an optimization objective, rather than strictly enforced as a quality constraint. In this regard, deformable NeRF [149] offers an advantage over time-modulated NeRF in terms of ensuring consistency since a single static canonical NeRF is used. However, we found that prior deformable NeRF methods do not perform well in the presence of large scene motions. To address this limitation, we introduce a novel method for training deformable NeRF that utilizes optical flow as an additional supervisory signal, in addition to the photometric loss. By leveraging the motion information from optical flow, our method can better handle large scene motions and improve the overall reconstruction quality. The key technical innovation is that we overcome the major challenge with respect to the computationally inefficiency of enforcing the flow constraints to the backward deformation field, used by deformable NeRFs. Specifically, we show that inverting the backward deformation function is actually not needed for computing scene flows between frames. This insight dramatically simplifies the problem, as one is no longer constrained to deformation functions that can be analytically inverted. Instead, thanks to the weak assumptions required by our derivation based on the inverse function theorem, our approach can be extended to a broad class of commonly used backward deformation field.

Finally, we note that the capability of our current implementation of deformable NeRF is still limited, as it only relies on motion parallax to disambiguite between motion and shape variations. In the last chapter, we provide an overview of additional supervision we can utilize to alleviate some of the ambiguities which motion parallax is not able to handle.

And we will conclude with a vision of future research in reconstructing dynamic scenes.

The following is the relevant publication list for each chapter.

- Chapter 2 – Wang *et al*., "PAUL: Procrustean Autoencoder for Unsupervised Lifting", CVPR 2021 [212].
- Chapter 3 – Wang *et al*., "Deep nrsfm++: Towards unsupervised 2d-3d lifting in the wild", 3DV 2020 [211].
- Chapter 4 Wang *et al*., "Distill Knowledge from NRSfM for Weakly Supervised 3D Pose Learning", ICCV 2019 [209].
- Chapter 5 – Wang *et al*., "Web stereo video supervision for depth prediction from dynamic scenes", 3DV 2019 [213].
- Chapter 6 – Wang *et al*., "Neural trajectory fields for dynamic novel view synthesis", arxived 2021 [207].
  – Wang *et al*., "Neural prior for trajectory estimation", CVPR 2022 [210].
- Chapter 7 – Wang *et al*., "Flow Supervision for Deformable NeRF", CVPR 2023 [214].

# Part I

# Learning 3D pose estimation from 2D annotations

# Chapter 2

# Procrustean Autoencoder for Atemporal Non-rigid Structure from Motion

## 2.1 Introduction

Inferring non-rigid 3D structure from multiple unsynchronized 2D imaged observations is an ill-posed problem. Non-Rigid Structure from Motion (NRS*f*M) methods approach the problem by introducing additional priors – of particular note in this regards are low rank [8, 19, 39] and union of subspaces [103, 251] methods.

Recently, NRSfM has seen improvement in performance by recasting the problem as an unsupervised deep learning problem [23, 143, 151]. These 2D-3D *lifting networks* have inherent advantages over classical NRS*f*M as: (i) they are more easily scalable to larger datasets, and (ii) they allow fast feed-forward prediction once trained. These improvement, however, can largely be attributed to the end-to-end reframing of the learning problem rather than any fundamental shift in the prior/constraints being enforced within the NRSfM solution. For example, both Cha *et al*. [23] and Park *et al*. [151] impose a classical low rank constraint on the recovered 3D shape. It is also well understood [39, 94, 103, 251] that such low rank priors have poor performance when applied to more complex 3D shape variations.

The NRSfM field has started to explore new non-rigid shape priors inspired by recent advances in deep learning. Kong & Lucey [94] proposed the use of hierarchical sparsity to

have a more expressive shape model while ensuring the inversion problem remains well conditioned. Although achieving significant progress in several benchmarks, the approach is limited by the somewhat adhoc approximations it employs so as to make the entire NRSfM solution realizable as a feed-forward lifting network. Such approximations hamper the interpretability of the method as the final network is a substantial departure from the actually proposed objective. We further argue that this departure from the true objective also comes at the cost of the overall effectiveness of the 2D-3D lifting solution.

In this paper we propose a prior that 3D shapes aligned to a common reference frame are compressible with an undercomplete auto-encoder. This is advantageous over previous linear methods, because the deeper auto-encoder is naturally capable of compressing more complicated non-rigid 3D shapes. What makes learning such an auto-encoder challenging is: (i) it observes only 2D projected measurements of the non-rigid shape; (ii) it must automatically resolve the unknown rigid transformation to align each projected shape instance. We refer to our solution as a *Procustean Autoencoder for Unsupervised Lifting* (PAUL). PAUL is considered unsupervised as it has to handle unknown depth, shape pose, and occlusions. Unlike Deep NRSfM [94], the optimization process of PAUL does not have to be realizable as a feed-forward network – allowing for a solution that stays tightly coupled to the proposed mathematical objective.

We also explore other alternative deep shape priors such as: decoder only and decoder + low-rank. A somewhat similar approach is recently explored by Sidhu *et al*. [177] for dense NRS*f*M. Our empirical results demonstrate the fundamental importance of the auto-encoder architecture for 2D-3D lifting.

**Contributions:** We make the following contributions:

- We present an optimization objective for joint learning the 2D-3D lifting network and the Procrustean auto-encoder solely from 2D projected measurements.
- A naive implementation of PAUL through gradient descent would result in poor local minima, so instead we advocate for a bilevel optimization, whose lower level problem can be efficiently solved by orthographic-N-point (OnP) algorithms.
- Our method achieves state-of-the-art performance across multiple benchmarks, and is empirically shown to be robust against the choice of hyper-parameters such as the dimension of the latent code space.

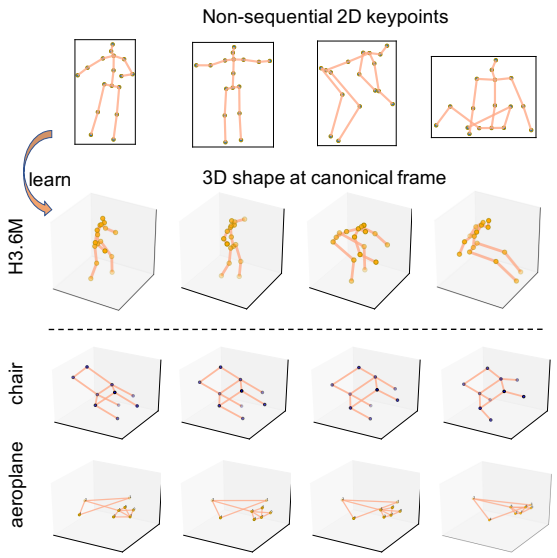



Figure 2.1: PAUL learns to reconstruct 3D shapes aligned to a canonical frame, using 2D keypoint annotations only. Bottom rows show 3D shapes interpolated from the learned latent space of the Procrustean auto-encoder.

## 2.2 Related Work

**Non-rigid structure from motion.** NRS*f*M concerns the problem of recovering 3D shapes from 2D point correspondences from multiple images, *without* the assumption of the 3D shape being rigid. It is ill-posed by nature, and additional priors are necessary to guarantee the uniqueness of the solution. We focus our discussion on the type of priors imposed on shape/trajectory:

(i) *low-rank* was advocated by Bregler *et al*. [19] based on the insight that rigid 3D structure has a fixed rank of three [192]. Dai *et al*. [39] proved that the low-lank assumption is standalone sufficient to solve NRS*f*M. It is also applied temporally [8, 50] to constraint 3D trajectories. Kumar [101] recently revisited Dai's approach [39] and showed that by properly utilizing the assumptions that deformation is smooth over frames, it is able to obtain competitive accuracy on benchmarks. However, since the rank is strictly limited by the minimum of the number of points and frames [39], it becomes infeasible to solve large-scale problems with complex shape variations when the number of points is substantially smaller than the number of frames [94].

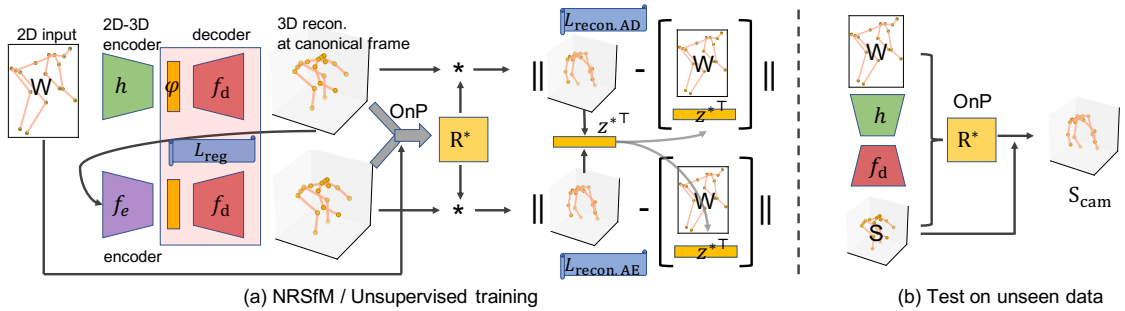



Figure 2.2: (a) In training, PAUL jointly optimizes the depth value $\mathbf{z}$, camera rotation $\mathbf{R}$ together with the network weights. It is realized through a bilevel optimization strategy, which analytically computes $\mathbf{R}^*$ and $\mathbf{z}^*$ as the solution to an OnP problem. The learning objective is formulated as a combination of reconstruction loss for the decoder-only stream (top row) and the auto-encoder (bottom row) together with regularizer $\mathcal{L}_{\text{reg}}$ applied on the code $\varphi$ and decoder's network weights; (b) in testing, only 2D-3D encoder $h$ and decoder $f_{\text{d}}$ are used. Camera rotation is directly estimated by OnP.

(ii) *union-of-subspaces* is inspired by the intuition that complex non-rigid deformations could be clustered into a sequence of simple motions [251]. It was extended to spatial-temporal domain [103] and structure from category [6]. The main limitation of using union-of-subspaces is how to effectively cluster deforming shapes from 2D measurements, and how to compute affinity matrix when the number of frames is huge.

(iii) *sparsity* [93, 95, 249], is a more generic prior compared to union-of-subspaces. However, due to the sheer number of possible subspaces to choose, it is sensitive to noise.

(iv) *Procrustean normal distribution* [109] assumes that the 3D structure follows a normal distribution if aligned to a common reference frame. It allows reconstruction without specifying ranks which are typically required by other methods. It was extended temporally as a Procrustean Markov process [111]. Limited by assuming normal distribution, it is less favorable to model deformation which is not Gaussian.

**Unsupervised 2D-3D lifting.** NRS*f*M can be recast for unsupervised learning 2D-3D lifting. Cha *et al*. [23] use low-rank loss as a learning objective to constraint the shape output of the 2D-3D lifting network. Park *et al*. [151] further modifies Cha's approach by replacing the camera estimation network with an analytic least square solution which aligns 3D structures to the mean shape of a sequence. Due to the inefficiency of low-rank to model complex shape variations, these methods are restricted to datasets with simpler

shape variations, or requires temporal order so as to avoid directly handling global shape variations.

Instead of using classical NRS*f*M priors, recent works explore the use of deeper constraints. Generative Adversarial Networks (GANs) [59] are used to enforce realism of 2D reprojections across novel viewpoints [26, 44, 99, 205]. These methods are only applicable for large datasets due to the requirement of learning GANs. It is also unclear how to directly learn GANs with training set existing missing data.

Novotny *et al*. [143] instead enforces self-consistency on the predicted canonicalization of the randomly perturbed 3D shapes. Kong & Lucey [94] proposed the use of hierarchical sparsity as constraint, and approximate the optimization procedure of hierarchical sparse coding as a feed-forward lifting network. It was recently extended by Wang *et al*. [211] to handle missing data and perspective projection. These approaches use complicated network architecture to enforce constraints as well as estimating camera motion, while our method uses simpler constraint formulation, and realized with efficient solution.

## 2.3 Preliminary

**Problem setup.** We are interested in the atemporal setup for unsupervised 2D-3D lifting, which is a general setup that not only works with single deforming objects, but also multiple objects from the same object category. Specifically, given a non-sequential dataset consist of $N$ frames of 2D keypoint locations $\{\mathbf{W}^{(1)}, \dots, \mathbf{W}^{(N)}\}$, where each $\mathbf{W} \in \mathbb{R}^{2\times P}$ represents 2D location for $P$ keypoints, and visibility masks represented as diagonal binary matrices $\{\mathbf{M}^{(1)}, \dots, \mathbf{M}^{(N)}\}$, we want to (i) recover the 3D locations for every keypoints in the dataset, and (ii) train a 2D-3D lifting network capable of making single frame prediction for unseen data.

**Weak perspective camera model.** We assume weak perspective projections, *i.e*. for a 3D structure $\mathbf{S}$ defined at a canonical frame, its 2D projection is approximated as:

$$\mathbf{W} \approx s\mathbf{R}_{xy}\mathbf{S} + \mathbf{t}_{xy} \tag{2.1}$$

where $\mathbf{R}_{xy} \in \mathbb{R}^{2\times 3}$, $\mathbf{t}_{xy} \in \mathbb{R}^2$ are the x-y component of a rigid transformation, and $s > 0$ is the scaling factor inversely proportional to the object depth if the true camera model is pin-hole. If all 2D points are visible and centered, $\mathbf{t}_{xy}$ could be omitted by assuming

the origin of the canonical frame is at the center of the object. Due to the bilinear form of (2.1), $s$ is ambiguous and becomes up-to-scale recoverable only when $\mathbf{S}$ is assumed to follow certain prior statistics. A typical treatment to handle scale is to approximate with orthogonal projection by normalizing the scale of $\mathbf{W}$, setting $s = 1$ and leaving $\mathbf{S}$ to be scaled reconstruction.

**Regularized auto-encoder (RAE) for $\mathbf{S}$.** We assume that the 3D shapes, if aligned to a canonical frame, are compressible by an undercomplete auto-encoder with a low-dimensional bottleneck, *i.e*.

$$\mathbf{S} \approx f_{\mathrm{d}} \circ f_{\mathrm{e}}(\mathbf{S}), \tag{2.2}$$

where $f_{\mathrm{e}}$ is the encoder which maps $\mathbf{S}$ to a $K$-dimensional latent code $\boldsymbol{\varphi} \in \mathbb{R}^K$ , $f_{\mathrm{d}}$ is the decoder function and $\circ$ denotes function composition. In this work, we choose deterministic RAE [57] instead of variational auto-encoder (VAE) [91] since RAE is easier to train and still leads to an equally smooth and meaningful latent space. The learning objective for RAE is a combination of reconstruction loss and regularizers on the latent codes as well as the decoder's weights,

$$\mathcal{L}_{\mathrm{RAE}}(\mathbf{x}; \boldsymbol{\theta}_{\mathrm{d}}, \boldsymbol{\theta}_{\mathrm{e}}) = \| f_{\mathrm{d}} \circ f_{\mathrm{e}}(\mathbf{x}) - \mathbf{x} \|_F + \mathcal{L}_{\mathrm{reg}}. \tag{2.3}$$

where $\boldsymbol{\theta}_{\mathrm{d}}, \boldsymbol{\theta}_{\mathrm{e}}$ are network weights for the auto-encode, $\mathbf{x}$ denote data samples, and the regularizer $\mathcal{L}_{\mathrm{reg}}$ is picked to be $\|\boldsymbol{\varphi}\|_2^2$ and weight decay, which was shown to give comparable performance to VAE when generating images and structured objects [57].

**2D-3D lifting network.** A 2D-3D lifting network is designed to take input from 2D keypoints and visiblity mask, and outputs 3D keypoint locations. We assume the network architecture is decomposed into two parts (i) a 2D-3D encoder $h$ which maps 2D observations to latent code $\boldsymbol{\varphi}$, and (ii) a decoder $f_{\mathrm{d}}$ (reused from the auto-encoder) to generate 3D shapes from $\boldsymbol{\varphi}$. Thus this type of 2D-3D lifting network can be expressed as $f_{\mathrm{d}} \circ h(\mathbf{W}, \mathbf{M})$, which is a general form for network architectures used in literature [94, 130, 143].

## 2.4 Learning Procrustean auto-encoder from 2D

For clarity, in this section we simplify the problem by assuming all points are visible, which allows removing translational component in (2.1). Description of handling occlusions are given in Sec. 3.3.2. Fig. 2.2 illustrates the proposed approach.

### 2.4.1 Learning objective

**Procrustean auto-encoder.** Directly compressing 3D shapes $\mathbf{S}_{\text{cam}}$ at camera frame is inefficient due to the inclusion of the degrees of freedom from camera motion. Therefore, we choose to impose compressibility on $\mathbf{S}$ at canonical frame as shown in (2.2). However, learning such auto-encoder from 2D observations requires overcoming several obstacles: (i) due to the objects being non-rigid, the definition of canonical frame is statistical and implicitly represented by the unknown rigid transformations to align $\mathbf{S}_{\text{cam}}$'s; (ii) choosing canonical frame requires knowing the statistics of $\mathbf{S}_{\text{cam}}$ which we do not have complete information, since only the first two rows of $\mathbf{S}_{\text{cam}}$ are given as $\mathbf{W}$ representing the x-y coordinates, while the 3rd row $\mathbf{z}$ representing depth values are unknown; (iii) reconstructing $\mathbf{S}_{\text{cam}}$ in turn requires the estimation of the rigid transformation as well as the statistical model of the shape. To overcome these, we propose a joint optimization scheme:

$$\min_{\substack{\theta_e,\theta_d,\\ \{\mathbf{z}^{(i)}\},\{\mathbf{R}^{(i)}\in SO(3)\}}} \sum_{i=1}^{N} \mathcal{L}_{\text{RAE}}({\mathbf{R}^{(i)}}^{\top} \begin{bmatrix} \mathbf{W}^{(i)} \\ {\mathbf{z}^{(i)}}^{\top} \end{bmatrix}; \boldsymbol{\theta}_e, \boldsymbol{\theta}_d), \tag{2.4}$$

where $\theta_e$, $\theta_d$ are network weights for the auto-encoder, $\mathbf{R}^{\top} \begin{bmatrix} \mathbf{W} \\ \mathbf{z}^{\top} \end{bmatrix}$ computes $\mathbf{S}$ at the canonical frame.

However, (2.4) is still steps away from being applicable to unsupervised 2D-3D lifting, since it misses the 2D-3D lifting network module in the objective function, and is difficult to optimize due to the inclusion of unknown rotation matrices in the input of the auto-encoder. In the following, we address these by reparameterizing the learning objective and propose an efficient optimization scheme.

**Reparameterization for learning 2D-3D lifting.** First we introduce an auxiliary variable as the latent code $\boldsymbol{\varphi}$, which satisfies

$$f_d(\boldsymbol{\varphi}) = \mathbf{R}^{\top} \begin{bmatrix} \mathbf{W}^{\top} & \mathbf{z} \end{bmatrix}^{\top}. \tag{2.5}$$

This leads to transforming (2.4) to a constrained optimization objective with (2.5) as the

constraint and the input to $f_{\mathrm{d}} \circ f_{\mathrm{e}}$ replaced by $f_{\mathrm{d}}(\boldsymbol{\varphi})$,

$$\begin{aligned} \min_{\substack{\theta_{\mathrm{e}},\theta_{\mathrm{d}},\\ \{\mathbf{R}^{(i)}\in SO(3)\}\\ \{\mathbf{z}^{(i)},\,\boldsymbol{\varphi}^{(i)}\}}} & \sum_{i=1}^{N} \underbrace{\| f_{\mathrm{d}} \circ f_{\mathrm{e}} \circ f_{\mathrm{d}}(\boldsymbol{\varphi}^{(i)}) - \mathbf{R}^{(i)\top} \begin{bmatrix} \mathbf{W}^{(i)} \\ \mathbf{z}^{(i)\top} \end{bmatrix} \|_F}_{\mathcal{L}_{\text{recon. AE}}(\boldsymbol{\varphi},\mathbf{R},\mathbf{z};\theta_{\mathrm{e}},\theta_{\mathrm{d}})} + \mathcal{L}_{\text{reg}}, \\ \text{s.t. } & f_{\mathrm{d}}(\boldsymbol{\varphi}) = \mathbf{R}^{\top} \begin{bmatrix} \mathbf{W}^{\top} & \mathbf{z} \end{bmatrix}^{\top}. \end{aligned} \tag{2.6}$$

Depending on the type of task, $\boldsymbol{\varphi}$ could be either treated as free variables to optimize if the task is to reconstruct the 'training' set as a NRS*f*M problem, or $\boldsymbol{\varphi}$ could be the network output of the 2D-3D encoder *i.e.* $\boldsymbol{\varphi} = h(\mathbf{W};\theta_{\mathrm{h}})$. We then relax the constrained optimization into an unconstrained one, which allows passing gradients to the weights of the 2D-3D encoder $h$,

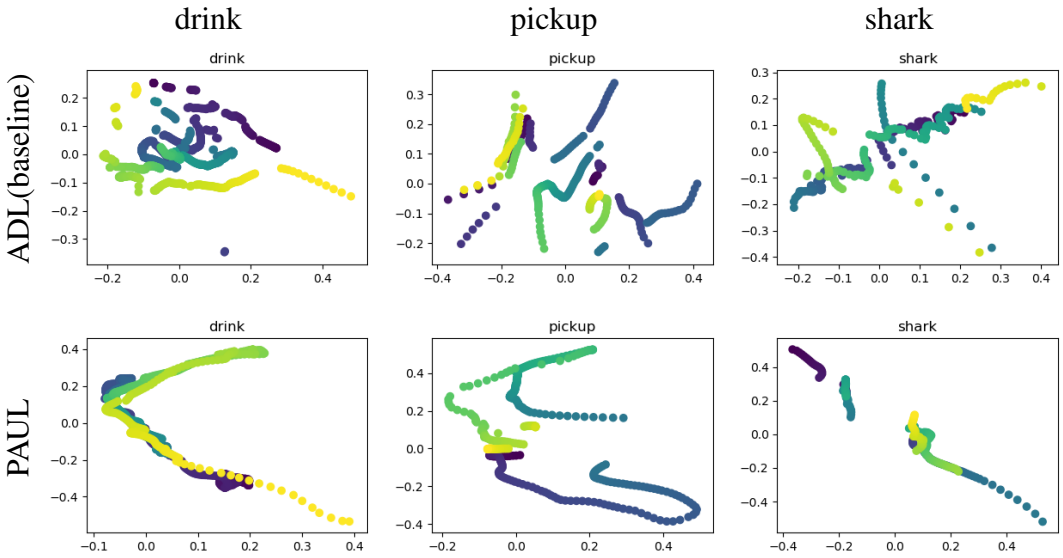
$$\begin{aligned} \min_{\substack{\theta_{\mathrm{h}},\theta_{\mathrm{e}},\theta_{\mathrm{d}},\\ \{\mathbf{z}^{(i)}\},\{\mathbf{R}^{(i)}\in SO(3)\}}} & \sum_{i=1}^{N} \mathcal{L}_{\text{recon. AE}}(h(\mathbf{W}^{(i)}),\mathbf{R}^{(i)},\mathbf{z}^{(i)}) \\ & + \| f_{\mathrm{d}} \circ h(\mathbf{W}^{(i)}) - \mathbf{R}^{(i)\top} \begin{bmatrix} \mathbf{W}^{(i)} \\ \mathbf{z}^{(i)\top} \end{bmatrix} \|_F + \mathcal{L}_{\text{reg}}. \end{aligned} \tag{2.7}$$

This loss function could be understood as the combination of the reconstruction losses for both an auto-encoder and an auto-decoder together with the regularizer from RAE, *i.e.* $\mathcal{L}_{\text{recon. AE}} + \mathcal{L}_{\text{recon. AD}} + \mathcal{L}_{\text{reg}}$.

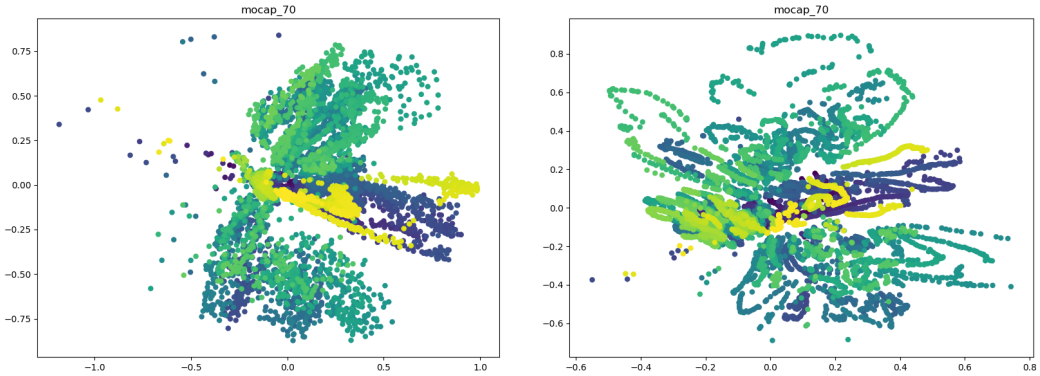
**Relation to learning with auto-decoder.** We note that an alternative auto-decoder approach with learning objective $\mathcal{L}_{\text{recon. AD}} + \mathcal{L}_{\text{reg}}$ is applicable, the additional $\mathcal{L}_{\text{recon. AE}}$ in our approach is to enforce the existence of a continuous inverse mapping from 3D shape to latent code. This encourages shapes with small variation to stay close in the latent space, which is helpful to learn a meaningful and smoother latent space. We investigate both approaches in Fig. 2.5 and compare the learnt latent space visually in Fig. 2.3.

### 2.4.2 Efficient bilevel optimization

Directly optimizing (2.7) with gradient descent is inefficient due to (i) the objective is non-convex and it is prone to poor local minima especially with respect to $\mathbf{R}$. One could use an off-the-shelf NRS*f*M method to provide initialization for $\mathbf{R}$ [177]. However this would make the solution sensitive to the accuracy of the chosen NRS*f*M algorithm. (ii) when using SGD for large datasets, it is problematic to properly update $\mathbf{R}^{(i)}$ and $\mathbf{z}^{(i)}$ if they are left as independent variables. Alternatively, one could introduce additional networks




(a) 2D-latent space on **short** sequences with **smooth** camera trajectories.




(b)

ADL (baseline) PAUL

2D-latent space on **long** sequence (CMU-MoCap S70) perturbed with **random** cameras.

Figure 2.3: Visualization of 2D latent space for ADL & PAUL. Each point represents the 2D latent code recovered for each frame of a sequence. The color of points (from dark blue to bright yellow) indicates the temporal order of points. Ideally, the points should form trajectories in the temporal order. PAUL gives clearer trajectory-like structures in its latent space, while ADL's recovered codes are either more spread-out or form broken trajectories.

to output $\mathbf{R}$ or $\mathbf{z}$ conditioned on 2D inputs [94, 205]. We find this unnecessary because it introduces extra complexity to solve the problem but is still subject to the inefficiency of gradient descent.

For a more efficient optimization strategy, we propose to first rearrange (2.7) to an equivalent bilevel objective:

$$\min_{\theta_{\mathrm{h}},\theta_{\mathrm{d}},\theta_{\mathrm{e}}} \sum_{i=1}^{N} \min_{\mathbf{R}^{(i)}\in SO(3),\ \mathbf{z}^{(i)}} \mathcal{L}^{(i)}_{\text{recon. AE}} + \mathcal{L}^{(i)}_{\text{recon. AD}} + \mathcal{L}^{(i)}_{\text{reg}}, \tag{2.8}$$

The benifit of this rearrangement is that the lower level problem, *i.e*. minimizing the reconstruction losses with respect to $\mathbf{R}$ and $\mathbf{z}$ can be viewed as an extension of the orthographic-N-point (OnP) problem [181], which allows the use of efficient solvers [18, 65, 134]. In addition, if an OnP solver refined by geometric loss is able to converge to local minima, it is not required to be differentiable due to the fact that both lower-level and upper-level problems share the same objective function, thus the gradient is zero at local minima [63]. This would lift the restrictions for the type of solvers we could use for the lower-level problem.

**Differentiable fast solver for the lower-level problem.** On the other hand, we opt to use an algebraic solution which is computationally more light-weight compared to OnP solvers iteratively minimizing the geometric error. The compromise of using an approximate (*e.g*. algebraic) solution is that, since it does not not necessarily reach local minima, it is required to be implemented as a differentiable operator, which could be easily accomplished via modern autograd packages. The solution we picked is:

1. Find the closed-form least square solution $\tilde{\mathbf{R}}^*$ for minimizing the reprojection error:

$$\min_{\tilde{\mathbf{R}}} \|\tilde{\mathbf{R}}(f_{\mathrm{d}} \circ f_{\mathrm{e}} \circ f_{\mathrm{d}})(\boldsymbol{\varphi}) - \mathbf{W}\|_2^2 + \|\tilde{\mathbf{R}} f_{\mathrm{d}}(\boldsymbol{\varphi}) - \mathbf{W}\|_2^2. \tag{2.9}$$

2. Project $\tilde{\mathbf{R}}^*$ to become a rotation matrix $\mathbf{R}^* \in SO(3)$ using SVD.
3. $\mathbf{z}^* = \frac{1}{2}((f_{\mathrm{d}} \circ f_{\mathrm{e}} \circ f_{\mathrm{d}})^\top(\boldsymbol{\varphi})\mathbf{r}_z^* + f_{\mathrm{d}}(\boldsymbol{\varphi})^\top \mathbf{r}_z^*)$, which is the closed-form least square solution for minimizing:

$$\min_{\mathbf{z}} \|(f_{\mathrm{d}} \circ f_{\mathrm{e}} \circ f_{\mathrm{d}})^\top(\boldsymbol{\varphi})\mathbf{r}_z^* - \mathbf{z}\|_2^2 + \|f_{\mathrm{d}}(\boldsymbol{\varphi})^\top \mathbf{r}_z^* - \mathbf{z}\|_2^2, \tag{2.10}$$

where $\mathbf{r}_z^{*\top}$ denotes the 3rd row of $\mathbf{R}^*$.

**End-to-end training.** Finally, with the approximate solution $\mathbf{R}^*$, $\mathbf{z}^*$ for the lower-level problem, the learning objective once again becomes a single level one, which is identical to (2.7) except that $\mathbf{R}$, $\mathbf{z}$ instead of being free variables, they are now replaced by $\mathbf{R}^*$, $\mathbf{z}^*$ which are differentiable functions conditioned on the network weights $\theta_{\text{h}}, \theta_{\text{e}}, \theta_{\text{d}}$. This allows learning these weights end-to-end via gradient descent.

**Prediction on unseen data.** To make 3D prediction of a single frame from unseen data, we first use the learned 2D-3D encoder $h$ and the decoder $f_{\text{d}}$ to compute $\mathbf{S}$ at the canonical frame, and then run OnP algorithm [181] to align it to the camera frame.

### 2.4.3 Handling missing data

If there exists 2D keypoints missing from the observation due to occlusions or out of image, the translational component in (2.1) is no longer removable simply by centering the visible 2D points. To avoid reintroducing $\mathbf{t}$ which would complicate derivations, we choose to follow the object centric trick to absorb translation through adaptively normalizing $\mathbf{S}$ according to the visibility mask $\mathbf{M}$. The normalized $\tilde{\mathbf{S}}$ is computed as:

$$\tilde{\mathbf{S}} = \mathbf{S} + \mathbf{S}(\mathbf{I}_P - \mathbf{M})\mathbf{1}_P\mathbf{1}_P^\top. \tag{2.11}$$

with this, the projection equation remains bilinear, *i.e.* $\tilde{\mathbf{W}} = \mathbf{R}_{\text{xy}}\tilde{\mathbf{S}}$, where $\tilde{\mathbf{W}}$ denotes the centered $\mathbf{W}$ by the average of visible 2D points. This allows to adapt PAUL to handle missing data with minimal changes. The detailed description and derivation of (2.11) will be provided in the next chapter.

## 2.5 Experiments

### 2.5.1 Implementation details

**Network architecture.** Throughout our experiment, we use the same auto-encoder architecture across datasets except the bottleneck dimension. The number of neural units in each layer is decreased exponentially, *i.e.* $\{256, 128, 64, 32, 16\}$. Ideally, if validation set with 3D groundtruth is provided, we could select optimal architecture based on cross validation. However, due to the unsupervised setting, we rather set the hyperparameters heuristically.

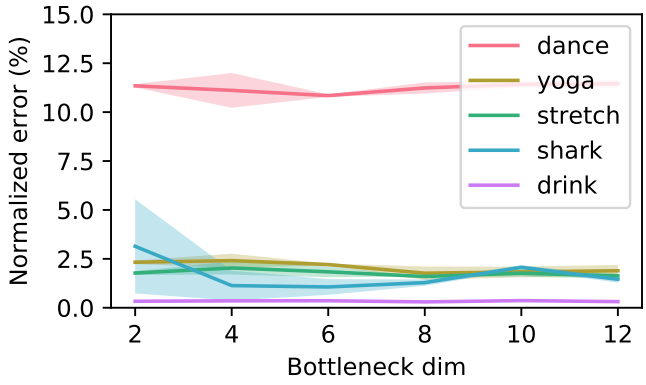



Figure 2.4: 3D reconstruction error with different bottleneck dimensions. For each configuration, PAUL is run 10 times and visualize with average accuracy (solid lines) together with standard deviation (colored area).

We pick a smaller bottleneck dimension, *i.e*. 4 for smaller datasets (*e.g*. synthetic NRSfM benchmarks) or datasets with mostly rigid objects (*e.g*. Pascal3D+), and pick a larger dimension, *i.e*. 8 for articulated objects such as human skeleton (H3.6M, CMU motion capture dataset) and meshes (UP3D). The robustness of our method against variations in hyperparameter settings is investigated in Sec. 2.5.3.

For the 2D-3D encoder, we experiment with both fully connected residual network [130] and convolutional network [94]. The only modification we make to those architecture is the dimension of their output so as to match the picked bottleneck dimension.

**Training details.** We keep the same weightings for $\mathcal{L}_{\text{reg}}$ across all experiments, *i.e*.

$$\mathcal{L}_{\text{reg}} = 0.01\|\boldsymbol{\varphi}\|_2^2 + 10^{-4}\|\boldsymbol{\theta}_{\text{d}}\|_2^2. \tag{2.12}$$

We use the Adam optimizer [90] for training. The optimization parameters are tuned according to specific datasets so as to guarantee convergence.

**Evaluation metric.** We follow two commonly used evaluation protocols:
(i) *MPJPE* evaluates the mean per-joint position error. To account for the inherent ambiguity from weak perspective cameras, we flip the depth values of the reconstruction if it leads to lower error. To account for the ambiguity in the object distance, we either subtract the average depth values or subtract the depth value of a root keypoint. The latter is used only for H3.6M dataset due to the evaluation convention in literature.

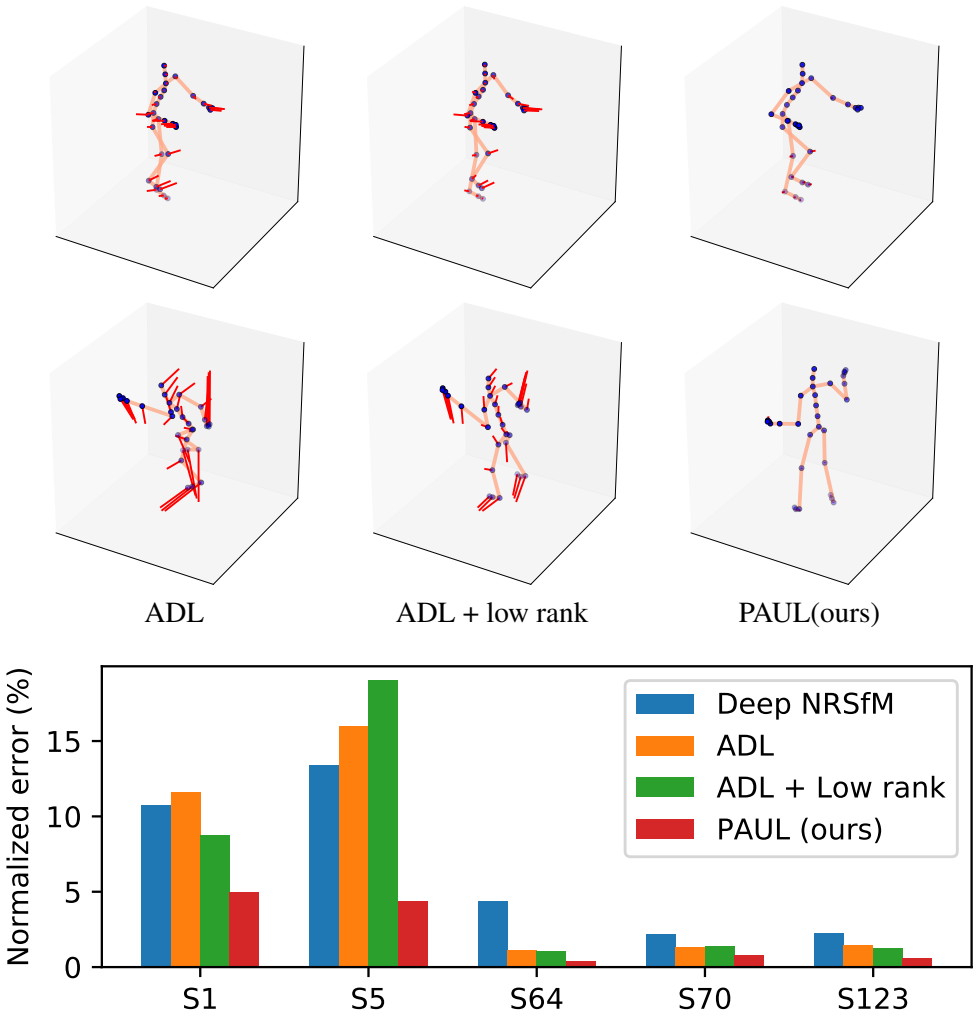



Figure 2.5: Comparison with auto-decoder baseline (*i.e.* ADL), and low rank constraint (ADL + low-rank) on CMU motion capture dataset. PAUL gives significantly more accurate reconstruction compared to ADL and low-rank. Red line visualizes the difference between reconstructed and groundtruth points.

| | short sequences | | | | | | long sequences (random cam. motion) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | drink | pickup | yoga | stretch | dance | shark | S1 | S5 | S64 | S70 | S123 |
| #frames | 1102 | 357 | 307 | 370 | 264 | 240 | 45025 | 13773 | 11621 | 10788 | 10788 |
| CNS [110] | 3.04 | 9.18 | 11.15 | 7.97 | **7.59** | 8.32 | 37.62 | 40.02 | 29.00 | 26.26 | 26.46 |
| PND [109] | **0.37** | 3.72 | 1.40 | 1.56 | 14.54 | 1.35 | - | - | - | - | - |
| BMM [39] | 2.66 | 17.31 | 11.50 | 10.34 | 18.64 | 23.11 | 16.45 | 14.07 | 18.13 | 18.91 | 19.32 |
| BMM-v2 [101] | 1.19 | 1.98 | **1.29** | **1.44** | 10.60 | 5.51 | - | - | - | - | - |
| Deep NRSfM [94] | 17.38 | **0.53** | 12.54 | 21.63 | 20.95 | 21.83 | 10.74 | 13.40 | 4.38 | 2.17 | 2.23 |
| PAUL | 0.47 | 2.03 | 1.71 | 1.62 | 10.22 | **0.37** | **4.97** | **4.38** | **0.39** | **0.77** | **0.59** |

Table 2.1: Comparison with state-of-the-art NRS*f*M methods on both short sequences and long sequences, report with normalized error. Long sequences are sampled from CMU motion capture dataset [1] with large random camera motion. Atemporal methods are highlighted by orange, methods using temporal information are marked by green. Due to the code for PND and BMM-v2 is unavailable, they are excluded from evaluation on CMU motion capture sequences.

| | aero. | car | tv. | sofa | motor. | dining. | chair | bus | bottle | boat | bicycle | train | Mean | 8 cls. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C3DPO | 6.56 | 8.21 | 15.03 | 7.30 | 7.48 | **3.77** | 3.46 | 20.41 | 7.48 | **7.58** | 3.47 | 33.70 | 10.4 | 7.58 |
| Deep NRS*f*M++ | 7.51 | 9.22 | 17.43 | 9.37 | 6.18 | 12.90 | 3.97 | 18.02 | 2.08 | 9.18 | 4.03 | **23.67** | 10.3 | 8.90 |
| PAUL | **3.99** | **7.13** | **9.88** | **3.99** | **3.74** | 5.70 | **2.19** | **14.11** | **1.03** | 8.08 | **1.74** | 38.78 | **8.4** | **5.32** |

Table 2.2: Per-category normalized error (%) on Pascal3D+ dataset. Follow the protocol of Agudo *et al*. [6], we further report the average error of 8 object categories which are annotated with $\geq 8$ keypoints.

(ii) *Normalized error* (**NE**) evalutes the relative error by: $\|\mathbf{S}_{\text{pred}} - \mathbf{S}_{\text{GT}}\|_F / \|\mathbf{S}_{\text{GT}}\|_F$.

### 2.5.2 Baselines

**Auto-decoder lifting (ADL).** As discussed in Sec. 2.4.1, an alternative approach for unsupervised lifting is only minimizing the loss $\mathcal{L}_{\text{recon. AD}} + \mathcal{L}_{\text{reg}}$, without the term $\mathcal{L}_{\text{recon. AE}}$ for the auto-encoder. Hence for this baseline, we are only training with respect to the decoder, thus regarded as an auto-decoder approach.

**ADL + low rank.** In addition, we experiment with adding the low rank constraint as another baseline. Similar to Cha *et al*. [23] and Park *et al*. [151], we evaluate the nuclear norm of the output of the shape decoder as the approximate low rank loss, *i.e*. $\|\mathbf{S}\|_*$. We empirically pick the weighting for the low rank loss as 0.01.

| | UP3D 79KP | Pascal3D+ |
|---|---|---|
| avg occlusion % | 61.89 | 37.68 |
| EM-S*f*M [194] | 0.107 | 131.0 |
| GbNRS*f*M [50] | 0.093 | 184.6 |
| Deep NRS*f*M [94] | 0.076 | 51.3 |
| C3DPO [143] | 0.067 | 36.6 |
| Deep NRS*f*M++ [211] | 0.062 | 34.8 |
| PAUL | **0.058** | **30.9** |

Table 2.3: Comparison on datasets with high percentage of missing data. Test accuracy is reported with MPJPE.

| | GT pts. | SH pts. [143] |
|---|---|---|
| Pose-GAN [99] | 130.9 | 173.2 |
| C3DPO [143] | 95.6 | 153.0 |
| PRN [151] | 86.4 | 124.5 |
| PAUL | 88.3 | 132.5 |

Table 2.4: MPJPE on H3.6M validation set. orange indicates atemporal method and green indicates methods use temporal information.

| | NE (%) |
|---|---|
| C3DPO [143] | 35.09 |
| PRN [151] | 13.77 |
| PAUL (ours) | **12.36** |
| PAUL (train set) | 4.30 |

Table 2.5: Test accuracy on SURREAL synthetic sequences. Training error is also reported for PAUL (bottom row).

### 2.5.3 NRS*f*M experiments

In the first set of experiments, we evaluate the proposed method for the NRS*f*M task, where we report how well the compared methods are able to reconstruct a dataset. The goal is to evaluate the robustness of the proposed Procrustean auto-encoder shape prior across different shape variations, without being convoluted by the inductive bias from a 2D-3D lifting network, which is not the interest of this work. To achieve this, on short sequences, instead of conditioning $\varphi$ with a 2D-3D encoder, we treat $\varphi$ as free variable to optimize directly; and on long sequences, we use the same 2D-3D encoder as in Deep NRS*f*M [94] to have a fair comparison.

**NRS*f*M datasets.** We report performance on two types of datasets: (i) short sequences with simple object motions, *e.g*. *drink, pickup, yoga, stretch, dance, shark* which are standard benchmarks used in NRS*f*M literature [9, 194].
(ii) long sequences with large articulated motions, *i.e*. CMU motion capture dataset [1]. We use the processed data from Kong & Lucey [94] which is intentionally made more challenging by inserting large random camera motions.

**Robustness against bottleneck dimension.** As shown in Fig. 2.4, we run the methods with varying bottleneck dimension from 2 to 12 on different datasets. To account for the stochastic behavior due to network initialization and gradient descent on small datasets, we run the methods 10 times and visualize with average accuracy (solid lines) together with standard deviation (colored area). PAUL gives stable results once the bottleneck dimension is sufficiently large. This indicates that PAUL is practical for unseen datasets by using an overestimated bottleneck dimension.

**Comparison with ADL and low rank.** As shown in Fig. 2.5, on sequences from CMU motion capture dataset, ADL achieves lower error in most sequences when comparing against Deep NRS*f*M, indicating it is indeed a strong baseline. Augmenting ADL with low rank constraint is able to further decrease error for several sequences, but the improvement is not consistent across the whole dataset. In comparison, PAUL gives significant error reduction for all the evaluated sequences, which demonstrates the effectiveness of the proposed Procrustean anto-encoder prior.

**Comparison with state-of-the-art NRS*f*M methods.** Table 4.1 collects results from some of the state-of-the-art NRS*f*M methods on the synthetic bechmarks, *e.g*. BMM-v2 [101], CNS [110] and PND [109]. All the well-performing methods utilize temporal information while PAUL does not, but still achieves competitive accuracy on short sequences. On long sequences from CMU motion capture dataset, the accuracy of temporal-based methods *e.g*. CNS deteriorates significantly due to the data perturbed by large random camera motion. Atemporal methods on the other hand gives stable results and PAUL outcompetes all the compared methods by a wide margin.

### 2.5.4 2D-3D lifting on unseen data

We compare against recent unsupervised 2D-3D lifting methods on the processed datasets by Novotny *et al*. [143]:

**Datasets.** (i) *Synthetic UP-3D* is a large synthetic dataset with dense human keypoints collected from the UP-3D dataset [108]. The 2D keypoints are generated by orthographic projection of the SMPL body shape with the visibility computed from a ray traccer. Similar to C3DPO, we report result for 79 representative vertices of the SMPL on the test set;

(ii) *Pascal3D+* [227] consists of images from 12 object categories with sparse keypoint annotations. The 3D keypoint groundtruth are created by selecting and aligning CAD models. To ensure consistency between 2D keypoint and 3D groundtruth, the orthographic projections of the aligned 3D CAD models are used as 2D keypoint annotations, and the visibility mask are taken from the original 2D annotations. For a fair comparison against C3DPO, we use the same fully-connected residual network as the 2D-3D encoder, and train a single model to account for all 12 object categories.

(iii) *Human 3.6 Million dataset* (H3.6M) [83] is a large-scale human pose dataset annotated by motion capture systems. Following the commonly used evaluation protocol, the first 5

human subjects (1, 5, 6, 7, 8) are used for training and 2 subjects (9, 11) for testing. The 2D keypoint annotations of H3.6M preserves perspective effect, thus is a realistic dataset for evaluating the practical usage of 2D-3D lifting.

**Robustness against occlusion.** Both synthetic UP3D and Pascal3D+ dataset simulate realistic occlusions with high occlusion percentage. We focus our comparison against C3DPO and Deep NRS*f*M++ [211] which is a recent update of Deep NRS*f*M for better handling missing data and perspective projections. As shown in Table 2.3, PAUL significantly outperforms both of them. To account for the distortion caused by the object scale, we switch the evaluation metric from MPJPE to normalized error in Table 2.2 and report per-class error. PAUL leads with even bigger margin.

**Robustness against labeling noise.** To work with in-the-wild data, 2D-3D lifting methods are required to be robust against annotation noise, which could be simulated by using 2D keypoints detected by a pretrained keypoint detector. In addition, 2D annotation with perspective effect could also be regarded as noise since it is not modeled by the assumed weak perspective camera model. We evaluate both scenarios on H3.6M dataset (see Table 3.6). PAUL outperforms the compared atemporal methods (*i.e*. C3DPO and Deep NRS*f*M++) and is competitive to recently proposed PRN [151] which requires training data to be sequential.

### 2.5.5 Dense reconstruction

We follow the comparison in Park *et al*. [151] on the synthetic SURREAL dataset [202], which consists of 5k frames with 6890 points for training, and 2,401 frames for testing. Unlike PRN [151] which subsamples a subset of points when evaluating the low rank shape prior due to the intense computational cost of evaluating nuclear norm, our auto-encoder shape prior is computationally cheaper when dealing with dense inputs, thus we made no modification when applying PAUL to SURREAL. As shown in Table 2.5, PAUL achieves lower test error compared to PRN, even though we use no temporal information in training. It is worth to point out that the current bottleneck in achieving better test accuracy is at the generalization ability of the 2D-3D encoder network, not at the proposed unsupervised training framework. As shown in the last row of Table 2.5, the reconstruction error on the training set is already much lower than the test error (*i.e*. 4.30% vs 12.36%).

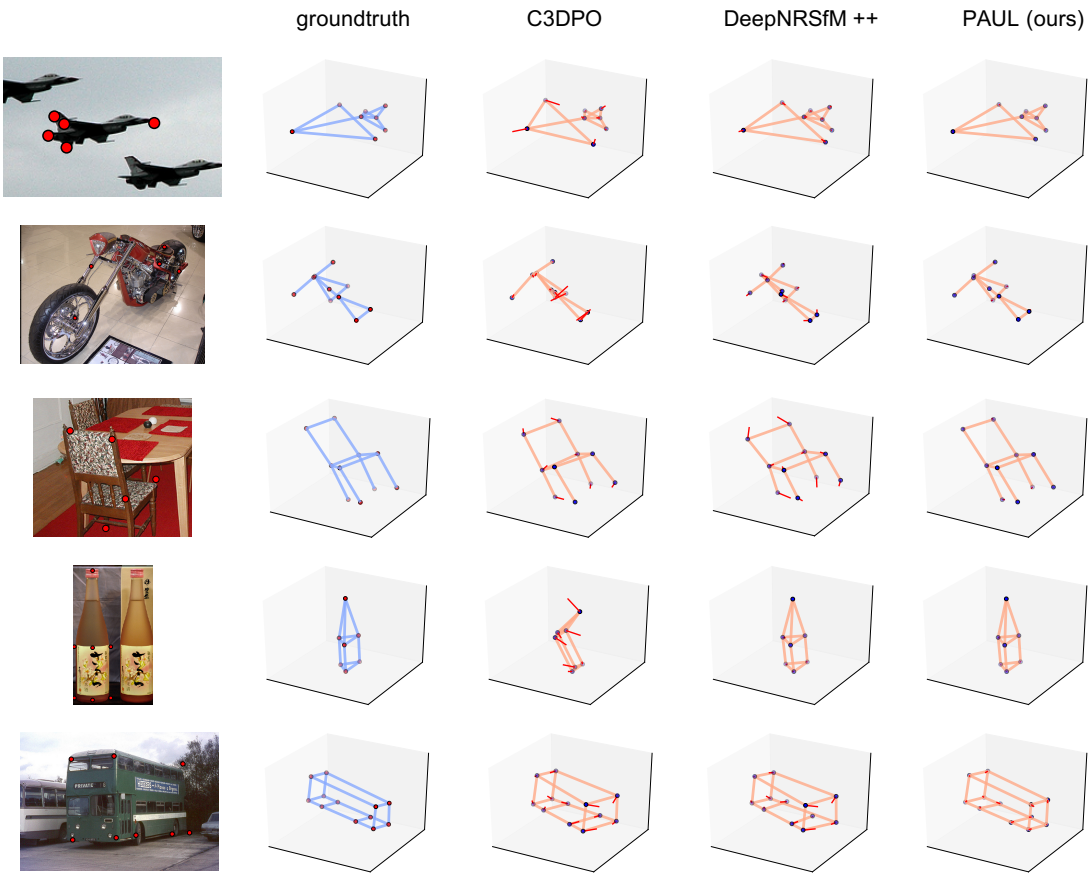



Figure 2.6: Qualitative comparison on Pascal3D+ dataset. Red lines visualize the difference between groundtruth points and predicted points. PAUL shows more accurate prediction in the compared samples.

## 2.6 Conclusion

We propose learning a Procrustean auto-encoder for unsupervised 2D-3D lifting capable of learning from no-sequential 2D observations with large shape variations. We demonstrate that having an auto-encoder performs favorably compared to an alternative auto-decoder approach. The proposed method achieves state-of-the-art accuracy across NRS*f*M and 2D-3D lifting tasks. For future work, theoretical analysis of the characterization of the solution (*e.g.* uniqueness) may help inspire further development. Interpreting the approach as learning manifold may also help provide guidance such as setting hyperparameters [22]. Finally, it is straightforward to extend the method to model perspective projection using

similar extensions outlined in [211, 244].

## 2.7 Appendix

### 2.7.1 Additional discussion of latent space in PAUL

PAUL uses the constraint that complex shape variation is compressible into a lower dimensional latent space with an auto-encoder. The use of latent space in our problem is different to generative modeling in that: (i) we focus on the compressibility instead of compactness of the representation, *i.e*. we do not require that any linear interpolation between two latent codes still corresponds to a valid 3D shape. (ii) due to the shape coverage from short sequences in NRS*f*M is sparse and arbitrary, we do not impose any prior distribution such as Gaussian on the latent space [91, 236]. However, these do not prevent us from sampling the learned latent space, which can be achieved with an ex-post density estimation step as shown by Ghosh *et al*. [57].

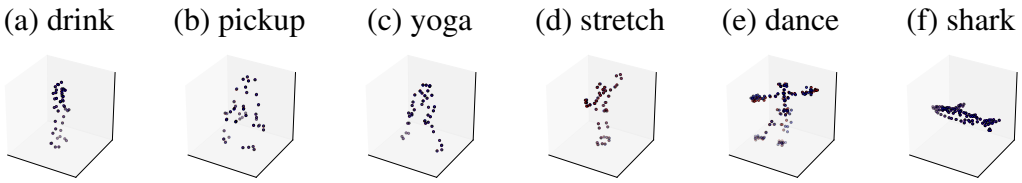



Figure 2.7: Results on NRSfM synthetic short sequences. blue: points reconstructed by PAUL; red: groundtruth points. Note the recovered points mostly overlaps with the groundtruth points, indicating the good performance of PAUL.

**auto-encoder v.s. decoder-only lifting.** In the main paper, we showed that our auto-encoder-based approach (PAUL) empirically outperforms the decoder-only baseline (*i.e*. ADL). The theoretical difference between the two is that auto-encoders additionally enforce the existence of a continuous mapping from the 3D shape to the low-dimensional latent space through learning the encoder network. The implication of this additional constraint is that it encourages shapes with small variation to stay close in the latent space. Consequently it improves the uniqueness of the 2D-3D lifting solution, as it implicitly forces each 3D shape to have a unique latent representation.

We analyze the difference by visualizing the latent space for both PAUL and ADL. We set the bottleneck dimension as 2 for both methods, and plot the 2D latent code for

each sample in the input sequence (see Fig. 2.3). The color (from dark blue to bright yellow) for each point represents its temporal order in the sequence. Since the motion is by nature smooth, temporarily close frames have smaller 3D shape variation, thus ideally their corresponding latent codes should be close to each other. This implies that a continuous code trajectory is expected in the visualization.

In Fig. 2.3(a), we first compare PAUL and ADL on short sequences with smooth camera trajectory. As expected, we observe that the reconstructed 2D codes from PAUL indeed forms one or a small number of continuous trajectories. We can also observe recurrent motion from the loop in the code trajectory on the drink sequence. In comparison, without the constraint from the encoder, ADL produces a set of more spread-out latent codes, which forms a number of shorter trajectories in a less interpretable spatial order.

One may argue that since a 2D-3D encoder is also a continuous mapping, it should partially fullfill the role of an encoder to encourage smoothness of the latent space. However, the counterargument to rely on the inductive-bias from a 2D-3D encoder is – since 2D projection is a combination of 3D shape and camera pose, small variation in 3D shape does not necessarily leads to small variation in 2D. Hitherto, the continuity of the output of a 2D-3D encoder conditioned on 2D inputs does not translates to the continuity of codes with respect to 3D shapes. To support this counterargument, we conduct experiment on a sequence with random camera motion (see Fig. 2.3(b)) which means temporal adjacent frames would have very different 2D projections. ADL with a 2D-3D encoder (adopted from Deep NRS*f*M [94]) loses the trajectory-like structure in its latent space, which PAUL preserves.

## 2.8 Additional results

**Robustness to noise.** We investigate the robustness of PAUL against noise by perturbed the groundtruth 2D keypoints using Gaussian noise with different standard deviation. We further investigate the implication of using different bottleneck dimension when learning from noisy data. As shown in Fig. 2.8, on CMU motion capture dataset, PAUL learned with clean data produces stable reconstruction accuracy with bottleneck dimension from 4-12. When noise is inserted to the training data, PAUL still keeps similar performance when the bottleneck dimension is relatively small (*e.g.* 4-8). The accuracy decrease becomes more noticeable only when the bottleneck dimension is large (*e.g.* 12). This indicates that PAUL

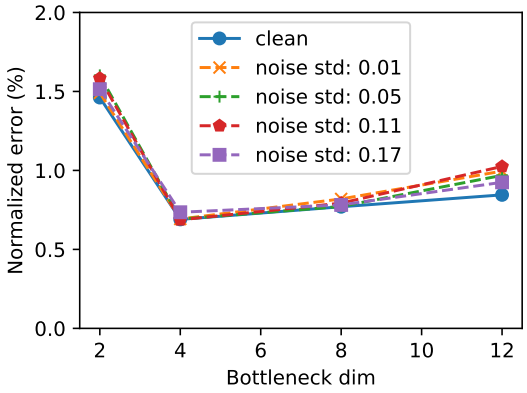



Figure 2.8: Results on CMU motion capture dataset S70 with different level of noise and bottleneck dimension.

overall is robust against random Gaussian noise, and a good practise to apply PAUL to real data is to start from smaller bottleneck dimension, which provides stronger constraint to denoise the reconstruction.

# Chapter 3

# Torwards Unsupervised 2D-3D Lifting in the Wild

## 3.1 Introduction

Non-rigid Structure from Motion (NRS*f*M) aims to reconstruct the 3D structure of a deforming object from 2D keypoint correspondences observed from multiple views [9, 19, 39, 93, 100, 103]. While the object deformation has classically been assumed to occur in time, the vision community has increasingly drawn attention to *atemporal* applications – commonly known as unsupervised 2D-3D lifting. Notable examples include (i) Structure from Category (S*f*C) [5, 6, 54, 95], where the deformation exists as the shape variation within a category-specific object dataset, and (ii) unsupervised human pose estimation [26, 99, 209], which aims to recover the 3D human structure from 2D keypoint annotations. More recently, learning-based methods [23, 94, 143] have shown impressive results for solving atemporal NRS*f*M problems. These models are advantageous because (i) they are scalable to large datasets and (ii) they allow fast feed-forward predictions once trained, without the need of rerunning costly optimization procedures for new samples during inference.

In this regard Deep NRS*f*M [94] is of particular interest. Deep NRS*f*M at its heart is a factorization method that learns a series of dictionaries end-to-end for the joint recovery of 3D shapes and camera poses. Unlike classical NRS*f*M approaches, it incorporates

hierarchical block sparsity as the shape prior, which was shown to be more expressible than the popular low-rank assumption [8, 19, 39, 50, 101] and more robust than single-level block sparsity [93, 95]. A key concept in Deep NRS*f*M is its imposition of this prior in the *architectural design* of the network. This allows one to solve the hierarchical block-sparse dictionary learning problem by solely optimizing the reprojection objective end-to-end using deep learning as the machinery. This also makes such structured bias in the network architecture to be interpretable as the optimization procedure used for estimating block-sparse codes. Deep NRS*f*M has shown superior performance on S*f*C and temporally unordered motion capture datasets, outperforming comparable NRS*f*M methods [6, 61, 71, 110] by an order of magnitude.

Despite the recent advances on NRS*f*M, however, two common drawbacks to state-of-the-art NRS*f*M methods [61, 62, 71, 101, 103] still remain: (i) the assumption of an *orthogonal camera model* and (ii) the need for cumbersome post-processing (*e.g.* low-rank matrix completion) to handle *missing data* (keypoints). This makes them impractical for most datasets collected in the wild, where images can exhibit strong perspective effects and structured missing keypoint annotations due to self/out-of-view occlusions. Although perspective effects have been studied in physics-based NRS*f*M [33, 104, 147, 204] where priors such as isometric deformation are enforced, they rarely hold for generic atemporal shapes (*e.g.* object categories in S*f*C) under consideration in this paper. The application of a perspective camera model to factorization methods, on the other hand, is nontrivial. Specifically, the bilinear relationship of 2D observations to the factorized 3D shape and camera matrices in (3.1) breaks down; as a consequence, the naïve block sparsity prior would no longer mathematically hold.

In this paper, we address the above issues of recent learning-based NRS*f*M methods and provide a solution to incorporate perspective cameras and handle missing data at a theoretical angle. We focus on Deep NRS*f*M [94] and show that the bilinear factorization relationship in (3.1) can be preserved by centering the common reference frame to the object center. This seemingly simple step, which draws inspiration from Procrustes analysis [14, 64, 109] and previously applied in Perspective-n-Point (PnP) problem [244], circumvents the need for explicitly modeling translation and projective depths in NRS*f*M. The resulting bilinear factorization induces a novel 2D-3D lifting network architecture which is adaptive to camera models as well as the visiblity of input 2D points. We refer to our approach as **Deep NRS*f*M++**, which achieves state-of-the-art results across a myriad of benchmarks.

In addition, we demonstrate significant improvements of Deep NRS*f*M++ in reconstruction accuracy by correcting an inherent caveat of Deep NRS*f*M, which made unwarranted relaxations of the block sparsity prior that deviates from the true block-sparse coding objective.

Our contributions are summarized as below:

- We derive a bilinear factorization formulation to incorporate perspective projection and handle missing data in learning-based NRS*f*M methods.
- We propose a solution at the architectural level that keeps a closer mathematical proximity to the hierarchical block-sparse coding objective in NRS*f*M.
- We outperform numerous classical and learning-based NRS*f*M methods and achieve state-of-the-art performance across multiple benchmarks, showing its effectiveness in handling large amounts of missing data under both weak and strong perspective camera models.

## 3.2 Related Work

**Non-rigid structure from motion.** NRS*f*M concerns the problem of reconstructing 3D shapes from 2D point correspondences from multiple images, *without* the assumption of the 3D shape being rigid. For a shape of $P$ points under orthogonal projection, *atemporal* NRS*f*M is typically framed as factorizing the 2D measurements $\mathbf{W} \in \mathbb{R}^{P\times 2}$ as the product of a 3D shape matrix $\mathbf{S} \in \mathbb{R}^{P\times 3}$ and camera rotation matrix $\mathbf{P} \in \mathbb{R}^{3\times 2}$:

$$\mathbf{W} = \mathbf{S}\mathbf{P} \quad \text{subject to} \;\; \mathbf{P}^\top\mathbf{P} = \mathbf{I}_2 \;, \tag{3.1}$$

where $\mathbf{I}$ is the identity matrix, and the $i$-th row of $\mathbf{W}$ and $\mathbf{S}$ corresponds respectively to the image coordinates $(u_i, v_i)$ and world coordinates $(x_i, y_i, z_i)$ of the $i$-th point.

This factorization problem is ill-posed by nature; in order to resolve the ambiguities in solutions, additional priors are necessary to guarantee the uniqueness of the solution. These priors include the assumption of shape/trajectory matrices being (i) low-rank [8, 19, 39, 50, 101], (ii) being compressible [93, 95, 249], or (iii) lying in a union of subspaces [6, 103, 251]. Although classical NRS*f*M methods incorporating such priors are mathematically well interpreted, they encounter limitations in large-scale datasets (*e.g.* with a large number

of frames). The low-rank assumption becomes infeasible when the data exhibits complex shapes variations. Union-of-subspaces NRS*f*M methods have difficulty clustering shape deformations and estimating affinity matrices effectively. The sparsity prior allows more powerful modeling of shape variations with large number of subspaces but also suffers from sensitivity to noise.

**Perspective projection.** Most factorization-based NRS*f*M research assumes an orthogonal camera model, but this assumption often breaks down for real-world data, where strong perspective effects may exhibit (*e.g.* when objects are sufficiently close to the camera). Modeling perspective projection thus become necessary for more accurate 3D reconstruction. Sturm & Triggs [182] formulate the rigid S*f*M problem under perspective camera as

$$\text{diag}(z_1, \ldots, z_P)[\mathbf{W} \;\; \mathbf{1}_P] = [\mathbf{S} \;\; \mathbf{1}_P]\tilde{\mathbf{P}} \;, \tag{3.2}$$

where $z_i$ is the projective depth of the $i$-th point and the camera matrix $\tilde{\mathbf{P}} \in \mathbb{R}^{4\times 3}$ additionally models translation. This is similar to (3.1) except that the 2D measurements $\mathbf{W}$ is now multiplied by the unknown depth. Since the problem is nonlinear, iterative optimization on the reprojection error is adopted to adjust the projective depth while adding constraints on the rank [36, 77, 144, 182, 198].

This formulation was later extended to solve NRS*f*M. Xiao & Kanade [228] developed a two-step factorization algorithm by recovering the projective depths with subspace constraints and then solving the factorization with orthogonal NRS*f*M. Wang *et al*. [217] proposed to update solutions by gradually increasing the perspectiveness of the camera and recursively refining the projective depth. Hartley & Vidal [73] derived a closed-form linear solution requiring an initial estimation of a multi-focal tensor, which was reported sensitive to noise. Instead of directly solving the problem in (3.2), we simplify the problem by fully utilizing the object-centric constraint, which has been previously used to remove projective depth and translaton in PnP [244]. This results in a bilinear reformulation which is more compatible to factorization-based methods.

**Missing data.** Missing (annotated) point correspondences are inevitable in real-world data due to self/out-of-view occlusions of objects. Handling missing data is a nontrivial task not only because the input data is incomplete, but also because the true object center becomes unknown, making naïve centering of 2D data an ineffective strategy for compensating translation. Previous works employ a strategy of either (i) introducing a visibility mask to

exclude missing points from the objective function [43, 61, 94, 95, 218], (ii) recovering the missing 2D points with matrix completion algorithms prior running NRS*f*M [39, 71, 101, 110], or (iii) treating missing points as unknowns and updating them in an iterative fashion [145]. We follow the first strategy and introduce a visibility mask, but with an additional object-centric constraint [109]. Our key difference to Lee *et al*. [109] is that we (i) derive a bilinear form with dictionaries adaptively normalized to the data; (ii) extend to handle perspective projection which is missing in [109].

**Unsupervised 2D-3D lifting** The problem of unsupervised 2D-3D lifting, which is equivalent to *atemporal* NRS*f*M, is recently approached by training neural networks to predict the third (depth) coordinate of each input 2D point. These networks are trained by minimizing the 2D reprojection error, but as with classical NRS*f*M, this is inherently an ill-posed problem that requires priors or constraints in certain forms. One popular prior is the use of Generative Adversarial Networks (GANs) [59] to enforce realism of 2D reprojections across novel viewpoints [26, 44, 99, 205]. Low-rank priors used in classical NRS*f*M was also explored as a loss function [23]. The recently proposed C3DPO [143] instead enforces self-consistency on the predicted canonicalization of the same 3D shapes. These methods use black-boxed network architectures without geometric interpretability, and thus their robustness to missing data and perspective effects depends solely on the variations within the data, making learning inefficient. In addition, most of these methods [23, 26, 44, 99, 205] are incapable of handling missing data. We mathematically derive a general framework which is applicable for both orthogonal and perspective cameras, robust to large portion of missing data, and interpretable as solving hierarchical block-sparse dictionary coding.

## 3.3 Generalized Bilinear Factorization

A wide range of 2D-3D lifting methods assume a linear model for the 3D shapes to be reconstructed, *i.e*. at canonical coordinates, the vectorization of 3D shape $\mathbf{S}$ in (3.1), denoted as $\mathbf{s} = \text{vec}(\mathbf{S}) \in \mathbb{R}^{3P}$, can be written as $\mathbf{s} = \mathbf{D}\boldsymbol{\varphi}$, where $\mathbf{D} \in \mathbb{R}^{3P\times K}$ is the shape dictionary with $K$ basis and $\boldsymbol{\varphi} \in \mathbb{R}^{K}$ is the code vector. Equivalently, this linear model could be writen as $\mathbf{S} = \mathbf{D}^{\sharp}(\boldsymbol{\varphi} \otimes \mathbf{I}_3)$, where $\mathbf{D}^{\sharp}$ is a $P \times 3K$ reshape of $\mathbf{D}$ and $\otimes$ denotes Kronecker product. Applying the camera extrinsics (*i.e*. rotation $\mathbf{R} \in \mathbb{SO}(3)$ and translation $\mathbf{t} \in \mathbb{R}^3$)

gives the 3D reconstruction in the camera coordinates, written as

$$\mathbf{S}_{\text{cam}} = \mathbf{D}^{\sharp}(\boldsymbol{\varphi} \otimes \mathbf{R}) + \mathbf{1}_P \mathbf{t}^{\top} \ , \tag{3.3}$$

where $\mathbf{1}$ is the one vector. Under unsupervised settings, $\mathbf{D}$, $\boldsymbol{\varphi}$, $\mathbf{R}$, and $\mathbf{t}$ are all unknowns and are usually solved under simplified assumptions, *i.e*. complete input 2D points under orthogonal camera projection. In addition, the translational component $\mathbf{t}$ could be removed if the input 2D points were pre-centered under the implicit object-centric assumption that the origin of the canonical coordinates is placed at the object center. This leads to a bilinear factorization problem which classical NRS*f*M methods [39, 94, 101] employ:

$$\mathbf{W} = \mathbf{D}^{\sharp}\boldsymbol{\Psi}_{\text{xy}} \quad \text{s.t.} \ \ \boldsymbol{\Psi} = \boldsymbol{\varphi} \otimes \mathbf{R} \ \ \text{and} \ \ \boldsymbol{\varphi} \in \mathcal{C} \ , \tag{3.4}$$

where $\boldsymbol{\Psi}_{\text{xy}} \in \mathbb{R}^{3\times 2}$ denotes the first two columns of $\boldsymbol{\Psi}$, $\boldsymbol{\Psi} \in \mathbb{R}^{3K\times 3}$ is the block code (as it is a Kronecker product), and $\mathcal{C}$ denotes the prior constraints applied on the code $\boldsymbol{\varphi}$, *e.g*. low rank [39, 104] or (hierarchical) sparsity [93, 94, 95].

In practical settings, however, the presence of either *missing data* (from occlusions) or *perspective projection* would break this bilinear form. With missing data in the 2D points $\mathbf{W}$, the translation $\mathbf{t}$ does not vanish if $\mathbf{W}$ is simply pre-centered by the average of visible points. Moreover, directly applying Strum & Triggs [182] under perspective projection would introduce unknown projective depth $z_i$, resulting in a non-bilinear form of

$$\text{diag}(z_1, \ldots, z_P)\mathbf{W} = \mathbf{D}^{\sharp}\boldsymbol{\Psi}_{\text{xy}} + \mathbf{1}_P \mathbf{t}_{\text{xy}}^{\top} \ , \tag{3.5}$$

which prevents direct application of NRS*f*M algorithms designed for the orthogonal case. Iterative methods which alternates between solving orthogonal NRS*f*M and optimizing $z_p$ [217] can be applied, but it is cumbersome and prone to poor local minima. On the other hand, similar issue is also encountered in PnP problems, where the projective depths make the problem nonlinear. A simple and effective solution for PnP is to turn the implicit object-centric assumption into an *explicit* constraint, *i.e*.:

$$\mathbf{t} = \frac{1}{P}{\mathbf{S}_{\text{cam}}}^{\top}\mathbf{1}_P \ , \tag{3.6}$$

The introduction of this simple equation allows the projective depth and translation to be

| | $\widetilde{\mathbf{W}}$ | | $\widetilde{\mathbf{D}}$ | | $\widetilde{\mathbf{\Psi}}$ | |
|---|---|---|---|---|---|---|
| | formulation | shape | formulation | shape | formulation | shape |
| orthogonal | $\mathbf{W} - \frac{1}{P}\mathbf{1}_P\mathbf{1}_P^\top\mathbf{M}\mathbf{W}$ | $\mathbb{R}^{P\times 2}$ | $\mathbf{D}^\sharp + \frac{1}{P}\mathbf{1}_P\mathbf{1}_P^\top(\mathbf{I}_P - \mathbf{M})\mathbf{D}^\sharp$ | $\mathbb{R}^{P\times 3K}$ | $\boldsymbol{\varphi}\otimes\mathbf{R}_{\text{xy}}$ | $\mathbb{R}^{3K\times 2}$ |
| perspective | (3.14) | $\mathbb{R}^{2P\times 1}$ | (3.14) | $\mathbb{R}^{2P\times 9K}$ | $\text{vec}(\boldsymbol{\varphi}\otimes\mathbf{R})$ | $\mathbb{R}^{9K\times 1}$ |

Table 3.1: Summary of the formulations of matrices $\widetilde{\mathbf{W}}$, $\widetilde{\mathbf{D}}$, and $\widetilde{\mathbf{\Psi}}$ under orthogonal and perspective cameras.

eliminated for PnP [244]. Similar tricks also apply to NRS*f*M to remove $z_i$ and $\mathbf{t}$ in (3.5), and naturally extends to handle missing data. Hence, we derive a generalized bilinear factorization, which considers both orthogonal and perspective projections, as well as missing data:

$$\mathbf{M}\widetilde{\mathbf{W}} = \mathbf{M}\widetilde{\mathbf{D}}\widetilde{\mathbf{\Psi}} \quad \text{s.t.} \;\; \mathbf{R}\in\mathbb{SO}(3) \;\; \text{and} \;\; \boldsymbol{\varphi}\in\mathcal{C}\;, \tag{3.7}$$

where $\mathbf{M} = \text{diag}(m_1, \cdots, m_P)$ is the input binary diagonal matrix indicating visibility of points; and $\widetilde{\mathbf{W}}$, $\widetilde{\mathbf{D}}$, $\widetilde{\mathbf{\Psi}}$ are the transformed forms of $\mathbf{W}$, $\mathbf{D}$, $\mathbf{\Psi}$, whose formulation under different settings are summarized in Table 3.1. Mathematical derivations are provided in Sec. 3.3.1 for perspective camera and Sec. 3.3.2 for missing data.

### 3.3.1 Perspective Camera

We first consider the case where all points are visible. Let $(x_i', y_i', z_i')$ be the 3D coordinates of the $i$-th point in $\mathbf{SR}$. Since $\mathbf{SR} = \mathbf{D}^\sharp\mathbf{\Psi}$, we can also express $(x_i', y_i', z_i')$ as

$$x_i' = \mathbf{d}_i^\top\boldsymbol{\psi}_x, \quad y_i' = \mathbf{d}_i^\top\boldsymbol{\psi}_y, \quad z_i' = \mathbf{d}_i^\top\boldsymbol{\psi}_z\;, \tag{3.8}$$

where $\mathbf{d}_i^T$ is the $i$-th row of $\mathbf{D}^\sharp$ and $\boldsymbol{\psi}_x$, $\boldsymbol{\psi}_y$, and $\boldsymbol{\psi}_z$ are the three columns in $\mathbf{\Psi}$ corresponding to the $i$-th point. The 3D point $(x_i', y_i', z_i')$, its 3D translation $(t_x, t_y, t_z)$, and its 2D projection $(u_i, v_i)$ on the unit focal plane are related via

$$x_i' + t_x = u_i(z_i' + t_z) \quad \text{and} \quad y_i' + t_y = v_i(z_i' + t_z)\;, \tag{3.9}$$

which states that the product of the depth and 2D coordinates is equivalent to the x-y coordinates in 3D. From the object-centric constraint in (3.6), the translation can be

expressed as the mean of back-projection of the 2D points as

$$t_x = \frac{1}{P}\sum_{i=1}^{P} u_i(z'_i + t_z) \;\text{ and }\; t_y = \frac{1}{P}\sum_{i=1}^{P} v_i(z'_i + t_z). \tag{3.10}$$

Substituting (3.10) and (3.8) into (3.9) and rearranging, we obtain the compact bilinear relationship of $\widetilde{\mathbf{W}} = \widetilde{\mathbf{D}}\widetilde{\mathbf{\Psi}}$, written as

$$\underbrace{\begin{pmatrix} \vdots \\ (u_i - \frac{1}{P}\sum_{j=1}^{P} u_j)t_z \\ (v_i - \frac{1}{P}\sum_{j=1}^{P} v_j)t_z \\ \vdots \end{pmatrix}}_{\widetilde{\mathbf{W}}\text{: normalized 2D projection}} = \underbrace{\begin{pmatrix} \vdots & \vdots & \vdots \\ \mathbf{d}_i^\top & 0 & -u_i\mathbf{d}_i^\top + \frac{1}{P}\sum_{j=1}^{P} u_j\mathbf{d}_j^\top \\ 0 & \mathbf{d}_i^\top & -v_i\mathbf{d}_i^\top + \frac{1}{P}\sum_{j=1}^{P} v_j\mathbf{d}_j^\top \\ \vdots & \vdots & \vdots \end{pmatrix}}_{\widetilde{\mathbf{D}}\text{: normalized dictionary}} \underbrace{\begin{pmatrix} \boldsymbol{\psi}_x \\ \boldsymbol{\psi}_y \\ \boldsymbol{\psi}_z \end{pmatrix}}_{\widetilde{\mathbf{\Psi}}} \tag{3.11}$$

where $\widetilde{\mathbf{W}}$ is computed from $\mathbf{W}$ via zero-centering by the mean and rescaling by $t_z$, the depth of the object center to the camera. Here, $t_z$ is a scalar that normalizes the 2D input and controls the scale of the 3D reconstruction, which is similar to the weak perspective case. $\widetilde{\mathbf{\Psi}}$ becomes a vectorization of $\mathbf{\Psi}$ where the columns are simply concatenated.

**Shape scale $t_z$.** As shown in (3.11), $t_z$ serves the purpose of normalizing the 2D inputs and consequently sets the scale of the shape reconstruction. We note that unlike rigid S*f*M or trajectory reconstruction, where scale constancy between frames (*e.g.* object staticity or temporal smoothness within the same dataset) is important, the purpose of "estimating" scale is fundamentally inapplicable to this work. Under the atemporal NRS*f*M settings, where we deal with non-rigid objects only in the shape space, it is sufficient to obtain a scaled 3D reconstruction, as was in the weak perspective case. This in turn allows us to determine an arbitrary scale $t_z$ for each sample as a preprocessing step to facilitate training. We leave details of determining $t_z$ in practice to Sec. 3.5.

### 3.3.2 Handling Missing Data

**Perspective camera.** When all the 2D keypoint locations might not be fully available, it becomes insufficient to compute $\mathbf{t}$ from (3.10). To resolve this, we propose to replace the occluded keypoints with the ones directly from $\mathbf{S}_{\text{cam}}$, *i.e.*

$$t_x = \frac{1}{P}\sum_{i=1}^{P} \underbrace{m_i u_i (z'_i + t_z)}_{\text{visible points}} + \underbrace{(1-m_i)(x'_i + t_x)}_{\text{occluded points}}, \tag{3.12}$$

where $m_i$ indicates the visibility. Rearranging (3.12), we have

$$t_x = \frac{\sum_{i=1}^{P} m_i u_i (z'_i + t_z) + (1-m_i)x'_i}{\sum_{i=1}^{P} m_i} \tag{3.13}$$

and similarly for $t_y$. Substituting the new expressions of the translational components $t_x$ and $t_y$ into (3.9), we have

$$\underbrace{\mathbf{M}\begin{pmatrix} \vdots \\ (u_i - \frac{\sum_{j=1}^{P} m_j u_j}{\sum_{j=1}^{P} m_j})t_z \\ (v_i - \frac{\sum_{j=1}^{P} m_j v_j}{\sum_{j=1}^{P} m_j})t_z \\ \vdots \end{pmatrix}}_{\widetilde{\mathbf{W}}\text{: normalized 2D projection}} = \underbrace{\mathbf{M}\begin{pmatrix} \vdots & \vdots & \vdots \\ \widetilde{\mathbf{d}}_i^\top & 0 & -u_i\mathbf{d}_i^\top + \frac{\sum_{j=1}^{P} m_j u_j \mathbf{d}_j^\top}{\sum_{j=1}^{P} m_j} \\ 0 & \widetilde{\mathbf{d}}_i^\top & -v_i\mathbf{d}_i^\top + \frac{\sum_{j=1}^{P} m_j v_j \mathbf{d}_j^\top}{\sum_{j=1}^{P} m_j} \\ \vdots & \vdots & \vdots \end{pmatrix}}_{\widetilde{\mathbf{D}}\text{: normalized dictionary}} \begin{pmatrix} \boldsymbol{\psi}_x \\ \boldsymbol{\psi}_y \\ \boldsymbol{\psi}_z \end{pmatrix} \tag{3.14}$$

where $\widetilde{\mathbf{d}}_i = \mathbf{d}_i + \frac{\sum_{j=1}^{P}(1-m_j)\mathbf{d}_j}{\sum_{j=1}^{P} m_j}$.

**Orthogonal camera.** Handling missing data under orthogonal cameras can be derived using as identical strategy as

$$\underbrace{\mathbf{M}\,(\mathbf{W} - \frac{1}{\widetilde{P}}\mathbf{1}_P\mathbf{1}_P^\top\mathbf{M}\mathbf{W})}_{\widetilde{\mathbf{W}}\text{: normalized 2D projection}} = \underbrace{\mathbf{M}\,(\mathbf{D}^\sharp + \frac{1}{\widetilde{P}}\mathbf{1}_P\mathbf{1}_P^\top(\mathbf{I}_P - \mathbf{M})\mathbf{D}^\sharp)}_{\widetilde{\mathbf{D}}\text{: normalized dictionary}}\,\boldsymbol{\Psi} \tag{3.15}$$

where $\widetilde{P}$ is the number of visible points. We leave the derivations to the supplementary

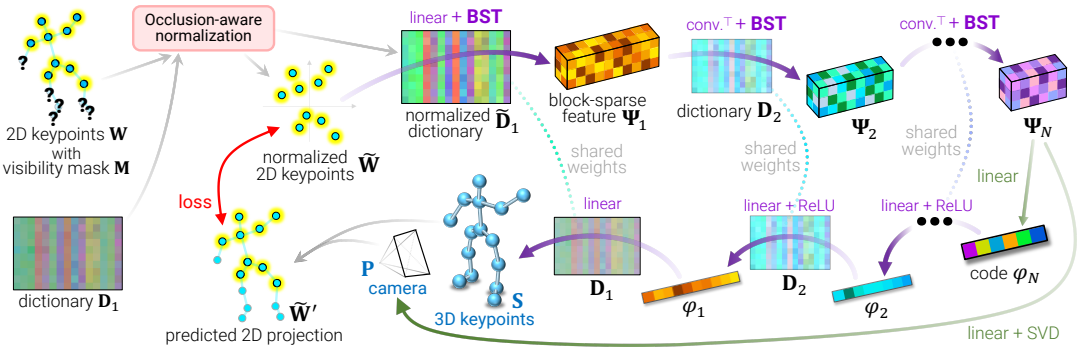



Figure 3.1: **Deep NRS*f*M++** is a general framework for learning hierarchical block-sparse dictionaries that operates under perspective camera models with the ability to handle missing data. The 2D keypoint input $\mathbf{W}$ and shape dictionary $\mathbf{D}_1$ are normalized according to the visibility mask $\mathbf{M}$ and camera model (Table 3.1). The encoder-decoder network, derived from hierarchical block-sparse coding, takes the normalized 2D input $\widetilde{\mathbf{W}}$ and jointly predicts the camera matrix $\mathbf{P}$ and the 3D shape $\mathbf{S}$, further reprojected to 2D as $\widetilde{\mathbf{W}}'$. The training objective is to minimize the difference between $\widetilde{\mathbf{W}}'$ and $\widetilde{\mathbf{W}}$.

material for conciseness. This solution aligns with the common practice employed for data normalization [94, 109, 143]. The difference is that we offset the shape dictionary as well to align the statistics to the shifted 2D inputs.

## 3.4 Deep NRS*f*M++

To solve the generalized bilinear factorization (3.7) given $N$ tuples of $(\mathbf{W}^{(n)}, \mathbf{M}^{(n)})$ as the dataset (with $n$ indexing the samples), two problems remain to address: (i) how to define heuristics $\mathcal{C}$, which is crucial to have an accurate solution, and (ii) how to formulate the optimization strategy. We choose to follow Deep NRS*f*M [94], which instead of using simple handcrafted priors (*e.g.* low rank or sparsity), it imposes hierarchical sparsity constraint $\mathcal{C}_\Theta$ with learnable parameters $\Theta$ (see Sec. 3.4.1). The learning strategy of Deep NRS*f*M is then interpreted as solving a bilevel optimization problem:

$$\min_{\mathbf{D},\Theta} \sum_{n=1}^{N} \min_{\substack{\boldsymbol{\varphi}^{(n)} \in \mathcal{C}_\Theta \\ \mathbf{R}^{(n)} \in \mathrm{SO}(3)}} \|\mathbf{M}^{(n)}\tilde{\mathbf{W}}^{(n)} - \mathbf{M}^{(n)}\tilde{\mathbf{D}}\tilde{\boldsymbol{\Psi}}^{(n)}\|_F, \tag{3.16}$$

where the lower level problems are to solve single frame 2D-3D lifting with given $\mathbf{D}$, $\Theta$; and the upper level problem is to find optimal $\mathbf{D}$,$\Theta$ for the whole dataset.

Descent method is employed to solve this bilevel problem. We first approximate the solver of the lower level problem as a feedforward network, *i.e.* $f(\mathbf{M}, \mathbf{W}^{(n)}; \mathbf{D}, \Theta) \mapsto (\mathbf{R}^{*(n)}_{\mathbf{D},\Theta}, \boldsymbol{\varphi}^{*(n)}_{\mathbf{D},\Theta})$. The architecture of the network (as illustrated in Fig. 3.1) is induced from unrolling one iteration of Iterative Shrinkage and Thresholding Algorithm (ISTA) [16, 74, 169] (see Sec 3.4.2). Then with $\boldsymbol{\Psi}^{*(n)}_{\mathbf{D},\Theta} = \boldsymbol{\varphi}^{*(n)}_{\mathbf{D},\Theta} \otimes \mathbf{R}^{*(n)}_{\mathbf{D},\Theta}$, the original bilevel problem is reduced to a single level unconstrained problem, *i.e.*

$$\min_{\mathbf{D},\Theta} \sum_{n=1}^{N} \|\mathbf{M}^{(n)}\tilde{\mathbf{W}}^{(n)} - \mathbf{M}^{(n)}\tilde{\mathbf{D}}\tilde{\boldsymbol{\Psi}}^{*}_{\mathbf{D},\Theta}\|_F, \tag{3.17}$$

which allows the use of solvers such as gradient descent. Finally, with $\mathbf{D}, \Theta$ learned, $f(\mathbf{M}, \mathbf{W}; \mathbf{D}, \Theta)$ is the 2D-3D lifting network applicable to unseen data.

**Note:** Due to the introduction of multiple levels of dictionaries and codes in Sec. 3.4.1, we will abuse the notation of $\mathbf{D}$, $\boldsymbol{\varphi}$, $\boldsymbol{\Psi}$ by adding subscript 1, *i.e.* $\mathbf{D}_1$, $\boldsymbol{\varphi}_1$, $\boldsymbol{\Psi}_1$ indicating that they are from the first level of the hierarchy.

### 3.4.1 Hierarchical Block-Sparse Coding (HBSC)

Assuming the canonical 3D shapes are compressible via multi-layer sparse coding, the shape code $\boldsymbol{\varphi}_1$ is constrained by $\mathcal{C}_\Theta$ as:

$$\boldsymbol{\varphi}_{l-1} = \mathbf{D}_l\boldsymbol{\varphi}_l, \quad \|\boldsymbol{\varphi}_l\|_1 \leq \lambda_l, \ \boldsymbol{\varphi}_l \geq \mathbf{0}, \quad \forall l \in \{1, \cdots, L\}\ , \tag{3.18}$$

where $\mathbf{D}_l \in \mathbb{R}^{K_{l-1} \times K_l}$ are the hierarchical dictionaries, $l$ is the index of hierarchy level, and $\lambda_l$ is the scalar specifying the amount of sparsity in each level. Thus the learnable parameters for $\mathcal{C}_\Theta$ is $\Theta = \{\cdots, \mathbf{D}_l, \lambda_l, \cdots\}$. Constraints on multi-layer sparsity not only preserves sufficient freedom on shape variation, but it also results in more constrained code recovery.

Multi-layer sparse coding induces a hierarchical block sparsity constraint on the block codes $\boldsymbol{\Psi}_l$ (equal to $\boldsymbol{\varphi}_l \otimes \mathbf{R}_{xy}$ if orthogonal projection and $\boldsymbol{\varphi}_l \otimes \mathbf{R}$ if perspective), which

leads to a relaxation of the lower level problem in (3.16):

$$\begin{aligned}\min_{\{\mathbf{\Psi}_1,\cdots\mathbf{\Psi}_L\}} \|\mathbf{M}\widetilde{\mathbf{W}} - \mathbf{M}\widetilde{\mathbf{D}}_1\widetilde{\mathbf{\Psi}}_1\|_F^2 + \sum_{l=1}^{L}\lambda_l\|\mathbf{\Psi}_l\|_F^{(3\times a)} \\ + \sum_{l=2}^{L}\|\mathbf{\Psi}_{l-1} - (\mathbf{D}_l \otimes \mathbf{I}_3)\mathbf{\Psi}_l\|_F^2\end{aligned} \tag{3.19}$$

where $\|.\|_F^{(3\times a)}$ denotes the sum of the Frobenius norm of each $3 \times a$ block. Here, $a = 2$ for orthogonal projections and $a = 3$ for perspective projections.

**Remark.** Deep NRS*f*M [94] further relaxes the block sparsity in (3.19) to $L_1$ sparsity with a nonnegative constraint to allow for the use of ReLU activations in the network architecture. Such nonnegative constraint is inapplicable to our generalized formulation of $\mathbf{\Psi}$, and we empirically find the use of ReLU activation to degrade performance (Tab. 3.2).

### 3.4.2 HBSC-induced Network Architecture

The 2D-3D lifting network $f$ serves as an approximate solver of the HBSC problem (3.19). The architecture is induced from unrolling one iteration of the Iterative Shrinkage and Thresholding Algorithm (ISTA) [16, 74, 169], one of the classic methods for sparse coding. Unrolling iterative solver as network architecture has been widely practised to insert inductive bias for better generalization [34, 78, 122, 139, 167, 186, 208]. However, the motivation here is different – it is used to insert learnable priors to constrain an unsupervised learning problem. We provide derivations in the following.

**Block soft thresholding.** We review the block-sparse coding problem and consider the single-layer case. To reconstruct an input signal $\mathbf{X}$, we solves:

$$\min_{\mathbf{\Psi}} \|\mathbf{X} - \mathbf{D}\mathbf{\Psi}\|_F^2 + \lambda\|\mathbf{\Psi}\|_F^{(3\times a)} . \tag{3.20}$$

One iteration of ISTA is computed as

$$\mathbf{\Psi}^{(t+1)} = \text{prox}_{\lambda\|\cdot\|_F^{(3\times a)}}(\mathbf{\Psi}^{(t)} - \alpha\mathbf{D}^\top(\mathbf{D}\mathbf{\Psi}^{(t)} - \mathbf{X})), \tag{3.21}$$

where $\text{prox}_{\lambda\|\cdot\|_F^{(3\times a)}}$ is the proximal operator for $L_1$ block sparsity of block size $3 \times a$. Let $\mathbf{\Psi} = [\Pi_1, \Pi_2, \ldots \Pi_K]^\top$, where $\Pi_i \in \mathbb{R}^{3\times a}\ \forall i$. Thus $\text{prox}_{\lambda\|\cdot\|_F^{(3\times a)}}$ is equivalent to applying

*block soft thresholding* (BST) to all $\Pi_i$, defined as

$$\text{BST}^{(3\times a)}(\mathbf{\Psi};\lambda) = \begin{bmatrix} \dots & (1-\frac{\lambda}{\|\Pi_i\|_F})^+\Pi_i^\top & \dots \end{bmatrix}^\top . \tag{3.22}$$

Assuming the block code $\mathbf{\Psi}$ is initialized to $\mathbf{0}$ with the step size $\alpha = 1$, the first iteration of ISTA can be written as

$$\mathbf{\Psi} = \text{BST}^{(3\times a)}(\mathbf{D}^\top \mathbf{X};\lambda) \ . \tag{3.23}$$

We interpret BST as solving for the block-sparse code and incorporate $\text{BST}^{(3\times a)}(\cdot)$ as the nonlinearity in our encoder part of the network, similar to a single-layer ReLU network interpreted as basis pursuit [146]. Our formulation is closer to the true block-sparse coding objective than Deep NRS*f*M, which uses ReLU as the nonlinearity to *relax* the constraint to $L_1$ sparsity with nonnegative constraint.

**Encoder-decoder network.** By unrolling one iteration of block ISTA for each layer, our encoder takes $\widetilde{\mathbf{W}}$ as input and produces the block code for the last layer $\mathbf{\Psi}_L$ as output:

$$\begin{aligned} \mathbf{\Psi}_1 =& \text{BST}^{(3\times a)}\left([\widetilde{\mathbf{D}}_1^\top \widetilde{\mathbf{W}}]_{3K_1\times a};\boldsymbol{\lambda}_1\right), \\ \mathbf{\Psi}_2 =& \text{BST}^{(3\times a)}\left((\mathbf{D}_2\otimes\mathbf{I}_3)^\top\mathbf{\Psi}_1;\boldsymbol{\lambda}_2\right), \\ &\vdots \\ \mathbf{\Psi}_L =& \text{BST}^{(3\times a)}\left((\mathbf{D}_L\otimes\mathbf{I}_3)^\top\mathbf{\Psi}_{L-1};\boldsymbol{\lambda}_L\right), \end{aligned} \tag{3.24}$$

where $\boldsymbol{\lambda}_l$ is the learnable threshold for each $K_l$ block and $[\cdot]_{3K_1\times a}$ is a $3K_1 \times a$ reshape. $(\mathbf{D}_l \otimes \mathbf{I}_3)^T\mathbf{\Psi}_{l-1}$ are implemented by convolution transpose. $\mathbf{\Psi}_L$ are then factorized into $\boldsymbol{\varphi}_L$, $\mathbf{R}$ (constraining to $\mathbb{SO}(3)$ using SVD [94]). The 3D shape $\mathbf{S}$ is recovered from $\boldsymbol{\varphi}_L$ via the decoder as:

$$\begin{aligned} \boldsymbol{\varphi}_{L-1} &= \text{ReLU}(\mathbf{D}_L\boldsymbol{\varphi}_L + \mathbf{b}_L) \ , \\ &\vdots \\ \boldsymbol{\varphi}_1 &= \text{ReLU}(\mathbf{D}_2\boldsymbol{\varphi}_2 + \mathbf{b}_2) \ , \\ \mathbf{S} &= \mathbf{D}_1\boldsymbol{\varphi}_1 \ . \end{aligned} \tag{3.25}$$

**Remark.** Our key technical differences to Deep NRS*f*M are: (i) the first layer of the network is adaptive according to the camera model and keypoint visibility; (ii) replacement of ReLU with BST; (iii) block size of $\mathbf{\Psi}_l$ becomes $3 \times 3$ under the perspective camera

model.

## 3.5 Experiments

**Architectural details.** The most important hyperparameter of Deep NRS*f*M++ is the dictionary size $K_L$ at the last level, which depends on the shape variation that exhibits within the dataset. The rest are set arbitrarily and has less affect on performance. Setting $K_L$ to 8 gives reasonable result across all evaluated tasks. For optimal performance by validating on hold-out validation set, we use 8-10 for articulated objects and 2-4 for rigid ones. The detailed description of the architecture and the analysis of the robustness of $K_L$ are included in the Supp.

**Shape scale $t_z$.** To set the scale $t_z$ in practice, we make use of available 2D information such as bounding boxes; in applications where relative scales between samples in a dataset is available (*e.g.* human skeletons), we can utilize a strategy to estimate and recorrect the scale $t_z$ in an iterative fashion. We use the detected 2D object bounding box to provide an initial estimate of $t_z$ and subsequently update the scale estimation (using the Frobenius norm of $\mathbf{S}$ or the average bone length of a skeleton model, if available). Once we have updated the scale estimation $t_z$, we rerun Deep NRS*f*M++ and update the reconstruction. This **scale correction** procedure allows the 3D reconstruction and scale estimation to improve each other.

**Evaluation metrics.** We employ the following metrics to evaluate the accuracy of 3D reconstruction. **MPJPE**: before calculating the mean per-joint position, we normalize the scale of the prediction to match against ground truth (GT). To account for the ambiguity from weak perspective cameras, we flip the depth value of the prediction if it leads to lower error. **PA-MPJPE**: rigid align the prediction to GT before evaluating MPJPE. **STRESS**: borrowed from Novotny *et al*. [143] is a metric invariant to camera pose and scale. **Normalized 3D error**: 3D error normalized by the scale of GT, used in prior NRS*f*M works [9, 39, 61, 94].

**BST vs ReLU.** To study the effect of replacing ReLU with BST as the nonlinearity, we compare our approach against Deep NRS*f*M [94] on orthogonal projection data with perfect point correspondences on the CMU motion capture dataset [1]. Normalized 3D error is reported per human subject in Table 3.2. Our approach using BST achieves better accuracy

| Subject | CNS | NLO | SPS | Deep NRS*f*M | ReLU | BST |
|---|---|---|---|---|---|---|
| 1 | 0.613 | 1.22 | 1.28 | 0.175 | 0.265 | **0.112** |
| 5 | 0.657 | 1.160 | 1.122 | **0.220** | 0.393 | 0.230 |
| 18 | 0.541 | 0.917 | 0.953 | 0.081 | 0.117 | **0.076** |
| 23 | 0.603 | 0.998 | 0.880 | 0.053 | 0.093 | **0.048** |
| 64 | 0.543 | 1.218 | 1.119 | 0.082 | 0.179 | **0.020** |
| 70 | 0.472 | 0.836 | 1.009 | 0.039 | 0.030 | **0.019** |
| 106 | 0.636 | 1.016 | 0.957 | **0.113** | 0.364 | 0.116 |
| 123 | 0.479 | 1.009 | 0.828 | 0.040 | 0.040 | **0.020** |

Table 3.2: 3D reconstruction error on CMU Motion Capture compared with NRS*f*M methods: CNS [110], NLO [43], SPS [93] and Deep NRS*f*M [94]. "ReLU" and "BST" are Deep NRS*f*M reimplemented with different nonlinearities.

| Method | MPJPE | STRESS |
|---|---|---|
| EM-SfM [194] | 0.107 | 0.061 |
| GbNRSfM [50] | 0.093 | 0.062 |
| Deep NRSfM [94] | 0.076 | 0.063 |
| C3DPO-base [143] | 0.160 | 0.105 |
| C3DPO [143] | 0.067 | 0.040 |
| Deep NRS*f*M++ | **0.062** | **0.037** |

Table 3.3: Test error on Synthetic UP-3D.

compared to Deep NRS*f*M, providing empirical benefits of its closer proximity to solving the true block sparse coding objective.

**Orthogonal projection with missing data.** We evaluate Deep NRS*f*M++ on two benchmarks with high amount of missing data (Table. **??**): (1) Synthetic UP-3D, a large synthetic dataset with dense human keypoints based on the UP-3D dataset [108]. The data was generated by orthographic projections of the SMPL body shape with the visibility computed from a ray tracer. We follow the same settings as C3DPO [143] and evaluate the 3D reconstruction of 79 representative vertices of the SMPL model on the test set. (2) PASCAL3D+ [227] consists of images of 12 rigid object categories with sparse keypoints annotations. To ensure consistency between 2D keypoint and 3D ground truth, we follow C3DPO and use the orthographic projections of the aligned CAD models with the visibility taken from the original 2D annotations. A single model is trained to account for all 12 object categories. We also include results where the 2D keypoints are detected by HRNet [183].

| Method | MPJPE | STRESS |
|---|---|---|
| EM-SfM [194] | 131.0 | 116.8 |
| GbNRSfM [50] | 184.6 | 111.3 |
| Deep NRS*f*M [94] | 51.3 | 44.5 |
| C3DPO-base [143] | 53.5 | 46.8 |
| C3DPO [143] | 36.6 | 31.1 |
| Deep NRS*f*M++ | **34.8** | **27.9** |

Table 3.4: Test error on PASCAL3D+, where the input keypoints are from the ground truth.

| Method | MPJPE | STRESS |
|---|---|---|
| Deep NRS*f*M [94] | 65.3 | 47.7 |
| CMR [88] | 74.4 | 53.7 |
| C3DPO [143] | 57.5 | 41.4 |
| Deep NRS*f*M++ | **53.0** | **36.1** |

Table 3.5: Test error on PASCAL3D+, where the input keypoints are off-the-shelf keypoint detection results from HRNet [183].

Our method achieves over 32% error reduction over Deep NRS*f*M (Table **??**(b)) while comparing favorably against other NRS*f*M methods and deep learning methods such as C3DPO.

**Perspective projection.** We evaluate our approach on two datasets with strong perspective effects: (1) Human 3.6M [83], a large-scale human pose dataset annotated by motion capture systems. We closely follow the commonly used evaluation protocol: we use 5 subjects (1, 5, 6, 7, 8) for training and 2 subjects (9, 11) for testing. (2) ApolloCar3D [179] has 5277 images featuring cars, where each car instance comes with annotated 3D pose. 2D

| Method | KSTA [61] | CNS [110] | Pose-GAN [99] | C3DPO [143] | Chen *et al.* [26] | Deep NRS*f*M++ | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | ortho. | persp. | persp. + scale corr. |
| MPJPE | - | 120.1 | 130.9 | 95.6 | - | 104.2 | 60.5 | **56.6** |
| PA-MPJPE | 123.6 | 79.6 | - | - | 58 | 72.9 | 51.8 | **50.9** |

Table 3.6: Test error on Human 3.6M compared against unsupervised methods. We report our results under both weak perspective and perspective camera models and demonstrate the effectiveness of applying scale corrections (2 iterations).

| | method | w/o missing pts. train | w/o missing pts. test | w/ missing pts. train | w/ missing pts. test |
|---|---|---|---|---|---|
| | CNS [110] | 1.30 | - | - | - |
| | KSTA [61] | 1.58 | - | 1.62 | - |
| Deep NRS*f*M++ | ortho. | 0.596 | 0.591 | 0.679 | 0.681 |
| | persp. | 0.152 | 0.145 | 0.182 | 0.185 |
| | + scale correction | 0.131 | 0.124 | 0.165 | 0.168 |

Table 3.7: Test error on ApolloCar3D, where we report the MPJPE in meters as the evaluation metric.

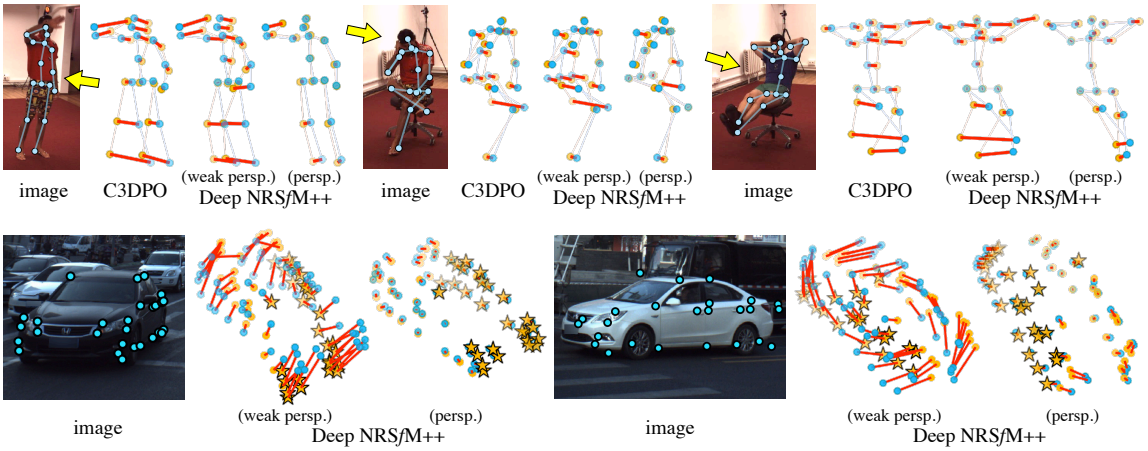



Figure 3.2: Qualitatative comparison. Blue points: GT, yellow points: prediction (stars indicate visible annotations), red lines: prediction error. The two rows are visual results from Human 3.6M and ApolloCar3D datasets respectively.

keypoint annotations are also provided without 3D ground truth. To evaluate our method, we render 2D keypoints by projecting 34 car models according to the 3D pose labels. Visibility of each keypoints are marked according to the original 2D keypoint annotations. To showcase strong perspective effects, we select cars within 30 meters with no less than 25 visible points (out of 58 in total), which gives us 2848 samples for training and 1842 for testing.

We evaluate different variants of our approach. We find that modeling perspective projection (Deep NRS*f*M++ persp.) leads to significant improvement over the orthogonal model (Deep NRS*f*M++ ortho.) and applying scale correction further improves accuracy. Deep NRS*f*M++ shows robustness under different level of noise and occlusion (see Ta-

| Noise ($\sigma$) | Missing pts. (%) | | |
|---|---|---|---|
| | 0 | 30 | 60 |
| 0 | 0.124 | 0.142 | 0.192 |
| 3 | 0.129 | 0.144 | 0.205 |
| 5 | 0.136 | 0.150 | 0.202 |
| 10 | 0.125 | 0.166 | 0.181 |
| 15 | 0.191 | 0.188 | 0.304 |

Table 3.8: Robustness with input under different occlusion rates and noises on ApolloCar3D.

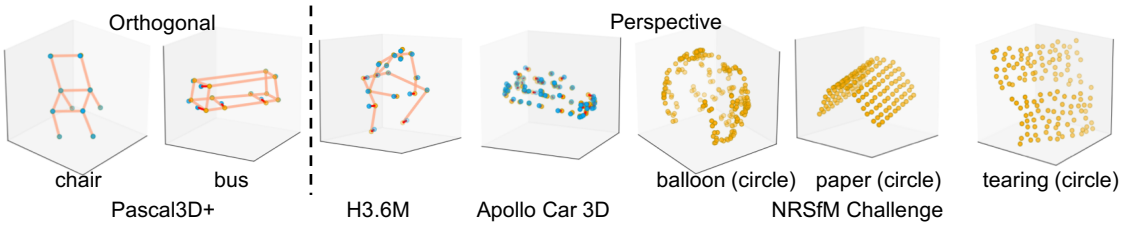



Figure 3.3: Qualitative results across different datasets. Blue points: ground truth; yellow points and orange lines: reconstruction; red lines: difference between ground truth and reconstruction. Best viewed in color and zoomed in.

ble 3.8) and achieves the best result compared to other unsupervised learning method. We outperform the leading GAN-based method [26] by a significant margin when using the same training set (50.9 v.s. 58); in addition, Chen *et al*. [26] reaches our level of performance (50.9 vs 51) only when external training sources and temporal constraints are used. We provide qualitative results in Fig. 3.2, which shows the benefit of Deep NRS*f*M++'s ability to model perspective camera models while also outperforming C3DPO.

**Deforming object in short sequences.** Our method is applicable to NRS*f*M problems with limited number of frames. We experiment on **NRS*f*M Challenge Dataset** [84], which consists of 5 deforming objects captured with six different camera paths. While the leading methods all utilize trajectory information, our atemporal approach still gives reasonable reconstruction as shown in Fig. 5.7. The detailed comparison is in the supp. material.

## 3.6 Conclusion

We propose an atemporal approach applicable to SfC and unsupervised pose estimation. It provides an unified framework to model both orthogonal and perspective camera as well

as missing data. The **limitations** of this work is: (i) the hierarchical sparsity prior is less robust to articulated objects when only limited number of frames is available (see Supp.). (ii) the connection between the proposed learning framework and the true objective of the NRS*f*M problem is still an approximation. Further theoritical investigation is in need to understand the benifit of this approximation or any possible alternatives, but is out of the scope of this paper.

## 3.7 Appendix

### 3.7.1 Derivation for handling missing data under orthogonal camera.

To take possible occlusions into account, we have

$$\mathbf{MW} = \mathbf{M}(\mathbf{SR}_{xy} + \mathbf{1}_P \mathbf{t}_{xy}^\top) \ . \tag{3.26}$$

where $\mathbf{M} = \text{diag}(m_1, \cdots, m_P)$ is a diagonal matrix indicating the visibility for each keypoints, $\mathbf{R}_{xy}$ is the first two columns in rotation matrix $\mathbf{R}$. By the object centric constraint, $\mathbf{t}_{xy}$ has the followin equation:

$$\mathbf{t}_{xy} = \frac{1}{P} \sum_{i=1}^{P} \underbrace{m_i \mathbf{w}_i}_{\text{visible points}} + \underbrace{(1 - m_i)(\mathbf{R}_{xy}^\top \mathbf{s}_i + \mathbf{t}_{xy})}_{\text{occluded points}}, \tag{3.27}$$

where $m_i$ indicates the visibility of the $i$-th keypoint and $\mathbf{s}_i$ denotes the $i$-th 3D point in $\mathbf{S}$. Rearranging (3.27) yields an expression of $\mathbf{t}_{xy}$ with unknowns only from $\mathbf{R}_{xy}$, $\mathbf{S}$:

$$\begin{aligned} \mathbf{t}_{xy} &= \frac{1}{\tilde{P}} \sum_{i=1}^{P} m_i \mathbf{w}_i + (1 - m_i) \mathbf{R}_{xy}^\top \mathbf{s}_i \\ &= \frac{1}{\tilde{P}} [\mathbf{MW} + (\mathbf{I}_P - \mathbf{M}) \mathbf{SR}_{xy}]^\top \mathbf{1}_P \ , \end{aligned} \tag{3.28}$$

where $\tilde{P}$ denotes the number of visible points. Substituting (3.28) into (3.26) and rearranging, we have

$$\mathbf{M}(\mathbf{W} - \mathbf{1}_P \frac{\mathbf{1}_P^\top \mathbf{MW}}{\tilde{P}}) = \mathbf{M}(\mathbf{SR}_{xy} + \mathbf{1}_P \frac{\mathbf{1}_P^\top (\mathbf{I}_P - \mathbf{M}) \mathbf{SR}_{xy}}{\tilde{P}}) \ . \tag{3.29}$$

Since $\mathbf{SR}_{xy} = \mathbf{D}^\sharp(\boldsymbol{\varphi} \otimes \mathbf{R}_{xy}) = \mathbf{D}^\sharp\boldsymbol{\Psi}$, we have

$$\mathbf{M} \underbrace{(\mathbf{W} - \mathbf{1}_P \frac{\mathbf{1}_P^\top \mathbf{M}\mathbf{W}}{\tilde{P}})}_{\tilde{W}\text{: normalized 2D projection}} = \mathbf{M} \underbrace{(\mathbf{D}^\sharp + \mathbf{1}_P \frac{\mathbf{1}_P^\top (\mathbf{I}_P - \mathbf{M})\mathbf{D}_1^\sharp}{\tilde{P}})}_{\tilde{\mathbf{D}}\text{: normalized dictionary}} \boldsymbol{\Psi} \tag{3.30}$$

### 3.7.2 Implementation details

**Dictionary sizes.** The dictionary size in each layer of the block sparse coding is listed in Table 3.9. We tried two strategies to set the dictionary sizes: (i) exponentially decrease the dictionary size, i.e. 512, 256, 128, ..., (ii) linearly decrease, i.e. 125, 115, 104, ... . Both strategies give reasonably good results. However, the hack we need to perform is to pick the size of the first and last layer dictionaries. We find that the size of the first layer would not have a major impact on accuracy as long as it is sufficiently large. The major performance factor is the size of the last layer dictionary $K_L$. In principal, we shall pick $K_l$ by measuring the accuracy on a small holdout validation set with 3D ground truth. We admit that this may become a problem in practise when no validation information is available. Indeed, having a more generalizable solution compared to hyperparameter tuning would be of interest to not only this work but also many other NRSfM or unsupervised 3D pose lifting approaches, which is an important topic to explore in the future. For now, we rely on the robustness of our approach against different bottleneck dimensions, and the rest hyperparameters are set arbitrarily without tuning. Figure 3.4 shows the 3D accuracy with different bottleneck dimensions on CMU-Mocap subject 64. It shows optimal result at dimension around 8 and 10, and has reasonable result at both lower and higher dimensions.

**Training parameters.** We use Adam optimizer to train. Learning rate = 0.0001 with linear decay rate 0.95 per 50k iterations. The total number of iteration is 400k to 1.2 million depending on the data. Batch size is set to 128. Larger or smaller batch sizes all lead to similar result.

**Initial scale for input normalization.** For weak perspective and strong perspective data, we need to estimate the scale so as to properly normalize the size of the 2D input shape. For Pascal3D+ and H3.6M datasets, we use the maximum length of the 2D bounding box edges, i.e. $t_z = 1/\max(\text{bbox\_height}, \text{bbox\_width})$. For Apollo 3D Car dataset, we choose

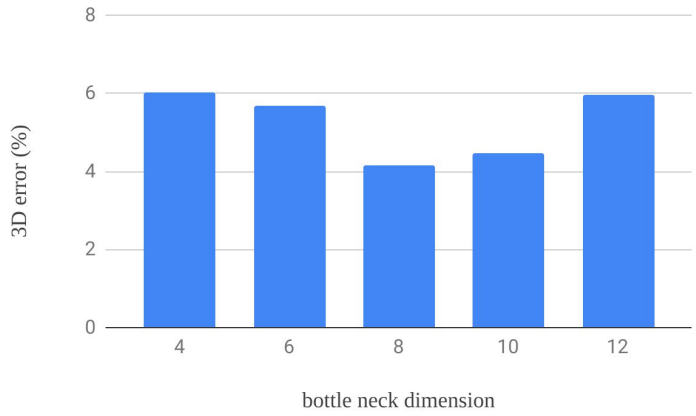



Figure 3.4: 3D reconstruction accuracy with different bottleneck dimensions on CMU-Mocap subject 64.

the minimum length of the bounding box, i.e. $t_z = 1/\min(\text{bbox\_height}, \text{bbox\_width})$ by assuming that the height of each car is identical.

### 3.7.3 Additional empirical analysis

**H3.6M.** We add the test result using detected 2D keypoint as input in Table 3.10. Deep NRS*f*M ++ achieves state-of-the-art result compared to other unsupervised methods.

**ApolloCar3D.** Figure 3.5 shows additional analysis of Deep NRS*f*M ++ on the training set. Our method achieves $< 25$cm error for over 80% testing samples with occlusions, while the compared baseline method, namely Consensus NRS*f*M [110] fails to produce meaningful reconstruction using perfect point correspondences. Average errors at different distances, rotation angles and occlusion rates are also reported. Overall, our method does not have a strong bias against a particular distance or occlusion rate. It does show larger error at $60^\circ$ azimuth, most likely due to the data distribution, where most cars are in either front ($\approx 0^\circ$) or back ($\approx 180^\circ$) view.

**NRSfM Challenge Dataset.** The leading methods on the leaderboard all utilize temporal constraints, while our **atemporal** approach solves the problem only in the shape space. Therefore it is natural for our method to fall behind those methods, but nonetheless, we achieve reasonable results compared to other approaches also only use shape constraints.

| dataset | dictionary sizes |
|---|---|
| CMU-Mocap | 512, 256, 128, 64, 32, 16, 8 |
| UP3D | 512, 256, 128, 64, 32, 16, 8 |
| Pascal3D+ | 256, 128, 64, 32, 16, 8, 4, 2 |
| H3.6M | 125, 115, 104, 94, 83, 73, 62, 52, 41, 31, 20, 10 |
| NRSfM Challenge | 125, 115, 104, 94, 83, 73, 62, 52, 41, 31, 20, 10 |
| Apollo | 128, 100, 64, 50, 32, 16, 8, 4 |

Table 3.9: Dictionary sizes used in each reported experiment.

In Table 3.11 and Table 3.12, we compare to a selection of classical methods in literature. For a complete comparison, please refer to the official leaderboard of the challenge [84].

### 3.7.4 Additional discussion

One of the benefit of solving NRS*f*M by training a neural network is that, in addition to 2D reconstruction loss, we can easily employ other loss functions to further constrain the problem. In summary, our preliminary study finds that: (i) adding the canonicalization loss [143] does not noticeably improve result. (ii) adding Lasso regularization on $\varphi_1$ gives marginally better result in some datasets. (iii) adding symmetry constraint on the skeleton bone length helps to improve robustness against network initialization, but does not lead to noticeable better accuracy.

| Method | MV/T | E3D | MPJPE | PA-MPJPE |
|---|---|---|---|---|
| Martinez *et al*. [130] | - | - | 62.9 | 52.1 |
| Zhao *et al*. [243] | - | - | 57.6 | - |
| 3DInterpreter [223] | | ✓ | - | 98.4 |
| AIGN [48] | | ✓ | - | 97.2 |
| Tome *et al*. [193] | ✓ | ✓ | 88.4 | - |
| RepNet [205] | | ✓ | 89.9 | 65.1 |
| Drover *et al*. [44] | | ✓ | - | 64.6 |
| Pose-GAN [99] | | | 173.2 | - |
| C3DPO [143] | | | 145.0 | - |
| Wang *et al*. [209] | | | 83.0 | 57.5 |
| Chen *et al*. [26] | ✓ | | - | 68 |
| Ours(persp proj) | | | 68.9 | 59.4 |
| + scale corr itr1 | | | 67.3 | 59.2 |
| + scale corr itr2 | | | **67.0** | **58.7** |

Table 3.10: Result on **H3.6M** dataset with detected 2D kepoint input. In our result, we use detected points from cascaded pyramid network (CPN [31]) which is finetuned on H3.6M training set (excluding S9 and S11) by [155].

| | temporal | Articulated | Balloon | Paper | Stretch | Tearing |
|---|---|---|---|---|---|---|
| CSF2 | ✓ | 35.51 | 19.01 | 33.95 | 23.22 | 18.77 |
| KSTA | ✓ | 42.11 | 18.45 | 32.18 | 22.88 | 17.59 |
| PTA | ✓ | 36.71 | 28.88 | 41.72 | 30.45 | 23.14 |
| Bundle | ✓ | 64.48 | 36.40 | 41.64 | 36.64 | 28.73 |
| SoftInext | x | 61.43 | 36.75 | 47.41 | 45.56 | 37.87 |
| Compressible | x | 72.77 | 52.53 | 62.44 | 57.45 | 54.71 |
| MDH | x | 88.66 | 58.27 | 66.98 | 66.27 | 56.67 |
| Ours | x | 47.80 | 30.16 | 44.33 | 36.59 | 30.74 |

Table 3.11: Comparison on **NRSfM Challenge Dataset** with **orthogonal** projection and no missing data.

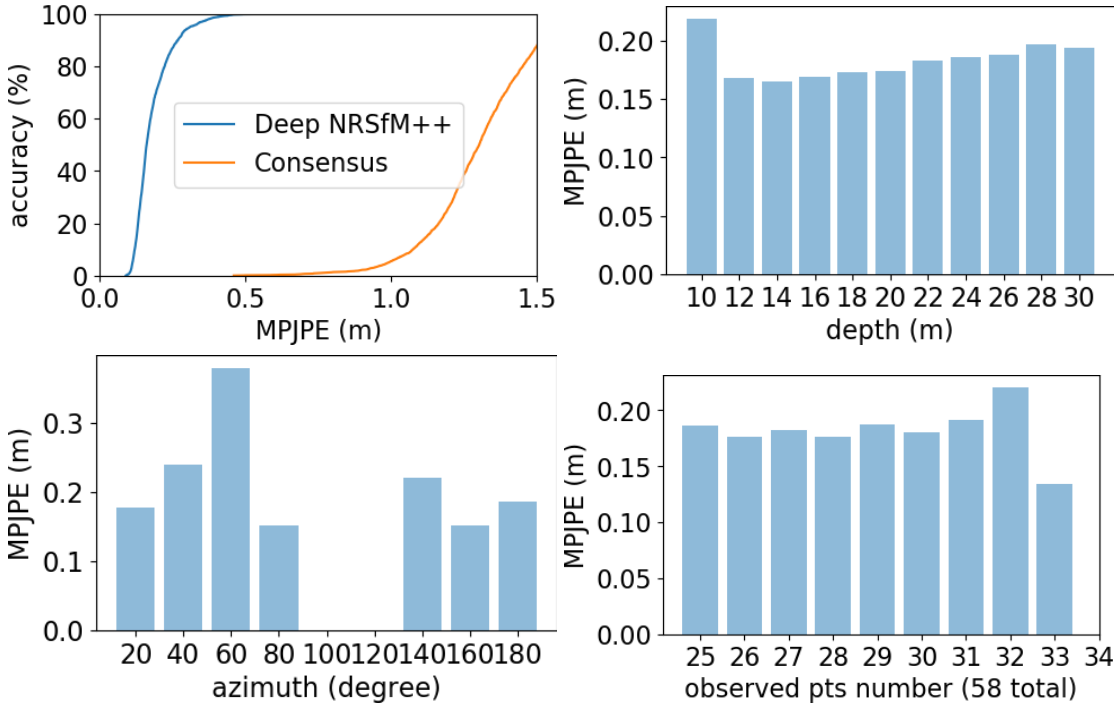



Figure 3.5: Additional result on **Apollo 3D Car** dataset. Top-left: percentage of sucess at different error thresholds. Rest: average error at different distances to the camera (top-right), azimuth angles of the car (bottom-left) and number of observed keypoints (bottom-right).

| | temporal | Articulated | Balloon | Paper | Stretch | Tearing |
|---|---|---|---|---|---|---|
| CSF2 | ✓ | 49.75 | 29.14 | 38.07 | 33.07 | 26.09 |
| KSTA | ✓ | 44.49 | 27.94 | 36.06 | 29.23 | 23.01 |
| PTA | ✓ | 58.08 | 37.20 | 47.42 | 41.40 | 30.96 |
| Bundle | ✓ | 58.95 | 37.07 | 40.88 | 38.16 | 27.40 |
| SoftInext | x | 69.11 | 41.76 | 53.91 | 48.98 | 45.93 |
| Compressible | x | 88.00 | 55.21 | 67.44 | 63.11 | 53.93 |
| MDH | x | 91.55 | 58.00 | 66.54 | 62.49 | 53.93 |
| Ours | x | 65.93 | 31.91 | 41.03 | 47.61 | 38.98 |

Table 3.12: Comparison on **NRSfM Challenge Dataset** with **Perspective** projection and no missing data.

# Chapter 4

# Distill Knowledge from NRSfM for Weakly Supervised 3D Pose Learning

## 4.1 Introduction

Learning to estimate 3D pose from images is bottlenecked by the availability of abundant 3D annotated data. Weakly supervised methods that reduce the amount of required annotation is of high practical value. Prior works approach this problem by supplementing their training set with: (i) extra 2D annotated data [250]; (ii) aligning 3D models to 2D annotations [178, 193, 223]; (iii) exploiting geometric cues from multi-view footage [165, 166, 197]; or (iv) utilizing adversarial framework to impose a prior on the 3D structure [45]. These methods, however, are either restricted to laboratory settings or still requires a 3D training set – which limits the type of target objects they can work with. This paper addresses a more general setting – we utilize image datasets with solely 2D landmark annotations (i.e. no 3D supervision). This allows our method to be applied to a wider scope of objects, not limited by the availability of 3D models, kinematic priors, or sequential/multi-view footage.

Our work is made possible by some recent advances in Non-Rigid Structure from Motion (NRSfM). NRSfM methods reconstruct 3D shapes and camera positions from multiple 2D projections of articulated 3D points. These points do not have to belong to the same object, but can be from multiple instances of the same object category, which naturally applies to our problem. Prior NRSfM methods are restricted by the number

of frames and the type of shape variability they can handle, which limits their usage to many real world problems. Kong and Lucey [94] recently proposed a neural network architecture (Deep-NRSfM) interpreted as solving a multi-layer block sparse dictionary learning problem, and can handle problems of unprecedented scale and shape complexity. Our *modified* version of Deep-NRSfM achieves state-of-the-arts accuracy on H3.6M [83] dataset, outperforming other NRSfM methods by a significant margin.

Despite this progress, NRSfM still has difficulty in predicting correct depth for shapes with strong ambiguity in terms of 2D projection, e.g. identifying if a leg is stretching towards/away from the camera, even though these are distinguishable with texture features. Therefore, directly using the depth output from NRSfM as labels to train a pose estimation network is affected by those errors. Instead of this hard assignment of training labels, we propose a softer approach – we want to penalize less when there's high ambiguity in 2D projection, so as to leave room for the pose estimation network to correct errors made by NRSfM through associating image features (see Fig. 7.1).

To design our learning objective, we review the dictionary learning problem used to solve NRSfM. Assuming the camera matrix fixed, a depth hypothesis defines a subspace of codes – any codes in this subspace is to have the same depth reconstruction as the hypothesis, but have different cost (2D reprojection error + regularizer). A natural way to characterize the quality of a depth hypothesis is by the minimum cost of codes in its subspace. However, directly using this as a learning objective leads to solving a constrained optimization problem numerically per SGD iteration, which is computationally intractable. Instead, we derive a convex upper bound by evaluating the cost at the projection of the NRSfM solution on the subspace. Experiments show that pose network trained by this loss noticeably reduces error on the training set compared to our already strong NRSfM baseline, and consequently leads to lower validation error as a weakly supervised learning task.

Another benefit of the proposed knowledge distilling loss is that, it poses no restriction on the architecture of the student pose estimation network, as long as it outputs the depth value for the landmarks. This is not the case for some of the prior works [48, 223], where the pose estimation network has to output the coefficients associated to some external shape dictionary.

In conclusion, contributions of this paper are:

- We propose a weakly supervised pose estimation method using solely 2D landmark

annotations. We do not use any 3D labels, multi-view footage, or target specific shape prior. In spite of using weaker supervision, we achieve the best results compared to other weakly supervised methods.

- We establish a strong NRSfM baseline modified from Deep-NRSfM [94], which outperforms current published state-of-the-art NRSfM methods on H3.6M dataset.
- We propose a new knowledge distilling algorithm applicable to NRSfM methods based on dictionary learning. We demonstrate that our learned network gets significantly lower error on the training set compared to its NRSfM teacher.

## 4.2 Related Works

**Non-rigid structure from motion** NRSfM is a classical ill-posed problem since the 3D shapes can vary between images, resulting in more variables than equations. To alleviate the ill-posedness, various constraints are exploited including 1) temporal smoothness [11, 60, 102, 103], 2) fixed articulation [160] and more commonly used 3) shape priors. The first statistical shape prior—non-rigid objects can be modeled by a local subspace in low rank—is first proposed by Bregler *et al*. [19] and later developed by Dai *et al*. [39]. Following this direction, increasing works are reported to model more complex objects while still maintaining a well-conditioned system. Among them, representatives are union-of-subspaces [7, 251], and block-sparsity [93, 96]. Of particular interest to this paper is the most recent work [94] that introduces deep neural network to accurately solving large scale NRSfM problem. Even though great success, majority NRSfM algorithms rely heavily on 2D annotation-based priors. However, as pointed in the introduction, much broader information are embedded under image itself, under pixel values. In this paper, we impose a novel image prior such that NRSfM is no longer trapped at 2D coordinates of landmarks but also learn from origin images.

**Weakly supervised 3D pose learning** Most 3D pose estimation methods [25, 38, 131, 152, 153, 154, 184, 232, 237, 250] are fully supervised. One bottleneck for the supervised methods is that data coming from multi-view motion capture systems [83, 86] includes limited number of human subject, and has simple backgrounds. This would affect the generalization ability of a trained model. Weakly supervised methods aim to alleviate this

problem by limiting the requirement for labeled data. They can be loosely categorized as: using synthetic datasets [29, 203] to increase the training set size. These methods face the problem of generalizing to new motions and environments that are different from the simulated data; On the other hand, given the existing large-scale image datasets with 2D annotation, Zhou *et al*. [250] train their model with 2D labeled images together with motion capture data. To further reduce dependency on paired 3D annotation, 3D interpreter network [223], multi-modal model [193] and generative adversarial networks [48, 206] are trained on external 3D data; multi-view footage is also used to enforce geometric constraints [166, 197]; However, these methods still require a large enough 3D training set to properly initialize and constraint their learning process.

Recently, Rhodin *et al*. [165] propose a method based on geometric-aware representation learning, which requires only a small amount of annotation. Its performance however is limited, which restricts its practical usage. A concurrent work of Drover *et al*. [45] propose to use adversarial framework to impose a prior on the 3D structure, learned solely from 2D projections. Yet they still utilize the ground-truth 3D poses to generate a large number of synthetic 2D poses for training, which augments the original 1.5M 2D poses in Human3.6M by almost 10 times.

## 4.3 Non-rigid Structure from Motion

Under weak perspective camera assumption, 2D projection $\mathbf{W} \in \mathbb{R}^{P\times 2}$ is the product of 3D shape $\mathbf{S} \in \mathbb{R}^{P\times 3}$ and camera matrix $\mathbf{M} \in \mathbb{R}^{3\times 2}$:

$$\mathbf{W} = \mathbf{S}\mathbf{M}, \quad \mathbf{W} = \begin{bmatrix} \vdots & \vdots \\ u_p & v_p \\ \vdots & \vdots \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} \vdots & \vdots & \vdots \\ x_p & y_p & z_p \\ \vdots & \vdots & \vdots \end{bmatrix}, \tag{4.1}$$

where $(u_p, v_p)$ and $(x_p, y_p, z_p)$ are the image and world coordinate of $p$-th point, and $\mathbf{M}$ is required to be orthonormal. The goal of NRSfM is to recover 3D shape $\mathbf{S}$ and camera matrix $\mathbf{M}$ given the observed 2D projections $\mathbf{W}$. This is an inherent ill-posed problem. Finding a unique solution requires sufficient regularization and prior knowledge.

One type of NRSfM methods approach the problem through dictionary learning. Denote $\mathbf{s} \in \mathbb{R}^{3P}$ is the vectorization of $\mathbf{S}$, it satisfies: $\mathbf{s} = \mathbf{D}\boldsymbol{\varphi}$, where $\mathbf{D} \in \mathbb{R}^{3P\times K}$ is a dictionary

with $K$ bases; and $\boldsymbol{\varphi} \in \mathbb{R}^K$ is a code vector. Given multiple observation of 2D projections $\mathbf{W}^{(i)}$ from an articulated object deforming over time, or different objects of the same category, these methods can be loosely interpreted as minimizing the following objective:

$$\min_{\mathbf{D},\{\boldsymbol{\varphi}^{(i)}\},\{\mathbf{M}^{(i)}\}} \sum_i \|[\mathbf{D}\boldsymbol{\varphi}^{(i)}]_{P\times 3}\mathbf{M}^{(i)} - \mathbf{W}^{(i)}\| + h(\boldsymbol{\varphi}^{(i)}) \tag{4.2}$$

where operator $[\,]_{P\times 3}$ is defined as reshaping the vecorized 3D shape into matrix form with dimension $P \times 3$; $h(\boldsymbol{\varphi})$ is a regularizer introduced to improve uniqueness of solution, e.g. low rank [39], sparsity [93], *etc*.

Our knowledge distilling method (see Section 4.4) is designed for this general type of NRSfM method, and in principal, it is agnostic to the type of regularizor they use, as long as the dictionary is overcomplete.

**Deep NRSfM** Kong and Lucey[94] propose a prior assumption that 3D shapes are compressible via multi-layer sparse coding:

$$\begin{aligned} \mathbf{s} &= \mathbf{D}_1\boldsymbol{\varphi}_1, \quad \|\boldsymbol{\varphi}_1\|_1 \le \lambda_1, \quad \boldsymbol{\varphi}_1 \ge 0, \\ \boldsymbol{\varphi}_1 &= \mathbf{D}_2\boldsymbol{\varphi}_2, \quad \|\boldsymbol{\varphi}_2\|_1 \le \lambda_2, \quad \boldsymbol{\varphi}_2 \ge 0, \\ &\vdots, \quad \vdots \\ \boldsymbol{\varphi}_{n-1} &= \mathbf{D}_n\boldsymbol{\varphi}_n, \quad \|\boldsymbol{\varphi}_n\|_1 \le \lambda_n, \quad \boldsymbol{\varphi}_n \ge 0, \end{aligned} \tag{4.3}$$

where $\mathbf{D}_i$ are hierarchical dictionaries, and code vectors $\boldsymbol{\varphi}_i \in \mathbb{R}^{K_i}$ are constrained to be sparse and non-negative. Compared to single level sparse coding, codes in multi-layer sparse coding not only minimizes the reconstruction error at their individual levels, but is also regularized by the codes from other levels. This helps to impose more constraints on code recovery while maintaining similar shape expressibility versus single level sparse coding with the same dictionary size.

To recover sparse codes, one of the classical method to use is Iterative Shrinkage and Thresholding Algorithm (ISTA) [17, 40, 170]. Papyan *et al*. [146] find that feed-forward neural netorks can be interpreted as approximating one iteration of inferencing sparse codes by ISTA, and the dictionaries $\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_n$ serves as the neural network weights. Based on this insight, Chen *et al*. derived a novel neural network architecture which approximates the solution of sparse codes $\boldsymbol{\varphi}_1$ and camera matrix $\mathbf{M}$. In this paper, we made

significant modification to their original architecture, which we find important to get good result in experiment. Limited by space, we put description about our version of camera matrix estimation network $q_{\mathbf{M}}(\mathbf{W}) : \mathbb{R}^{P\times 2} \mapsto \mathbb{R}^{3\times 2}$, and sparse code estimation network $q_{\boldsymbol{\varphi}}(\mathbf{W}, \mathbf{M}) : \mathbb{R}^{P\times 2} \times \mathbb{R}^{3\times 2} \mapsto \mathbb{R}^{K_1}$ in the supplementary material.

With the feed-forward code/camera estimation networks parameterized by the dictionaries, we can now learn the dictionaries through minimizing reprojection error of all samples in the dataset. Denote $\tilde{\boldsymbol{\varphi}}_1^{(i)}$, $\tilde{\mathbf{M}}^{(i)}$ to be the output of networks $q_{\boldsymbol{\varphi}}$, $q_{\mathbf{M}}$ given $i$th 2D projection $\mathbf{W}^{(i)}$, the loss function is:

$$\min_{\mathbf{D}_1,\mathbf{D}_2,\ldots,\mathbf{D}_n} \sum_i \|[\mathbf{D}_1\tilde{\boldsymbol{\varphi}}_1^{(i)}]_{P\times 3}\tilde{\mathbf{M}}^{(i)} - \mathbf{W}^{(i)}\|_2 + \lambda\|\tilde{\boldsymbol{\varphi}}_1\|_1. \tag{4.4}$$

In this loss function, in addition to reprojection error, we add sparisity penalty using a small weighting, which we find helpful to improve results.

## 4.4 Distilling Knowledge from NRSfM

**Problem setup**: Given an image dataset paired with annotated 2D locations of landmarks on target objects: $\{(\mathbf{I}^{(i)}, \mathbf{W}^{(i)})\}$, we want to train a 3D pose estimation network able to predict 3D landmark positions from image input. The main difficulty of this task is how to learn to predict depth of landmarks without any depth supervision. Our cue is from dictionary learning-based NRSfM method (Deep-NRSfM in our experiment), which gives us a 3D shape dictionary $\mathbf{D}$, and recovered camera matrices $\mathbf{M}^{(i)}$ and codes $\boldsymbol{\varphi}_{\text{nrsfm}}^{(i)}$.

With the dictionary, camera matrices and codes from NRSfM, depth in the image coordinate can be computed by simply rotating the 3D shape reconstruction $\mathbf{D}\boldsymbol{\varphi}_{\text{nrsfm}}^{(i)}$. Given this, a simple baseline for this task would be: we use the depth reconstruction as labels to train the 3D pose estimation network. However, as shown in Fig. 7.1, we find that NRSfM tends to make wrong estimation due to strong ambiguity in 2D projections. Using those as hard target for regression would bottleneck the accuracy of learned pose estimation network. We propose a better approach - we want to establish a direct relation between depth prediction and the cost function (Eq. 4.2) we used in NRSfM, which is the better metric to evaluate the quality of predicted 3D shapes. In this way, we can avoid confusing our student network with wrong labels, and allow them to implicitly associate image features to disambiguate difficult poses for NRSfM. This intuition is inline with other

geometric self-supervised learning, e.g. self-supervised depth estimation [58, 215, 247], in which photometric loss is used to train a depth estimation network.

**Outline**: The core problem is how to design a loss function which properly evaluates the quality of a depth hypothesis produced by the pose estimator. To derive our loss function, We first show that a depth hypothesis associates with a subspace of codes (see Section 4.4.1). We then advocate that the loss should be the minimum cost value of codes in the subspace (see Section 4.4.2). Finally, we derive a convex upper bound for the loss, which is computationally trackable for SGD training (see Section 4.4.3). A 2D illustration is given in Fig. 4.2 to help decipher the text.

### 4.4.1 Depth hypothesis defines a subspace of codes

From NRSfM, we get the dictionary $\mathbf{D}$, and per example camera matrix $\mathbf{M}^{(i)}$. We find that the camera matrices from our modified Deep-NRSfM are accurate, thus we treat them as oracle and fixed in our learning algorithm. With this, we can simplify our notation by absorbing camera matrix into dictionary through rotation. Rotation matrix $\mathbf{R}^{(i)} \in \mathbb{R}^{3\times3}$ is formed from camera matrix by:

$$\mathbf{R}^{(i)} = [\mathbf{m}_1^{(i)}, \mathbf{m}_2^{(i)}, \mathbf{m}_1^{(i)} \times \mathbf{m}_2^{(i)}], \tag{4.5}$$

where $\mathbf{m}_1^{(i)}$, $\mathbf{m}_2^{(i)}$ are columns of camera matrix $\mathbf{M}^{(i)}$. Then the dictionary is rotated by multiplying every 3D coordinates inside $\mathbf{D}$ with $\mathbf{R}^{(i)}$:

$$\mathbf{B}^{(i)} = \begin{bmatrix} [\mathbf{d}_x^1, \mathbf{d}_y^1, \mathbf{d}_z^1]\mathbf{R}^{(i)} & \dots & [\mathbf{d}_x^P, \mathbf{d}_y^P, \mathbf{d}_z^P]\mathbf{R}^{(i)} \end{bmatrix}^T \tag{4.6}$$

We further split $\mathbf{B}^{(i)}$ into two matrices – one matrix takes all the $x$, $y$ coordinate elements of $\mathbf{B}^{(i)}$, while the other takes all the rest $z$ coordinate elements.

$$\begin{aligned} \mathbf{B}_{xy}^{(i)} &= \begin{bmatrix} \mathbf{b}_x^{1(i)} & \mathbf{b}_y^{1(i)} & \dots & \mathbf{b}_x^{P(i)} & \mathbf{b}_y^{P(i)} \end{bmatrix}^T, \\ \mathbf{B}_{z}^{(i)} &= \begin{bmatrix} \mathbf{b}_z^{1(i)} & \dots & \mathbf{b}_z^{P(i)} \end{bmatrix}^T, \end{aligned} \tag{4.7}$$

With this, $\mathbf{B}_{xy}^{(i)}\boldsymbol{\varphi}^{(i)}$ computes 2D projection of shape reconstructed by code $\boldsymbol{\varphi}^{(i)}$; and $\mathbf{B}_{z}^{(i)}\boldsymbol{\varphi}^{(i)}$ is reconstructed depth in the image coordinate.

For a depth hypothesis $\mathbf{z}' = f_z(\mathbf{I}^{(i)}; \boldsymbol{\theta})$ produced by the pose estimation network, codes giving depth reconstruction equal to $\mathbf{z}'$ forms a subspace:

$$\mathcal{S}^{(i)}(\mathbf{z}') = \{\boldsymbol{\varphi} : \mathbf{B}_z^{(i)}\boldsymbol{\varphi} = \mathbf{z}'\}. \tag{4.8}$$

The subspace is not empty assuming that dictionary is overcomplete. In Fig. 4.2, the subspaces are visualized as orange lines in 2D.

### 4.4.2 Loss = minimum cost on subspace

The quality of a depth hypothesis $\mathbf{z}'$ could be represented by the best code inside its subspace. As in NRSfM, the quality of a code is measured by the cost function = reprojection error + some regularizer, i.e.:

$$\mathcal{C}^{(i)}(\boldsymbol{\varphi}) = \|\mathbf{B}_{xy}^{(i)}\boldsymbol{\varphi} - \mathbf{w}^{(i)}\| + h(\boldsymbol{\varphi}), \tag{4.9}$$

where $\mathbf{w}^{(i)}$ is the vectorization of $\mathbf{W}^{(i)}$. To keep formulation general, we don't specify the type of norm and regularizer here. Thereby we have the following definition of quality function for $\mathbf{z}'$, which we use as the loss function for knowledge distilling:

$$\mathcal{L}^{(i)}(\mathbf{z}') = \min_{\boldsymbol{\varphi} \in \mathcal{S}^{(i)}(\mathbf{z}')} \mathcal{C}^{(i)}(\boldsymbol{\varphi}). \tag{4.10}$$

This computes the minimum cost value of codes inside the subspace defined by the depth hypothesis $\mathbf{z}'$.

To evaluate this loss function, we need to first solve for the minima $\boldsymbol{\varphi}^*$ of the constrained convex optimization problem in Eq. 4.10 (red cross in Fig. 4.2). Suppose we can express $\boldsymbol{\varphi}^*$ as a differentiable function of $\mathbf{z}'$, i.e. $\boldsymbol{\varphi}^* = q^{(i)}(\mathbf{z}')$, Eq. 4.10 becomes:

$$\mathcal{L}^{(i)}(\mathbf{z}') = \|\mathbf{B}_{xy}^{(i)} q^{(i)}(\mathbf{z}') - \mathbf{w}^{(i)}\| + h(q^{(i)}(\mathbf{z}')). \tag{4.11}$$

This loss is explicitly a function of $\mathbf{z}'$, and thus allows the gradients to be propagated to the pose estimation network.

As a side note, suppose the pose network has unlimited capacity, in other words, able to overfit any depth values, then the end result of minimizing this loss function would be a network predicting the same depth as the NRSfM algorithm (illustrated in Fig. 4.2(a)). We argue that this would not be the case in practice, since convolution networks constrained by

their structure, is equivalent to have a deep image prior [199] imposed on their output. This image prior provides extra constraint to disambiguate confusing 2D projections, thus is the key source for our improvement over the NRSfM teacher.

### 4.4.3 Convex upper bound of Eq. 4.11

Using Eq. 4.11 requires to form the (sub)differentiable function $q^{(i)}(\mathbf{z}')$ which produces the solution to the constrained optimization problem in Eq. 4.10. However, solving this constrained optimization problem requires iterative numerical method due to the existence of regularizer. As a result, it's computationally intractable to solve it exactly per SGD iteration during training. Therefore we derive an approximate solution as follow:

Suppose $\boldsymbol{\varphi}^{(i)}_{\text{nrsfm}}$ is the solution we get from NRSfM, and it approximates the minima of the optimization problem in Eq. 4.10 without the subspace constraint, then an approximate solution for the constrained problem could be the projection of $\boldsymbol{\varphi}^{(i)}_{\text{nrsfm}}$ onto the subspace $\mathcal{S}^{(i)}(\mathbf{z}')$:

$$\tilde{\boldsymbol{\varphi}}^{(i)}(\mathbf{z}') = \arg \min_{\boldsymbol{\varphi} \in \mathcal{S}^{(i)}(\mathbf{z}')} \frac{1}{2} \|\boldsymbol{\varphi} - \boldsymbol{\varphi}^{(i)}_{\text{nrsfm}}\|_2^2 \tag{4.12}$$

The closed form solution to Eq. 4.12 is:

$$\tilde{\boldsymbol{\varphi}}^{(i)}(\mathbf{z}') = \boldsymbol{\varphi}^{(i)}_{\text{nrsfm}} + (\mathbf{B}^{(i)}_z)^{\dagger}(\mathbf{z}' - \mathbf{B}^{(i)}_z \boldsymbol{\varphi}^{(i)}_{\text{nrsfm}}), \tag{4.13}$$

where $(\mathbf{B}^{(i)}_z)^{\dagger} = \mathbf{B}^{(i)^T}_z (\mathbf{B}^{(i)}_z \mathbf{B}^{(i)^T}_z)^{-1}$ is the right inverse of $\mathbf{B}^{(i)}_z$. Eq. 4.13 is implemented as a differentiable operator thanks to modern deep learning library.

Substitute the exact solution $q^{(i)}(\mathbf{z}')$ in Eq. 4.11 by the approximate solution $\tilde{\boldsymbol{\varphi}}^{(i)}(\mathbf{z}')$ gives a convex upper bound of Eq. 4.11:

$$\tilde{\mathcal{L}}^{(i)}(\mathbf{z}') = \|\mathbf{B}^{(i)}_{xy} \tilde{\boldsymbol{\varphi}}^{(i)}(\mathbf{z}') - \mathbf{w}^{(i)}\| + h(\tilde{\boldsymbol{\varphi}}^{(i)}(\mathbf{z}')) \tag{4.14}$$

In our experiment, we find that using this convex upper bound as training loss, is sufficient to give lower error on the training set compared to our already strong NRSfM baseline.

### 4.4.4 Learning the 3D pose estimator

We use the state-of-the-art integral regression network [184] as our student pose estimator. The network directly predicts 3D coordinates of landmarks in the image coordinate. During

| | P-MPJPE | MPJPE | depth error |
|---|---|---|---|
| Ranklet [43] | 281.1 | - | - |
| Sparse [93] | 217.4 | - | - |
| SPM(2k) [39] | 209.5 | - | - |
| SFC [96] | 167.1 | 218.0 | 135.6 |
| KSTA(5k) [61] | 123.6 | - | - |
| RIKS(5k) [71] | 103.9 | - | - |
| Consensus [110] | 79.6 | 120.1 | 111.5 |
| Deep-NRSfM* [94] | 73.2 | 101.6 | 76.5 |
| Weaksup-bs | 61.2 | 86.2 | 75.3 |
| Ours | 56.4 | 80.9 | 71.2 |

Table 4.1: Compare with NRSfM methods on the training set of H3.6M ECCV18 challenge dataset. KSTA, RIKS are evaluated on a subset of 5k images, and SPM is evaluated on 2k images. $*$ Our implementation of Deep-NRSfM has significant difference compared to the original paper.

training, the $(x, y)$ coordinate is directly supervised by 2D landmark annotations; while $z$ coordinate is supervised by our knowledge distilling loss (Eq. 4.14). The proposed learning objective is:

$$\min_{\boldsymbol{\theta}} \sum_{i} \|f_{xy}(\mathbf{I}^{(i)}; \boldsymbol{\theta}) - \mathbf{w}^{(i)}\|_1 + \tilde{\mathcal{L}}^{(i)}(f_z(\mathbf{I}^{(i)}; \boldsymbol{\theta})), \tag{4.15}$$

where $f_{xy}$,$f_z$ denote the output of the network at $(x, y)$ and $z$ coordinates; and $\boldsymbol{\theta}$ refers to the network weights. For the knowledge distilling loss $\tilde{\mathcal{L}}$, we use $L_2$ norm for the reprojection error, and $L_1$ norm for the regularizer in our experiment. The regularizer is weighted by an empirically found coefficient, which is 0.3 in our experiment.

## 4.5 Experiment

### 4.5.1 Implementation details

**Data preprocessing:** We assume no knowledge of 3D label in both training and testing. We crop the image according to the 2D human bounding box, and then resize and pad such that it is 256x256 resolution. The 2D points are then represented by the patch coordinate. In evaluation, we follow the same procedure as in [184], which aligns the scale of the prediction by average bone length before computing the metrics.

| | 2D | 3D | MV | P-MPJPE | MPJPE |
|---|---|---|---|---|---|
| Sun *et al.* [184] | - | - | - | - | 86.4 |
| Rhodin *et al.* [165] | | ✓ | ✓ | 98.2 | 131.7 |
| Tung *et al.* [197] | ✓ | ✓ | ✓ | 98.4 | - |
| 3Dinterp. [223] | ✓ | ✓ | | 98.4 | - |
| AIGN [48] | ✓ | ✓ | | 97.2 | - |
| Tome *et al.* [193] | ✓ | ✓ | | - | 88.4 |
| Drover *et al.* [45] | ✓ | ✓ | | 64.6 | - |
| Weaksup-bs | ✓ | | | 67.3 | 95.0 |
| Ours | ✓ | | | 62.8 | 86.4 |
| + MPII | ✓ | | | **57.5** | **83.0** |

Table 4.2: Compare with weakly supervised methods on H3.6M validation set. Supervision source used by each method is marked: '2D' refers to 2D landmark annotation; '3D' represents any training source with 3D annotation, including synthetic 3D dataset, external human 3D model, etc.; 'MV' is the abbreviation for multi-view.

| | Direct. | Disc. | Eat | Greet | Phone | Photo | Pose | Purch. | Sit | SitD | Smoke | Wait | Walk | WalkD | WalkP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3Dinterp. [223] | 78.6 | 90.8 | 92.5 | 89.4 | 108.9 | 112.4 | 77.1 | 106.7 | 127.4 | 139.0 | 103.4 | 91.4 | 79.1 | - | - |
| AIGN [48] | 77.6 | 91.4 | 89.9 | 88.0 | 107.3 | 110.1 | 75.9 | 107.5 | 124.2 | 137.8 | 102.2 | 90.3 | 78.6 | - | - |
| Drover *et al.* [45] | 60.2 | 60.7 | 59.2 | 65.1 | 65.5 | 63.8 | 59.4 | 59.4 | 69.1 | 88.0 | 64.8 | 60.8 | 64.9 | 63.9 | 65.2 |
| Weaksup-bs | 58.8 | 62.4 | 56.7 | 59.8 | 68.6 | 60.8 | 59.7 | 81.0 | 93.4 | 68.5 | 75.8 | 65.9 | 61.5 | 67.6 | 65.0 |
| Ours | 54.7 | 57.7 | 54.8 | 55.8 | 61.6 | 56.3 | 52.7 | 73.7 | 95.5 | 62.3 | 68.5 | 60.8 | 55.5 | 64.0 | 58.0 |
| +MPII | **50.3** | **48.9** | **52.7** | **53.9** | **59.9** | **50.7** | **48.3** | 70.9 | 82.6 | **58.0** | 65.3 | **54.7** | **50.8** | **57.7** | **55.6** |

Table 4.3: Per action PA-MPJPE reported on H3.6M validation set. Our approach performs favorably compared to other weakly supervised methods.

**3D pose estimation network:** We select the integral regression network [184] due to its state-of-the-art performance in human pose estimation. Throughout our experiment, we use ResNet50 as the backbone for the regression network, and the input image resolution is set as $256 \times 256$. Using deeper backbone network (e.g. ResNet152) and higher image resolution would improve result, as already shown in [184]. We choose this cheaper setting for a fairer comparison with other weakly supervised methods which use ResNet50.

During training, we follow most of the settings in [184], i.e. the base learning rate is 1e-3, and it drops to 1e-5 when the loss on the validation set saturates. Limited by our computational resources, we use a smaller batch size of 32.

**Deep-NRSfM:** We use dictionaries with 6 levels. The size for the dictionaries from lower level to higher is: 256, 128, 64, 32, 16, 8. When learning the dictionaries, the sparsity weight ($\lambda$ in Eq. 4.2) is selected through cross validation and set as 0.01. For more details of our modified version of Deep-NRSfM, we refer the reader to our supplementary material.

### 4.5.2 Experiment setup

**Dataset:** We validate our method on Human3.6M dataset (H3.6M) [83], which is the major dataset used in current 3D human pose estimation research. Despite our experiment is focused on human pose estimation, we'd like to emphasize that the proposed method is a general algorithm. Unlike other weakly supervised methods which are deeply coupled with external 3D human model, our method doesn't require any target specific prior knowledge, thus should be applicable to other type of objects without restriction.

H3.6M includes sequences of 11 actors performing 15 type of actions captured from 4 camera locations. Footage of 7 out of 11 actors are released for training/validation. We follow the experiment convention conducted by prior papers: 5 subjects (S1, S5, S6, S7, S8) are used as training set, and 2 subjects (S9, S11) for testing. Although H3.6M dataset comes with 3D annotation, we use only 2D annotation during training, and 3D labels are kept for validation.

Strategies to sample frames from the training footage can have a direct impact on validation accuracy. For reproducibility, we use the subset (35k+ images) selected by H3.6M ECCV18 Challenge for training. We augment the training set through random image warping and perturbation as in [184].

**Evaluation metric:** We follow the two common evaluation protocols used in literature, and report both of them.

- MPJPE: mean per joint positioning error measures the mean euclidean distance between the reconstructed and ground truth joints after shifting them to have the same root joint coordinate.
- PA-MPJPE: Align the reconstructed joints to the ground truth through rigid transformation before evaluating MPJPE. This metric is more often used in NRSfM to measure the correctness of the reconstructed shape.

In addition, we also report 'depth error' which measures the mean difference along z-axis. This is the most important metric to validate our method, because the core problem of weakly supervised learning is how to recover depth without annotation.

**Weakly supervised learning baseline:** As previously mentioned, a simple weakly supervised learning baseline is using the depth output from our Deep-NRSfM method as training labels. We use this baseline (refer as "Weaksup-bs") to validate the contribution of our novel knowledge distilling loss. To train the pose estimation network, we employ L1

regression loss which has been proven effective in [184].

**Weighting value for the $L_1$ regularizer:** We study the effect of different weighting values for the $L_1$ regularizer in the propoesed knowledge distilling loss (Eq. 4.14). As shown in Table 4.4, under a reasonable range (0.1-0.5) of the weights, our method consistently outperforms the baseline.

| $L_1$weight | 0.01 | 0.1 | 0.3 | 0.5 | Weaksup-bs |
|---|---|---|---|---|---|
| depth error (mm) | 79.0 | 74.6 | **73.1** | 76.7 | 78.0 |
| PA-MPJPE (mm) | 73.0 | 73.6 | **70.5** | 71.0 | 75.8 |

Table 4.4: Comparing different weighting values for the $L_1$ regularizer in Eq. 4.14. Numbers reported on the validation set of H3.6M ECCV18 challenge.

**Using extra data from MPII:** Prior works [250] has shown that including external 2D data such as MPII [12] as training source can improve generalization ability of the learned 3D pose estimator. Thus, we also report result of our method trained with H3.6M+MPII. Due to our current method does not handle missing joints, we apply our proposed knowledge distilling loss only to those MPII images with complete 2D skeleton annotation; for images with occluded/out-of-view joints, we only use 2D regression loss as in [184].

### 4.5.3 Compare with NRSfM methods

We compare with 7 state-of-the-art NRSfM methods on our training set (35k+ images from H3.6M ECCV18 Challenge). We find this dataset is challenging to the compared methods due to: 1) large variation in camera positions; 2) difficult poses such as sitting and prone occupy a significant portion of the dataset; 3) variation in scale is large, due to the fact that without the knowledge of 3D, we cannot normalize 2D projections by distance or calculating bone length. The best we can do is to normalize 2D points by the size of 2D bounding box. This leads to certain pose e.g. sitting appears larger compared to others after normalization; 4) some of the methods fails to cope with a large number of samples (e.g. $>$5k). For those methods, we report result on the largest subset they can handle. We also try to compare with the recently proposed MUS [7], but their implementation fails to handle H3.6M dataset with large number of frames.

Despite of these difficulties, our implementation of Deep-NRSfM outperforms all of them. As shown in Table. 4.1, it reduces depth error by more than $33\%$ compared to the

second best. This means that switching to other NRSfM method is bound to inferior result of training a 3D pose estimator.

More interestingly, although our weakly supervised learning baseline (Weaksup-bs) is trained to reconstruct the same depth value produced by deep NRSfM, it actually gets slightly lower depth error compared to its regression target. This indicates that the deep image prior is taking effect, but still restricted by the noisy labels from Deep-NRSfM.

Finally, the pose estimation network learned by our knowledge distilling loss reduces the depth error from Deep-NRSfM's 76.5mm to 71.2mm. As shown in Fig. 4.3 and 7.1, this 5.3mm average difference includes a huge improvement in cases such as identifying if a leg is stretching towards or away from the camera.

### 4.5.4 Compare with weakly supervised methods

We compare with other weakly supervised 3D pose learning methods on the H3.6M validation set. In Table. 4.2, we first list the performance of Integral regression network by Sun *et al*. [184] as a supervised learning baseline. We copied its MPJPE (corresponding to ResNet50 with $256 \times 256$ input size and $I_1$ loss) from their paper. Since in our experiment, we're using exactly the same pose estimation network architecture, this serves as the upper bound of accuracy, which a weakly supervised learning method can achieve.

Next, we list results from 7 weakly supervised methods, and the type of their training source is marked. '2D' refers to 2D landmark annotation; '3D' represents any external 3D training source, including 3D human models, unpaired 3D skeleton dataset, synthetic dataset with 3D annotations, etc.; MV is the abbreviation for multi-view footage. We find that our method outperforms all the compared methods, while using the least amount of supervision. We also experiment with including MPII as extra training source, which leads to more error reduction. Fig. 4.4 shows some qualitative results of our method on the validation set. For per action error break down, we list PA-MPJPE of 13 different actions in Table 4.3.

## 4.6 Conlusion

In this paper, we presented a weakly supervised 3D pose learning algorithm requires zero 3D annotation. We proposed a novel loss to distill knowledge from a general type of

NRSfM method based on dictionary learning. We also established a strong NRSfM baseline on a challenging dataset, beating all the state-of-the-arts. Despite its current sucess, the limitations of our method are: 1) we require weak perspective projection, thus objects with strong perspective change is not ideal for the proposed method; 2) we do not model missing labels yet, thus another iteration is needed to extend the method to datasets with lots of occluded/out-of-view objects. We leave these for future work.

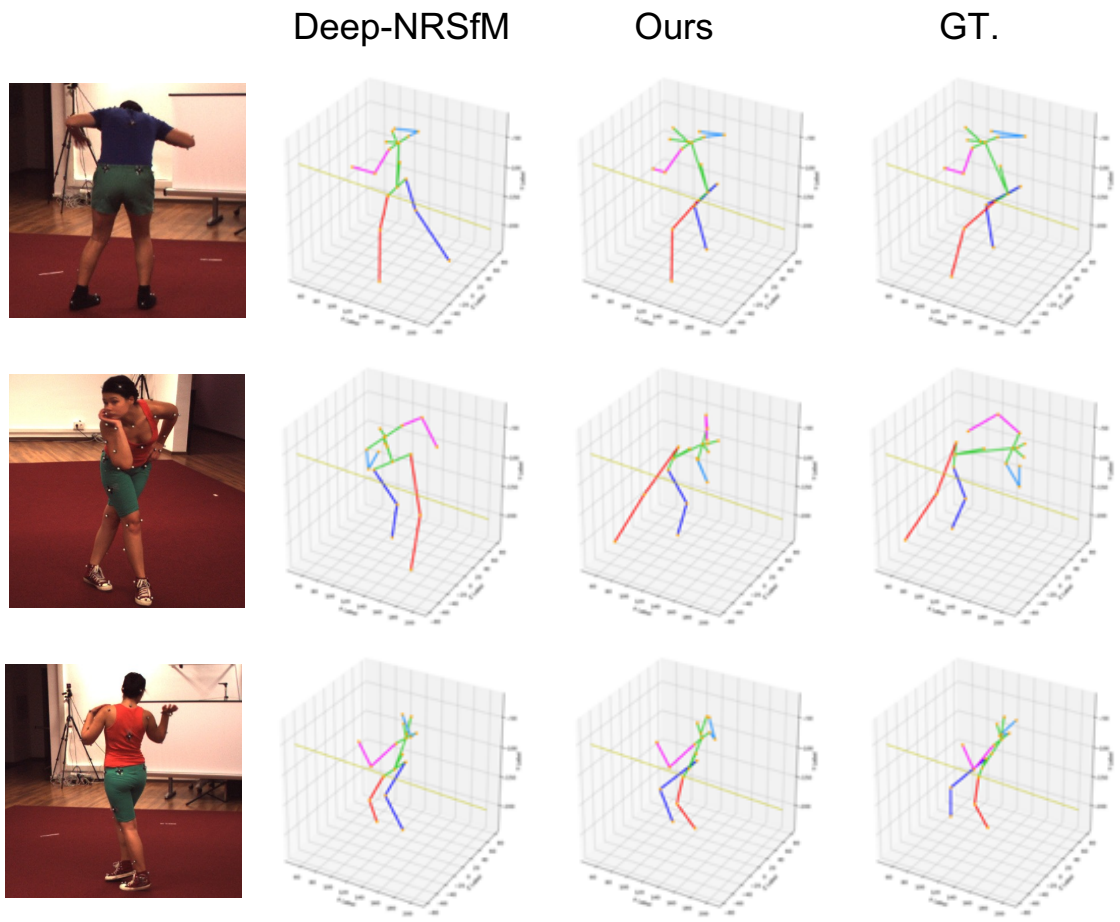



Figure 4.1: NRSfM methods often achieve poor reconstructions when the 2D projections have strong ambiguity. Our proposed knowledge distilling method lets the student pose estimation network (3rd column) correct some of the mistakes made by its NRSfM teacher (2nd column).

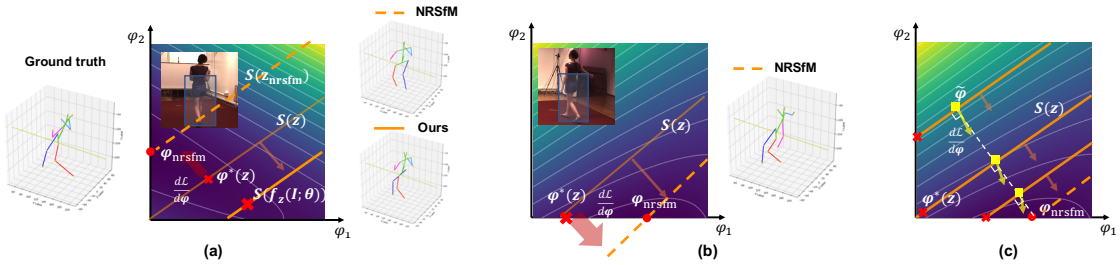



Figure 4.2: Illustration of the proposed knowledge distilling algorithm. **(a)** For illustration purpose, we assume the code $\boldsymbol{\varphi}$ is 2-dimensional. We plot the cost function (Eq. 4.9) as a 2D heatmap. The NRSfM solution $\boldsymbol{\varphi}_{\text{nrsfm}}$ is approximately the minima of this heat map (represented as red dot). Given a depth hypothesis $\mathbf{z}$, all the codes satisfies $\mathbf{z}$ forms a subspace $\mathcal{S}(\mathbf{z})$, which is shown as the orange line. The quality of a depth hypothesis is evaluated by the best point on its subspace, denoted as $\boldsymbol{\varphi}^*(\mathbf{z})$ (red cross). Given different depth hypothesis is equivalent to parallel translate the line. Suppose $\mathbf{z}$ is free to have any value, then minimizing our loss function (Eq. 4.10) would push the line to cross $\boldsymbol{\varphi}_{\text{nrsfm}}$(see the dashed orange line). This gives the same wrong depth reconstruction as the NRSfM method. (**b**) Suppose we get another image of similar pose but with less 2D projection ambiguity. In this case, NRSfM gives correct shape recovery. Since texture features are similar for both images, the pose estimation network is implicitly constrained to make similar depth predictions. Then minimizing our loss for both images would lead to a better solution for image 1 (shown as solid orange line), because gradients are larger from the 2nd image due to the fact that it has less ambiguity. **(c)** We approximate the loss by evaluating at the projection of $\boldsymbol{\varphi}_{\text{nrsfm}}$ on the subspace (yellow square). This approximation is a convex upper bound for the original loss. It would still reflect the degree of projection ambiguity, and push the subspace (lines) torwards $\boldsymbol{\varphi}_{\text{nrsfm}}$.

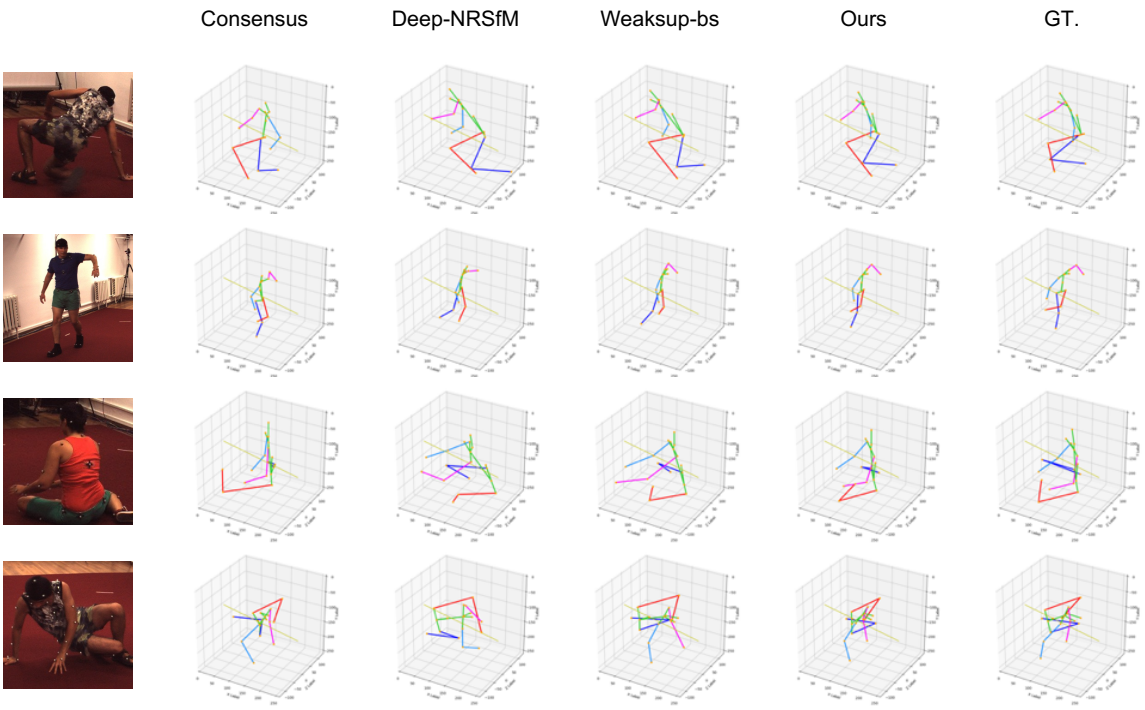



Figure 4.3: Visual comparison of NRSfM methods versus methods which include image as extra constraint (i.e. our weakly supervised baseline and our knowledge distilling method) on the training set. Our method shows significant improvement over its teacher, i.e. deep-NRSfM. Skeletons are rendered from side view for better visualization of the difference in depth reconstruction. We use red and magenta to color left leg and arm, while blue and dodgerblue are used to color right leg and arm.

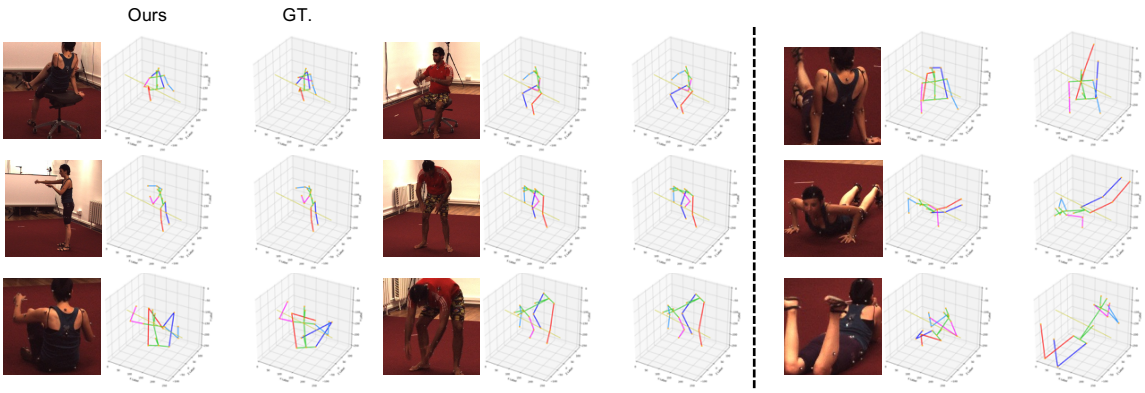



Figure 4.4: Qualitative results of ours on H3.6M validation set. The right part shows some of our failure cases. Our method may fail under severe occlusion and rare body poses.

# Part II

# Dense reconstruction for dynamic scenes

# Chapter 5

# Web Stereo Video Supervision for Depth Prediction from Dynamic Scenes

## 5.1 Introduction

Recovering depth maps of nonrigid scenes from monocular video sequences is a challenging problem in 3D vision. While rigid scene reconstruction methods can use geometric consistency assumptions across multiple frames, the problem becomes severely under-constrained for nonrigid scenes. As a result, nonrigid reconstruction methods must rely more heavily on strong scene priors. In fact, we see that data-driven single image reconstruction methods, which can learn *only* scene priors, sometimes outperform multi-frame geometric methods on nonrigid scenes (Sec. 5.5). In this work, we attempt the model nonrigid scene priors in a *data-driven* manner, by training on real-life stereoscopic videos, while also learning to taking advantage of geometric cues between neighboring frames of video.

Recent advances in data-driven methods have shown some advantages over traditional 3D reconstruction pipelines. However, due to restrictions on architectures and available training data, such approaches have largely been used to predict depth from from single images [28, 47, 107, 117, 225] or multiple views of rigid scenes [174, 200], or have been trained on narrow domains, e.g, driving data [56].

To overcome these limitations, we introduce a new large-scale (1.5M frame) dataset collected in-the-wild from internet stereo videos, which contains large amounts of nonrigid

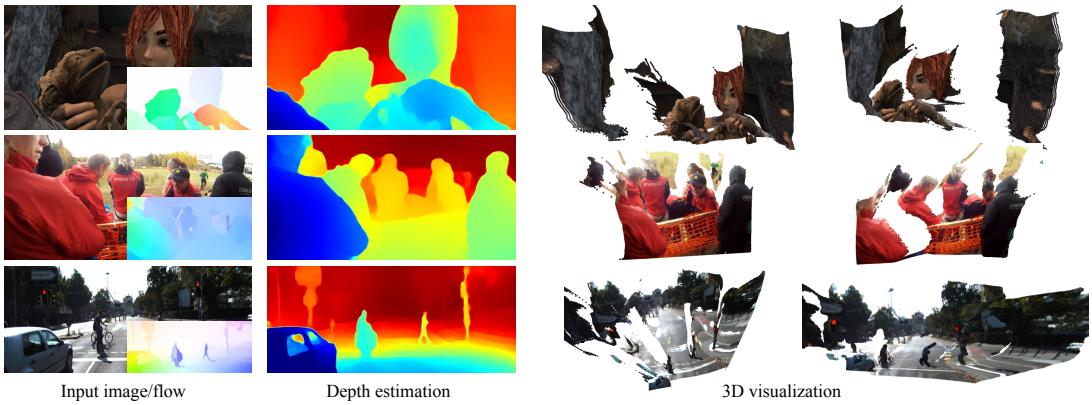



Figure 5.1: Depth prediction for nonrigid scenes from our multi-view depth estimator, which is trained on a new large scale database of web stereo videos.

objects and diverse scene types. We use derived stereo disparity for training, and at test time, predict depth from a pair of single-view, sequential frames. Our approach is designed so that it learns to utilize both semantic (single-image) and geometric (multi-frame) cues to address the nonrigid reconstruction problem.

One challenge of using Internet stereo videos as a training source is that they contain unknown (and possibly temporally varying) camera intrinsics and extrinsics, and consequently, we cannot directly translate disparities to depth values (even up to scale). This prevents the use of existing regression loss including the scale invariant logarithmic depth/gradient loss [47]. Instead, we observe that in a stereo camera configuration, the *difference* in disparity between two pixels is proportional to the difference in their inverse depth. This motivates a new normalized multiscale gradient loss that allows for supervision of depth prediction networks directly from estimated disparities from uncalibrated stereo video data. Compared to the ordinal loss used in [28, 225], we show that our proposed loss has the advantage of retaining distance information between points, and yields smoother, more accurate depth maps.

While the computed disparity maps contain errors, we show that training a deep network on our dataset with the proposed loss generalizes well to other datasets, and that by using temporal information in the form of flow maps and sequential frames as input into the network, we are able to improve over single image depth prediction methods. Our code and data is made available [1].

[1] https://sites.google.com/view/wsvd/

In summary, we present the following contributions:

1. A simple architecture that leverages stereo video for training, and produces depth maps from monocular sequential frames with nonrigid objects at test time. To our knowledge this is the first multi-frame data-driven approach that outperforms single-image reconstruction on nonrigid scenes.
2. A new stereo video dataset that features a wide variety of real world examples with nonrigid objects (e.g., humans) in an unconstrained setting.
3. A novel loss function that allows us to train with unknown camera parameters, while outperforming previously used ordinal losses.

## 5.2 Related Work

**Traditional geometric approaches** for predicting depth from video sequences rely on SLAM [138] or SfM [172] pipelines for camera pose and 3D map estimation, followed by dense multiview stereo [174] to get per-pixel depth values. These methods are mature systems that produce highly accurate reconstructions in the right circumstances. However, they rely on hand-designed features, assumptions such as brightness constancy across frames, and often require a good map initialization and sufficient parallax to converge. Additionally, while camera pose estimation can handle some amount of nonrigid motion by simply ignoring those regions, dense multiview stereo requires a rigid scene.

Non-rigid SfM methods try to recover 3D by replacing the rigid assumption with additional constraints. Bregler et al. [19] introduces a fixed rank constraint on the shape matrix for non-rigid structures. Numerous innovations have followed, introducing additional priors to make the problem less ambiguous and to scale to dense reconstruction[9, 39, 93, 102, 109, 251]. These approaches usually assume weak perspective cameras and rely on hand-designed features for input feature tracks.

Other recent works[104, 162] explore dense reconstructions from two perspective frames, using an as-rigid-as-possible (ARAP) assumption to address the scale ambiguity issue inherent in non-rigid reconstruction. Although promising results have been shown on mostly rigid scenes, their ARAP assumption is not enough to handle complicated dynamic scenes. Our approach learns priors from data, and we show that it can often produce good result for highly dynamic scenes.

**Data driven approaches** that leverage deep networks for scene priors and feature learning, have become a potential way to overcome the limitations of traditional methods. Recent works [80, 200, 246] demonstrate promising result for rigid scene reconstruction. However, since they include explicit rigid transformation inside their network architecture, they cannot handle non-rigid motion. We propose a new data-driven method that focuses on non-rigid objects and diverse scenes, and introduce a new internet stereo video dataset to train this approach.

**Supervision using depth sensors** is a common strategy for existing methods that directly regress depth values. These approaches rely on datasets collected for example, by laser scanner [171], Kinect depth camera [47], car-mounted lidar sensor [47, 106], synthetic rendered data (for pretraining) [68], or dual-camera iPhone [118]. One challenge of requiring specialized hardware is that it is hard to acquire sufficiently diverse data for training, and as such they tend to be used only in constrained domains, e.g., driving sequences.

Recent approaches have proposed using more common stereo camera rigs for training depth prediction networks [58, 233, 238]. These approaches are trained using stereo video data on KITTI [56], by treating depth prediction as an image reconstruction problem. Our approach differs to theirs in that we are learning from stereoscopic videos with unknown camera parameters, and we also utilize temporal information at test time. Another method, Deep3D [229] uses 3D movies for training, however this approach focuses on synthesizing novel stereoscopic views rather than scene reconstruction.

**Supervision using internet images** is a powerful tool that allows for the collection of diverse datasets for learning depth reconstruction priors. MegaDepth [117] generates a set of 3D reconstructions using traditional SfM and multi-view stereo reconstruction pipelines from internet images. These reconstructions serve as ground truth normalized depth maps for supervision. Another recent work scrapes stereo *images* from the web, and computes disparity from these [225]. This approach uses ordinal depth constraints from these disparity estimates to train a single image depth prediction network.

In contrast, we are interested in data that allows us to take advantage of multiple frames of video. We show that reconstruction quality is improved when such data is available, indicating that it is possible to learn both scene priors from single images, and geometric information from multiple frames. In addition, we compare our loss formulation to that presented in the above method, and show that it outperforms relative depth constraints.

A concurrent work [115] derives training data for scenes with humans in it by leveraging so called "mannequin challenge" videos where people hold as still as possible while the camera moves. This data allows for depth supervision using standard multi-view reconstruction techniques.

**Supervision using single-view video data** has been a popular recent approach for scene reconstruction. In this case, supervision is derived from temporal frames of a video by predicting depth and camera pose, and computing a loss based on warping the one frame to the next based on these values. Then at test time, a single frame is used for depth prediction [55, 126, 127, 216, 233, 247, 248, 254]. As geometric projection is only valid for rigid scenes, these methods often include a rigidity mask, where the loss is treated differently outside these regions [247, 254], and introduce regularization to prevent the method from degenerating.

While these approaches are elegant as they do not require extra supervision other than a single-view video, in practice many of these methods require both known intrinsics, and assume mostly rigid scenes. In our work we are interested in highly non-rigid scenes, and therefore do not explicitly rely on a geometric reprojection loss. Instead we use temporal flow and flow-warped images as an input to our network to learn the geometric relationship between flow and depth.

## 5.3 Dataset

We introduce the Web Stereo Video Dataset (WSVD), a large, in-the-wild collection of stereo videos featuring many nonrigid objects (especially people). Figure 5.3 shows a selection of representative frames from WSVD.

To collect WSVD, we scraped YouTube for videos that were tagged as being viewable in stereo 3D. 3D videos on YouTube are stored internally in a left/right format, meaning that each video is squeezed in half width-wise and stored side by side in a single video file. We additionally searched for videos on Vimeo with a "side-by-side 3D" query to match this format (Vimeo does not naively support 3D). In theory all videos with the proper tag should be 3D, but in practice, a large number of these videos were either regular monocular video, or different kinds of stereo, e.g., anaglyph, top/bottom, etc. We therefore conduct two stages of filtering to make this data usable.

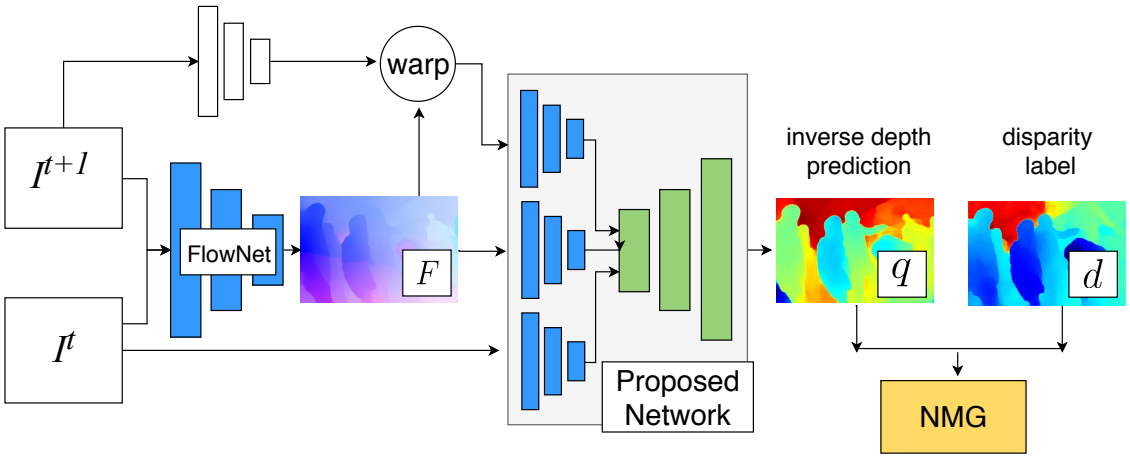



Figure 5.2: Our network takes input from: the frame $I^t$, the second frame $I^{t+1}$ and the flow between $I^t, I^{t+1}$. We extract feature pyramids for both input images and the flow map. Moreover, we warp the feature pyramid for $I^{t+1}$ with the flow. The warped feature pyramid of $I^{t+1}$ is then fused with the feature pyramids for $I^t$ and the flow map, fed into a decoder with skip connections and produces a depth map. This is supervised with the disparity directly using our Normalized Multi-Scale Gradient (NMG) Loss.

**Filtering** Our initial scraping yielded $7k+$ videos from YouTube and Vimeo. We first identified all videos that were actual left-right stereo videos from this set. To do this, we computed the MSE of the left and right half of each video, split the videos into two classes based on on this metric and then manually removed outliers.

After this step, all videos remaining were left/right stereo videos, but still not all of them were usable. For further filtering, we split up each video into shots using histogram differences [4], and performed a per-shot categorization of good and bad videos. To do this, we first calculated the average brightness of the middle frame to filter out shots with black screens. We also performed text detection to remove shots with large text titles.

We then automatically computed the disparity for the middle frames per shot using a flow approach [82]. We found that many samples displayed substantial lens distortion, near-zero baselines, vertical disparities, color differences, inverted cameras, and other poor stereo characteristics. To reject those shots, we used the following criteria as an initial step for filtering: pixels with vertical disparity $> 1$ pixel is greater than $10\%$; range of horizontal disparity is $< 5$ pixels, and the percentage of pixels passing a left-right disparity consistency check is $< 70\%$. This was followed by a second curation step, to remove videos with obviously incorrect disparity maps (e.g., due to radial distortion, or incorrect camera

Figure 5.3: Random selection of frames from WSVD showing a sampling of the diversity of scenes and subjects contained. Below each image we show the computed disparity map used for supervision.

configurations).

Next, we removed static frames by filtering frames with maximum flow magnitude less than 8 pixels. For each remaining frame, we calculated disparity map as our ground truth using FlowNet2.0 [82], which we found to produce the best results. The left-right disparity check is also used to mask out outliers, which are not used for supervision. Finally, we are left with 10788 clips from 689 videos, consisting of around 1.5 million frames.

**Analysis** In order to understand what type of content is present in the WSVD dataset, we run a two-stage image classifier similar to Mask R-CNN [76] on the middle frames of each shot. The classifier is pre-trained on the 600 classes annotated in the Open Images dataset [98]. We retained bounding boxes with confidence score $\geq 0.7$. The analysis of the results indicates that roughly $79\%$ of the video sequences contain either humans, animals, or vehicles, which are likely to display nonrigid motion. In Figure 5.4, we show the word cloud generated from the class frequencies of the detected bounding boxes. In contrast to other datasets, i.e. KITTI and NYU, WSVD reflects the diverse content that people watch on YouTube, with many nonrigid objects, especially humans.


Figure 5.4: WSVD word cloud of the two hundred most frequent classes estimated with an object detector trained on OpenImages [98] bounding boxes, showing a high quantity of nonrigid objects, especially people.

## 5.4 Approach

Our supervision comes in the form of a disparity map. Assuming that the stereoscopic videos are captured by two horizontally placed cameras, and the vertical disparity has already been removed, then disparity can be translated to inverse depth $q$ using the following equation:

$$q = \frac{d - (c_x^R - c_x^L)}{fb} \tag{5.1}$$

Where $d$ is the disparity, $b$ is the camera baseline, $f$ is the focal length, and $c_x^R$, $c_x^L$ denote the horizontal coordinates of the principal points for the left/right cameras. $c_x^R - c_x^L$ defines the minimum possible disparity $d_{\text{min}}$ in the camera configuration. In practice, $d_{\text{min}}$ can have arbitrary values depending on the stereo rig configuration and post-production. For example, to release visual fatigue, most stereo rigs for 3D movies are towed-in to place the object of interest at the 0 disparity plane, which creates negative disparity values.

Unlike most other video datasets with depth supervision, this dataset has unknown and variable focal length $f$, and camera stereo configuration (baseline $b$ and $d_{\text{min}}$). Among those, $d_{\text{min}}$ is the key parameter preventing us from converting the estimated disparity into an inverse depth map up to scale, as is commonly done in other self-supervised learning

methods [58, 117]. This term also prevents us to apply the widely used scale-invariant logarithmic depth (gradient) loss [47], due to the fact that subtracting the mean/neighboring pixel's logarithmic depth value is not enough to cancel out $d_{\text{min}}$.

Although in theory, we could estimate $d_{\text{min}}$ with the disparity value of pixels at infinity distance, it would not be robust due to the fact that regions in distance does not always present in videos, and that the usual choice for such regions i.e. textureless sky, are likely to have incorrect disparity values.

Due to this issue, prior work [225] only uses an ordinal relation for supervision, and does not attempt to recover the relative distance from the disparity map. We take a different approach – the proposed loss function takes supervision from the whole disparity map, and preserves the continuous distance information between points in addition to ordinal relation. In comparison, ordinal loss [28, 225] only enforce binary ordinal relation between a sparse set of pixel pairs.

### 5.4.1 Normalized multiscale gradient (NMG) loss

From Eq. 5.1 we derive that the difference in disparity between two pixels is proportional to the difference in their inverse depth:

$$q_i - q_j = \frac{d_i - d_j}{fb} \tag{5.2}$$

This allows us to design a novel loss invariant to the minimum possible disparity value. The idea is to enforce the gradient of inverse depth prediction to be close to the gradient of the disparity map up to scale (normalized). The gradient is evaluated at different spacing amounts (multiscale) to include both local/global information [200]. The loss can be written as:

$$\mathcal{L} = \sum_k \sum_i |s\nabla_x^k q_i - \nabla_x^k d_i| + |s\nabla_y^k q_i - \nabla_y^k d_i|, \tag{5.3}$$

where $\nabla_x^k$, $\nabla_y^k$ denote the difference evaluated with spacing $k$ (we use $k = \{2, 8, 32, 64\}$); and the scale ratio $s$ is estimated by:

$$s = \frac{\sum_k \sum_i |\nabla_x^k d_i| + \sum_k \sum_i |\nabla_y^k d_i|}{\sum_k \sum_i |\nabla_x^k q_i| + \sum_k \sum_i |\nabla_y^k q_i|}. \tag{5.4}$$

We choose to scale the inverse depth prediction to match disparity, and not the other way around, as this is more robust when the signal-to-noise ratio of the disparity map is low, which often happens when the range of disparity values is narrow, either due to a small baseline, or distant scenes. Scaling such noisy disparity with low contrast would amplify noise in the depth prediction.

Compared to the scale-invariant gradient loss of Eigen et al. [47], the proposed loss NMG is different in that it is defined in terms of disparity, not the log of depth. which is inapplicable in our scenario since we do not have depth as supervision.

Compared to the ordinal loss [28, 225], our NMG loss enforces relative distance and smoothness in addition to pairwise ordinal relation. As shown in Figure 5.5, the NMG loss yields more accurate global structure and preserves edge details better.

### 5.4.2 Depth prediction network

To utilize temporal information for depth prediction, we propose a network architecture inspired by recent work [225], modified to take input from two sequential frames and their optical flow. The general architecture is a multi-scale feature fusion [225, 254] net with three feature pyramids streams (features for 1st image, warped features for 2nd frame, and features for the flow map). These features are then projected and concatenated together and fed into an decoder with skip connections. Please refer to the supplementary material for a detailed description of the architecture.

**Feature pyramids** We use a ResNet-50 network to extract feature pyramids from the 1st and 2nd input frames. These feature pyramids have 4 levels with a coarsest scale of 1/32 of the original resolution. For flow input, we first use conv7$\times$7 and conv5$\times$5 layers with stride 2 to get features at 1/4 resolution. Then we apply 4 blocks of 2 conv3$\times$3 layers to get pyramid flow features with 4 scales. We also apply residual blocks identical to [225] to project each image feature map to 256-channels, and each flow feature map to 128-channels. Finally, we warp the features in the pyramid for the 2nd frame to the 1st using the flow.

At each level of the feature pyramids, we concatenate the feature map of the 1st frame and the warped feature of the 2nd frame, and then use a residual block to project it to 256-channels, and concatenated it with the flow features. A final residual block projects this tensor to 256-channels.

**Depth decoder** The fused feature pyramid is fed into a depth decoder with skip connections.

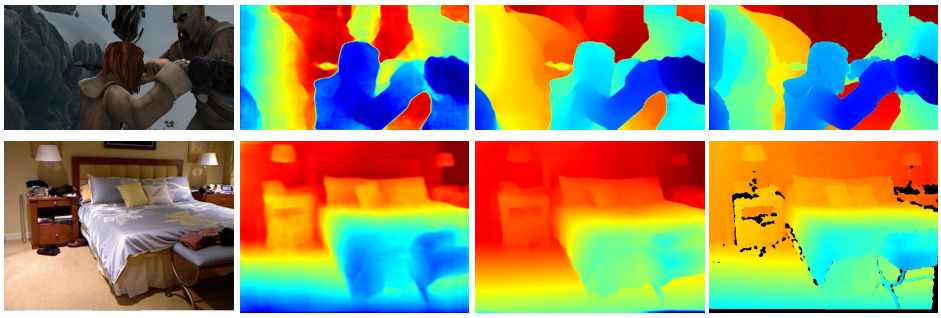



Figure 5.5: Comparison of NMG with an Ordinal loss. Top: trained on WSVDand tested on SINTEL; Bottom: trained and tested on NYU-v2.

Starting from the coarsest scale, the output from previous scale of the decoder is bilinearly upsampled and then added to the corresponding fused feature from the pyramid. After reaching 1/4 of the full resolution, a stack of 2 $3 \times 3$ conv layers and 1 bilinear upsampling layer is applied to produce final full-resolution depth prediction in log scale.

As shown in Figure 5.8 and our ablation study, we find that compared to using features only from single image, fusing features from the second frame and the flow helps identify foreground objects and produces more accurate result.

## 5.5 Evaluation

We evaluate our network on three video datasets with nonrigid scenes, and compare to other methods trained on a variety of training sets as well as a traditional geometric method (COLMAP [174]). We seek to answer the following questions:

- How does the proposed NMG loss compared to ordinal loss used by previous methods?
- How important is the temporal information used by our network to improve depth prediction?
- How does WSVD compare to other multi-view datasets when used as training source?
- How does our method generalize to other dataset compared to state-of-the-art methods?

**Experiment setup** We hold out 50 videos, and sample 597 frame pairs as our testing set.

We use all the rest videos in our dataset for training and validation. The validation set consists of 1000 frames from 500 randomly sampled clips. To avoid bias in sampling from longer videos, we randomly sample 1 clip per video at each training epoch. We train our network using Adam with default parameters and use batchsize of 6. For our ablation study, we also train a single-view depth prediction network which has identical architecture of our proposed network, but without the feature pyramids from the flow and the second input frame.

**Metrics** For comparison on SINTEL, we use the commonly used mean relative error (MRE) and scale invariant logarithmic error (SILog) evaluated on inverse depth. These two errors are measured for pixels up to 20 meters away. We also use the proposed NMG as an additional metric. To avoid bias towards testing samples with large inverse depth value, we scaled the NMG error by the mean of the ground truth inverse depth.

For KITTI, we use the 697 test samples (parired with consecutive frame for multi-view testing) from the Eigen split. We report absolute relative difference (abs rel), squared relative difference (sqr rel), relative mean squared logarithmic error (RMSE log) and percentage of error lower than a threshold (e.g. $\delta < 1.25$). Since our method cannot predict metric depth, we align the prediction to the groundtruth by the median of the depth map, as is also done in [247].

In addition, we perform evalution on our WSVD test set. We compute the NMG error excluding pixels which fail the left-right disparity consistency check (visualized as black region in Figure 5.5). While we do not claim that this is any indication of the generalization ability of our approach, we do so as an indication of how methods might perform on diverse nonrigid scenes.

### 5.5.1 Evaluation of NMG Loss

In order to evaluate the effect of our normalized multiscale gradient (NMG) loss, we compare it to the ordinal loss used in RedWeb [225]. To do this, we first compare both losses on the NYU-v2 dataset [140]. To mimic disparity maps from Internet stereoscopic images, we apply affine transformations to the ground truth depth values with uniformly sampled slope and bias parameters. These synthesized disparity maps are then used to train a single view depth estimator with different loss functions. As shown in Table 5.1, NMG loss has lower testing errors compared to ordinal loss, and its depth map prediction appears

to be smoother as well (see Fig. 5.5).

Moreover, we train the proposed multi-view depth estimator on WSVD with both losses. We see in Table 5.2, and Figure 5.5, that our loss function again yields visually smoother, and more accurate depth maps when tested on SINTEL.

| | loss by Eigen [47] | ordinal | NMG |
|---|---|---|---|
| train label | depth | {synthesized | disparity} |
| rmse | 0.467 | 0.767 | **0.706** |
| abs rel | 0.128 | 0.184 | **0.164** |
| $\delta < 1.25$ | 0.840 | 0.753 | **0.768** |
| $\delta < 1.25^2$ | 0.961 | 0.927 | **0.945** |
| $\delta < 1.25^3$ | 0.990 | 0.979 | **0.988** |

Table 5.1: Comparison between NMG and ordinal loss on NYU-v2. Trained using synthesized disparity map with randomly sampled camera parameters. Result of directly using depth map as training label (loss by Eigen [47]) is also provided as reference.

| | NMG | MRE | SILog |
|---|---|---|---|
| ordinal [225] | 0.963 | 0.350 | 0.228 |
| NMG (Ours) | **0.890** | **0.311** | **0.172** |

Table 5.2: Comparison between NMG and ordinal loss. Trained on WSVD and tested on SINTEL. Columns show different evaluation metrics.

### 5.5.2 Multi-view v.s. Single-view Prediction

We evaluate the effect of incorporating temporal information at *test* time, by comparing our approach against two baselines: 1) a single view depth prediction network; 2) our proposed network with two identical frames ($I_t$, $I_t$) and a zero flow map as input. This baseline is used to verify if the improvement is truly from temporal information, instead of the difference in network architecture. Table 5.5, and Figure 5.8 show that the two baselines achieve similar performance, our proposed method with additional temporal information as input can identify foreground objects better and gives more accurate depth estimation.

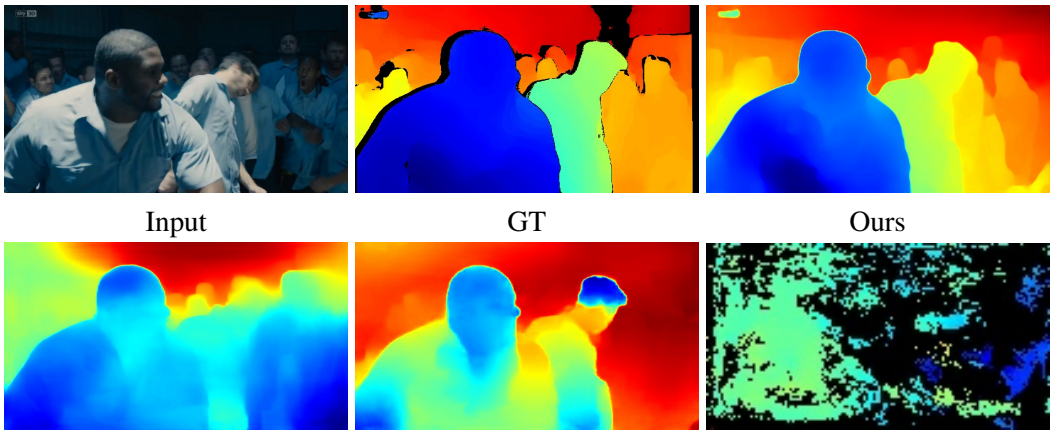



Figure 5.6: Qualitative comparison between our multi-view depth network and the state-of-the-art single/multi-view depth prediction methods.

### 5.5.3 Training on Different Multi-view Datasets

We conduct a controlled experiment of training our multi-view depth prediction network on different multi-view datasets and compare their cross dataset generalization performance. We use scale invariant logarithm loss [47] to train on KITTI; and a combination of scale invariant logarithm depth loss and gradient loss to train on the training set (SUN3D+ RGBD + Scene11) proposed by DEMON [200], consisting mostly of rigid scenes. As shown in Table 5.4 and Figure 5.7, training on our WSVD dataset has the lowest error on Sintel and outperforms the DEMON training set on KITTI, while models learned from KITTI have the worst performance on other datasets.

### 5.5.4 Cross Dataset Evaluation

We compare the generalization capability of the compared methods by testing on different video datasets, i.e. SINTEL, KITTI and WSVD. We notice that most of the scenes in the KITTI test set are rigid, therefore, we select a subset of 24 images with pedestrians, and use it to analyze the performance of handling nonrigid scenes.

Table 5.3 and Figure 5.6 show that our (single/multi-view) network compares favorably to single view depth prediction methods, i.e. MegaDepth [117], our re-implementation of RedWeb [225] and DDVO [216], which is unsupervised , and learned on KITTI. On the

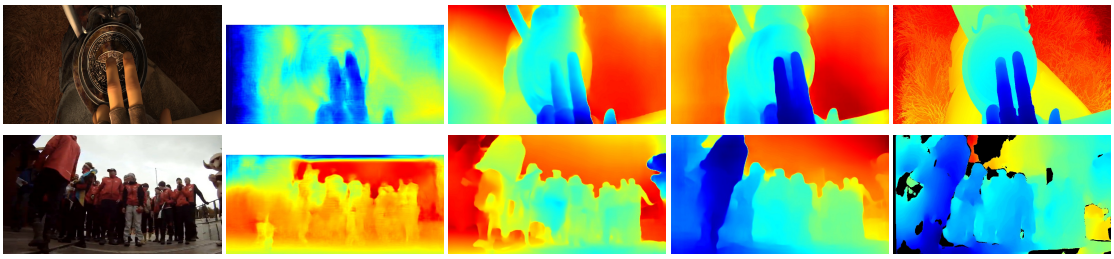



Figure 5.7: We show the performance of the same multi-view depth network trained on different datasets, and tested on Sintel and WSVD. Note that KITTI does not have groundtruth for the top region of the image, thus network trained with KITTI may produce arbitrary values in this area. For fairer comparison, we cropped out the top part.

| | | single view | | | | multi view | | |
|---|---|---|---|---|---|---|---|---|
| Test set | Metric | MegaDepth [117] | ReDWeb* | DDVO [216] | Ours | COLMAP | DEMON [200] | Ours |
| SINTEL | MRE | 0.364 | 0.401 | 0.435 | 0.333 | - | 0.478 | **0.311** |
| | SILog | 0.266 | 0.311 | 0.360 | 0.206 | - | 0.397 | **0.172** |
| | NMG | 1.212 | 1.041 | 1.492 | 0.993 | - | 1.311 | **0.890** |
| KITTI | abs rel | 0.220 | 0.234 | **0.148** | 0.230 | - | 0.235 | 0.213 |
| | $\delta < 1.25$ | 0.632 | 0.617 | **0.812** | 0.606 | - | 0.605 | 0.637 |
| KITTI | abs rel | 0.227 | 0.231 | **0.183** | 0.230 | - | 0.307 | 0.207 |
| pedestrian | $\delta < 1.25$ | 0.625 | 0.622 | **0.746** | 0.610 | - | 0.458 | 0.654 |
| WSVD | NMG | 1.269 | 1.152 | 1.505 | 1.012 | 2.021 (69.8%) | 1.418 | **0.899** |

Table 5.3: Quantitative comparison of methods. All metrics in this table are *lower is better* except $\delta < 1.25$. Best results are shown in bold, and underlining indicates the best result second only to DDVO trained on KITTI. COLMAP returns semi-dense depth values. In this case, 69.8% of pixels had valid depths computed, we compute the metric over these pixels only. * numbers reported for RedWeb is from our re-implementation of [225] due to their code has not been released.

KITTI test set, MegaDepth performs similarly to ours, which could be due to their abundant training for street and landmark scenes, while our method demonstrates better result on the pedestrian subset. Not surprisingly, DDVO performs the best on KITTI since it is trained on this dataset, but it generalizes poorly on other test sets.

We also run DEMON [200] – a deep learning method for predicting depth and camera pose from two views. We find that DEMON produces plausible results for rigid scenes with sufficient parallax, but generates worse results on nonrigid scenes. In addition, we compare the quality of reconstruction on some clips from our dataset using COLMAP [172], and show quantitative results on Table 5.3. We note that due to running time constraints we ran COLMAP on a random subset of our data.

| Training set | Multi-view test set | | |
|---|---|---|---|
| | SINTEL | KITTI | WSVD |
| | SILog | RSME(log) | NMG |
| KITTI | 0.3871 | **0.180** | 1.620 |
| DEMON | 0.222 | 0.356 | 1.205 |
| WSVD | **0.172** | 0.317 | **0.899** |

Table 5.4: Testing result of our proposed network trained on different datasets. DEMON refers to the video datasets used for training in [200], SUN3D, RGBD, and Scenes11.

| | SINTEL | | | KITTI | | | WSVD |
|---|---|---|---|---|---|---|---|
| Input | NMG | MRE | SILog | abs rel | sq rel | RMSE(log) | NMG |
| $I^t$ | 0.993 | 0.333 | 0.206 | 0.230 | 1.937 | 0.327 | 1.012 |
| $I^t$, $I^t$, 0 flow | 0.972 | 0.326 | 0.198 | 0.235 | 2.103 | 0.341 | 0.977 |
| $I^t$, $I^{t+1}$,flow | **0.890** | **0.311** | **0.172** | **0.213** | **1.849** | **0.317** | **0.899** |

Table 5.5: Evaluate the improvement due to adding the flow and second frames as input in test time. Each row describes the input to the network. i.e. ($I^t$, $I^t$) means the same image is given twice; "0-flow"means feeding flow map filled with zeros.

Throughout the evaluation above, our method consistently outperforms our single-view depth prediction baseline, which indicates that our performance gain is not solely due to our training set but also from the proposed network's ability to take temporal information into account. Finally, geometric-based methods by Kumar et al. [104] and Ranftl et al. [162] are related to ours, but we're unable to provide meaningful comparison here as their code and results are not publicly available, and the numbers reported in their paper are from an undisclosed subset of the dataset.

### 5.5.5 Limitations

One limitation of our approach, is that by using stereo disparity as supervision, we are restricting ourselves to scenes whose disparity can be computed. This can be problematic with large-scale scenes such as landscapes, where stereo baselines would have to be very large to have nonzero disparity. In Fig 5.9, we can see that our "ground-truth" is in fact incorrect in such scenes. We can also have similar difficulties in untextured regions,

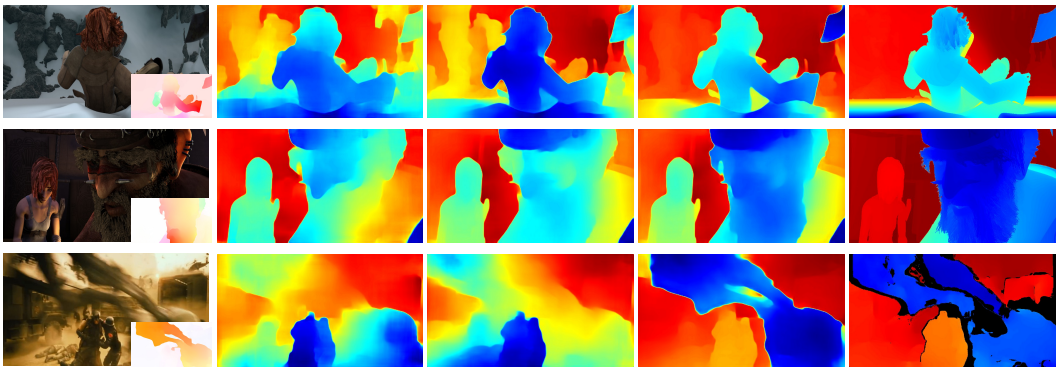



Figure 5.8: Depth maps predicted with different inputs, showing the impact of additional temporal information made available to the network. Using sequential frames and optical flow (4th column) improves baselines without temporal information.

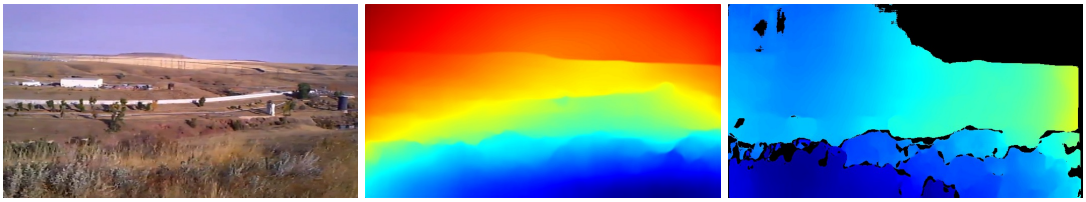



Figure 5.9: Limitations. Stereo supervision is less reliable at long distances or texture-less regions.

where reliable correspondences do not exist for supervision. In general, any biases in the reconstruction step will likely persist in our final results.

## 5.6 Conclusion

In conclusion, we present a step towards data-driven reconstruction of nonrigid scenes, by introducing the first in-the-wild stereo video dataset that features a wide distribution of nonrigid object types. We hope that this dataset will encourage future work in the area of diverse nonrigid video reconstruction, a topic with many exciting applications.

## 5.7 Appendix

### 5.7.1 Network architecture

Figure 5.10 shows our two-frame network architecture for depth prediction. Please see the caption for more detail.

### 5.7.2 Additional samples from WSVD

We include additional samples from our WSVD dataset in Figure 5.11. Each sample is rendered with an RGB frame, estimated disparity map, and optical flow.

### 5.7.3 Additional qualitative results

Additional qualitative comparison on WSVD test set is included in Figure 5.12. We compare our method, which takes sequential frames $I_t$, $I_{t+1}$ and an optical flow map as input, to two baseline monocular (single-frame) depth predictors: 1) the hourglass network provided by MegaDepth [117]; 2) our network architecture, but with the feature streams from optical flow and next frame removed. All architectures are trained with our WSVD dataset using the same loss function. As shown in the figure, our method produces more accurate estimation especially in the foreground regions.

We also include qualitative results on KITTI test set in Figure 5.13. Compared to the baselines, our method gives more details in pedestrians and light poles.

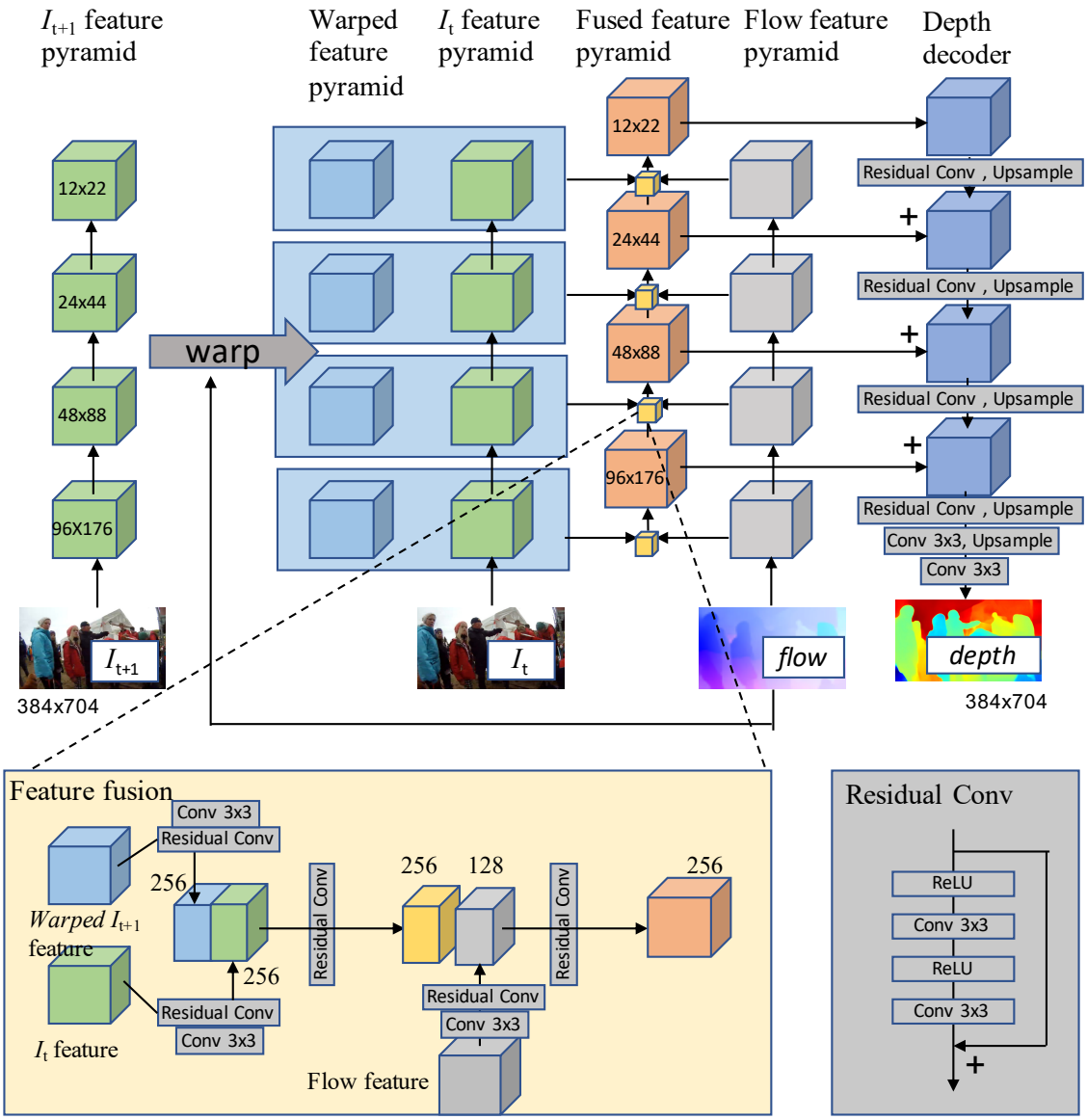



Figure 5.10: Our architecture consists of three feature extraction pyramids. The features extracted from the pyramid for frame $I_t$ is concatenated with features extracted from the next frame $I_{t+1}$, warped into $I_t$ using the flow computed between the two. This is fused with the feature pyramid extracted from the flow, and is then decoded into the final depth (right column). Feature fusion is shown in detail below (bottom left), it consists of concatenation and reprojection operations using residual convolution blocks (bottom right).

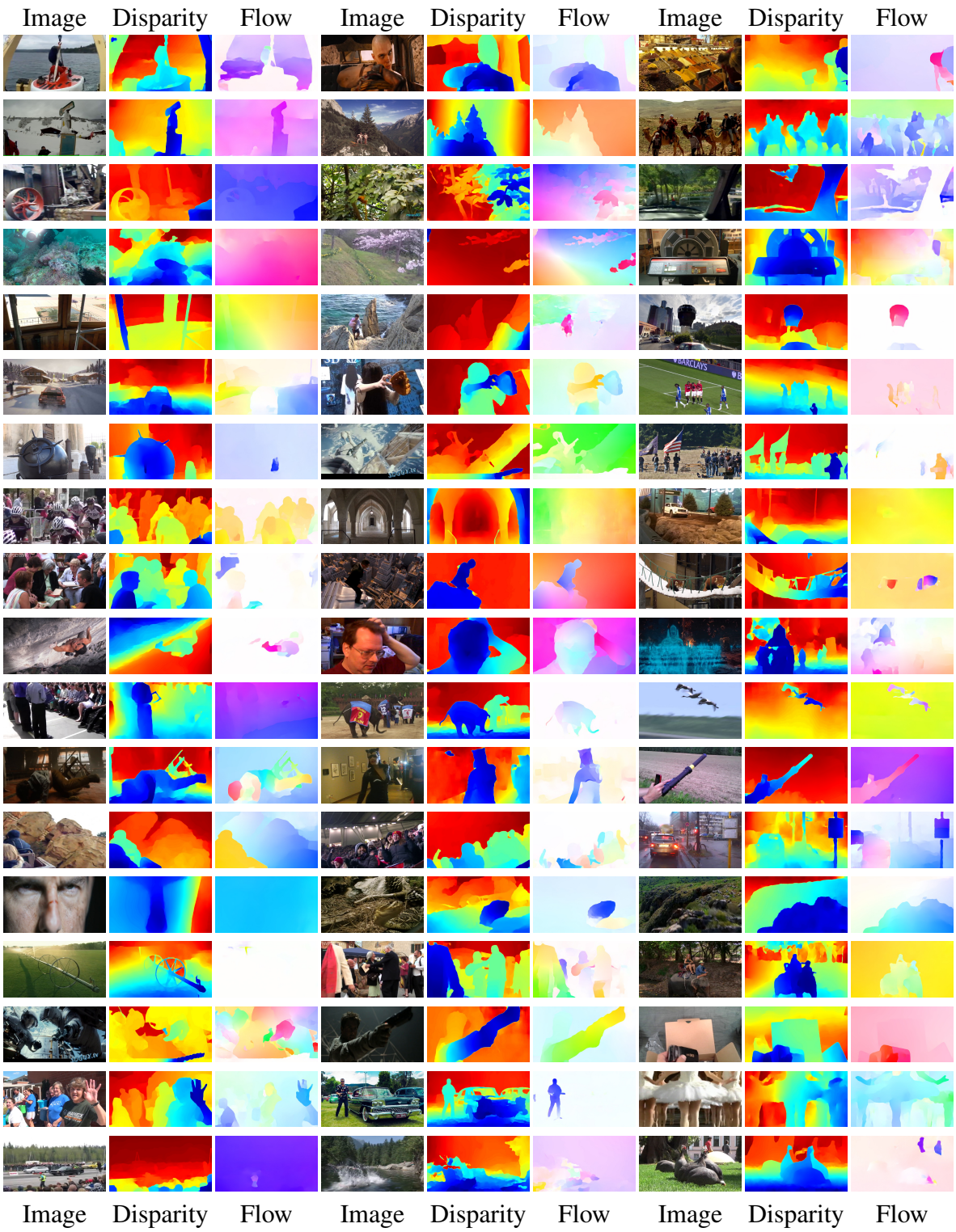



Figure 5.11: Selection of frames from WSVD showing diverse scenes, objects and motions.

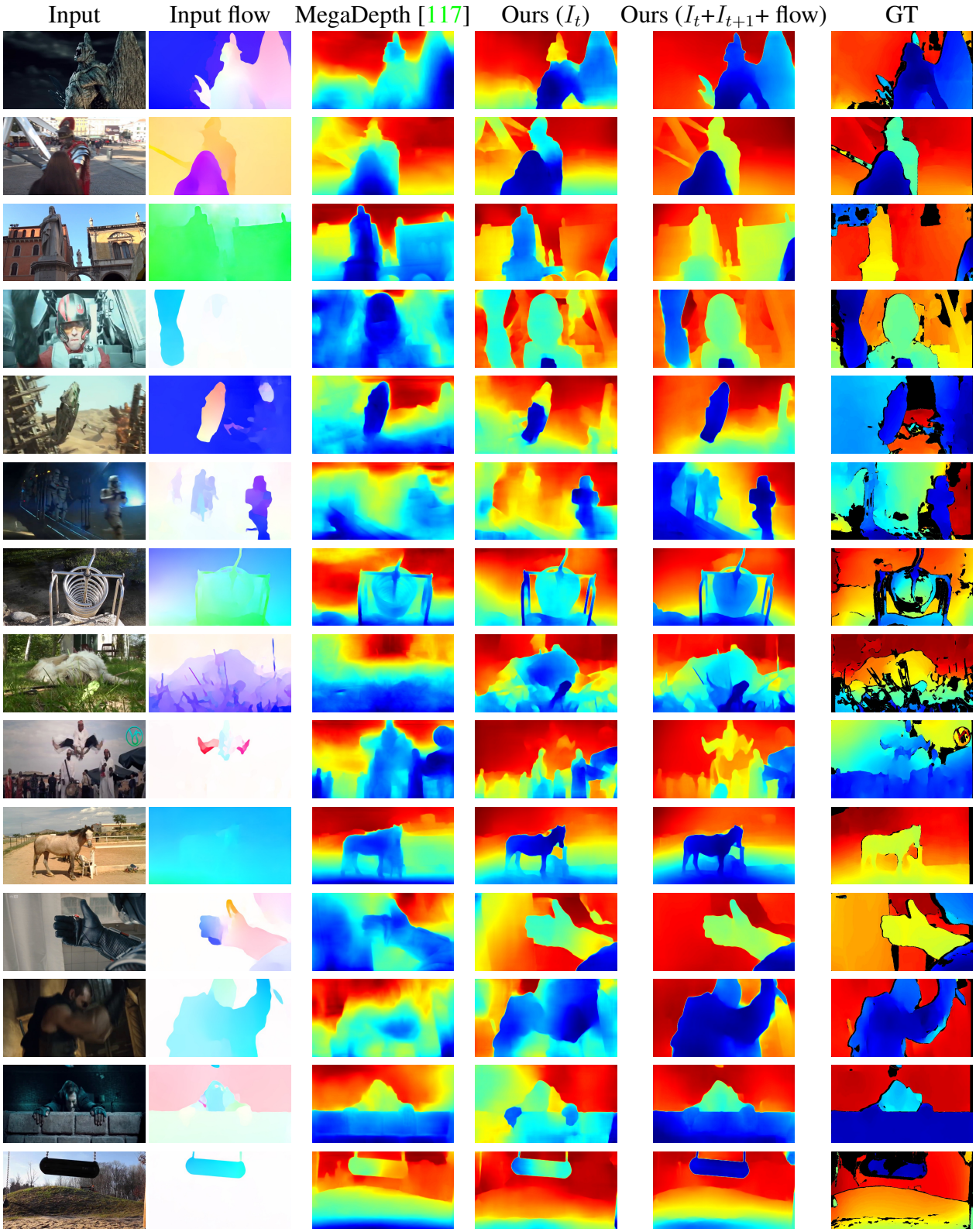





Figure 5.12: Qualitative results of comparing MegaDepth vs baseline monocular depth prediction vs Ours.

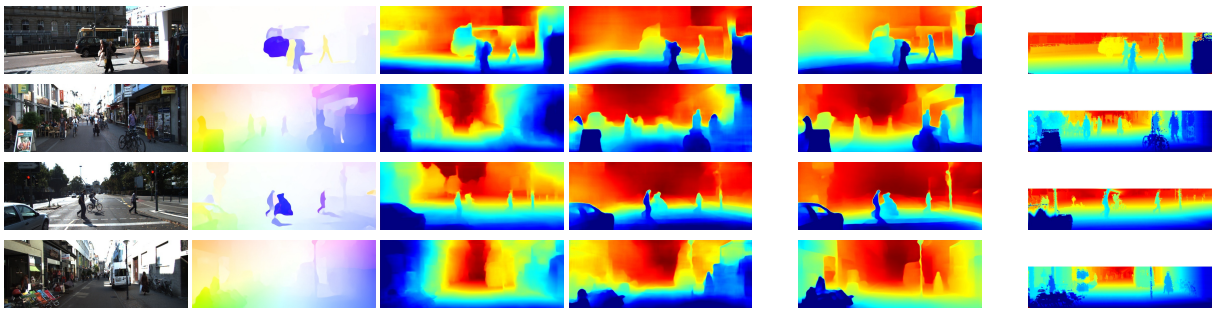



Figure 5.13: Qualitative results on KITTI

# Chapter 6

# Neural Trajectory Fields for Reconstructing Dynamic Scenes

## 6.1 Introduction

Novel view synthesis (NVS), the ability to create a photorealistic rendition of a scene from an unseen viewpoint, is a challenging, long-standing problem. After impressive early attempts [27, 41, 113], the computer vision and graphics communities made relatively slow progress towards a general and practical solution. That changed with the recent exploration of neural rendering [191], in particular the Neural Radiance Field (NeRF) [133], a coordinate-based neural representation for differentiable volumetric rendering which has since fueled a new wave of interest [121, 175, 190].

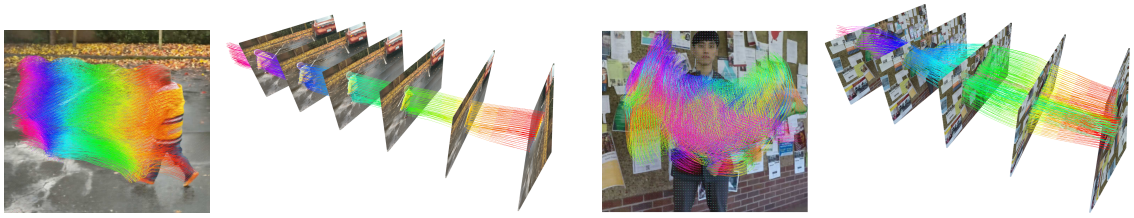


Figure 6.1: Neural trajectory fields define dense, parametric, spacetime trajectories that enable consistent novel view synthesis even in the presence of dynamic and complex motion. The trajectories shown here are estimated for points in the first view. Note how faithfully they follow the dynamic objects. Parametrized using the Discrete Cosine Transform (DCT), these trajectories allow us to extrapolate motion even beyond the last available frame. **To view the animations, please open this document in a media-enabled PDF viewer, such as Adobe Reader.**

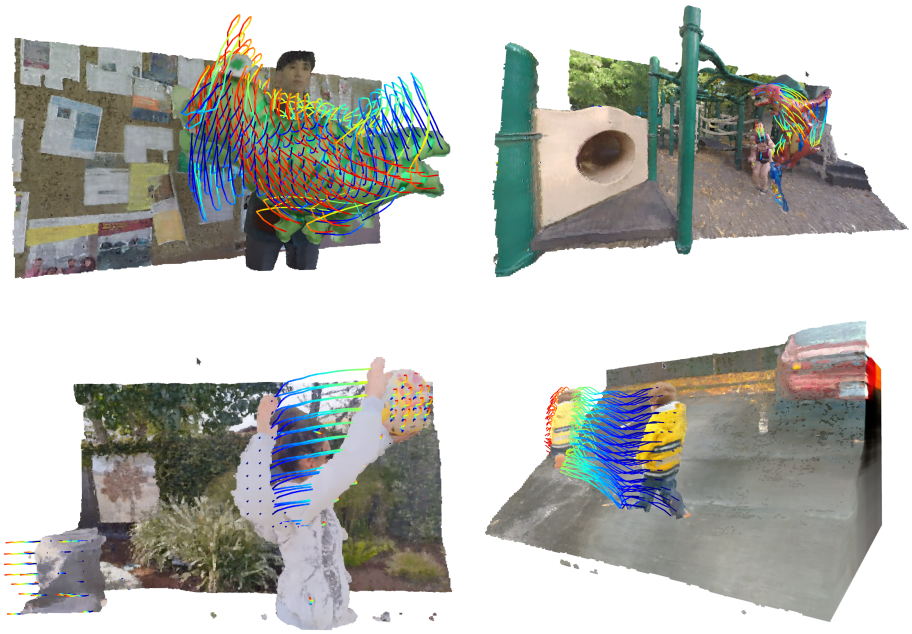


Figure 6.2: Our method predicts dense sequence-long trajectories for each point in the scene. Here we show examples of trajectories sampled at sparse locations. The 3D scenes are rendered by overlapping the radiance fields from two different frames of each sequence. The visualized 3D trajectories are directly estimated from the output of the first radiance field.

NeRF uses a multilayer perceptron (MLP), $\Psi$, to implicitly represent the geometry and radiance information of a *static* 3D scene. The color and transparency of a 3D point in space is estimated by querying the coordinates of the corresponding point and an associated viewing angle: $(\mathbf{c}, \sigma) = \Psi(\mathbf{p}, \mathbf{d})$. The network $\Psi$ is trained using a render-and-compare strategy, employing volumetric rendering from ray samples for view synthesis. NeRF makes an assumption common to most existing methods: the subject must be static so that the epipolar geometry constraints hold across views.

Generalizing NeRF for dynamic scenes captured from a single camera is challenging, as it requires disambiguating displacements due to parallax from those due to non-rigid motion. Recent pre-prints extend NeRF in one of two ways: they either parametrize time explicitly, *i.e.*, $(\mathbf{c}, \sigma) = \Psi(\mathbf{p}, t, \mathbf{d})$ [116, 226], or they learn a warping field $\phi$ that maps the query coordinates at time $t$ to coordinates in a canonical frame, *i.e.*, $(\mathbf{c}, \sigma) = \Psi\left(\phi(\mathbf{p}, t), \mathbf{d}\right)$ [159, 195]. When attempting to regularize the motion, these methods face a trade-off: broadly speaking, the more powerful the prior (*e.g.*, the subject is a portrait [52]), the more stringent the assumptions on the scene content. On the other hand, more general priors (*e.g.*, piece-wise linear motion [116]) tend to be weaker in enforcing consistency in dynamic regions.

This tension is not unique to NeRF, or even to NVS methods. A similar issue affects structure from motion (SfM) estimation. In that context, Akhter *et al*. argue that motion lives in space and time and can be tackled by using priors in either domain [10]. Motion, then, can be accounted for by using a combination of shape basis (space), which tend to be object-specific, or by using physical constraints on the trajectory (time), which are more object-agnostic [105, 201, 252]. Akhter *et al*. posit that using trajectories parameterized as a linear combination of sinusoidal bases (*e.g.*, DCT transform) offers a good compromise between motion regularization and generality of the priors.

Inspired by this observation, we propose DCT-NeRF, a new spatiotemporal, coordinate-based neural scene representation. In addition to color and transparency, DCT-NeRF outputs continuous 3D trajectories across the whole input sequence: $(\mathcal{T}, \mathbf{c}, \sigma) = \Psi(\mathbf{p}, t, \mathbf{d})$, where $\mathcal{T}$ is the discrete cosine transform trajectory of the point with spatiotemporal position $(\mathbf{p}, t)$. As can be seen in Figure 7.1, this representation produces physically plausible motions that stay consistent across all frames, rather than simply predicting warping fields between neighboring frames. Moreover, in contrast to existing works, the trajectory field provides arbitrarily long spatiotemporal correspondence predictions that allow us to establish a

principled regularization scheme.

Rendering views from significantly different times with our trajectory parametrization requires careful handling of two factors. First, we need to account for the potential change in reflectance of a point along its trajectory in both space and time. We do this by allowing the predicted color of a spacetime location to be a function of time. Second, we need to deal with regions that become occluded due to motion. Li *et al*. rely on a disocclusion mask learned for each pair of neighboring input images [116], but this approach does not scale when comparing any two frames in the sequence—the number of masks is combinatorial with respect to the sequence length. However, we note these problematic occlusion happen only when empty space at one time becomes occupied at another. This observation allows us to discard regions that may get occluded by dynamic objects simply by leveraging the transparency field $\sigma$, which our network already estimates for volumetric rendering.

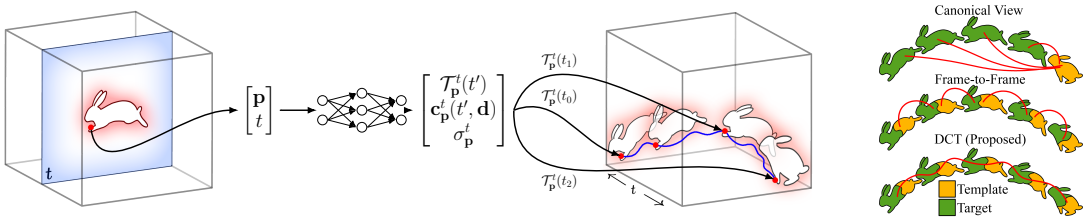
Figure 6.2 shows examples of reconstructed radiance fields and trajectories. Note that the visualized trajectories are estimated by querying the network *only at the time of the first frame*, but capture the motion of objects over the entire sequence of frames. Our method produces high-quality renderings of dynamic scenes from novel viewpoints, both numerically and visually.

## 6.2 Related Work

**Novel view synthesis.** Early attempts at rendering a scene from an unseen view point date back several decades [20, 27, 41, 66, 113]. In part due to novel deep learning techniques, we have observed a proliferation of methods in just the last few years, which can be categorized based on the way they represent the scene.

A traditional approach is to estimate depth explicitly and use it to warp pixels, or learned features, from the input images to the novel view [35, 87, 156, 168]. Such scene representations offer flexibility, but inherit the shortcomings of depth-estimation algorithms, and require ways to explicitly deal with disocclusions at rendering time. Thanks to advances in single-image depth estimation, novel views can now even be generated from just a single image [222].

A second class of methods uses multiplane images (MPI) [248]. These algorithms learn a representation in which objects in the scene are associated to fronto-parallel layers. At inference time MPIs are warped and fused into a novel view. This approach implicitly




(a) Method Overview (b) Types of Warping

Figure 6.3: **(a)** We estimate the transparency $\sigma^t_{\mathbf{p}}$ and color $\mathbf{c}^t_{\mathbf{p}}$ for each spacetime location $(\mathbf{p}, t)$ with viewing direction $\mathbf{d}$. $\mathbf{c}^t_{\mathbf{p}}$ is modeled as a temporal-view dependent function. To model the deformation of dynamic objects, we also predict the trajectory $\mathcal{T}^t_{\mathbf{p}}(\cdot)$ of any $(\mathbf{p}, t)$, which allows us to follow the point as it moves through space and time. We parametrize $\mathcal{T}^t_{\mathbf{p}}$ as a DCT trajectory. **(b)** Existing approaches either define a canonical volume and warp all the frames to that one, or only warp neighboring frames. Thanks to our trajectories, our method can warp information from any frame to any frame.

regularizes depth and optimizes the layers to render the novel view. As a result, MPI-based methods produce impressive NVS results [49, 180, 248], even when using a single image as an input [196], or a single semantic map instead of an RGB image [69, 79].

The third and perhaps most promising strategy is to build an implicit neural representation of the scene [32, 132]. In their pioneering work, Mildenhall *et al*. propose to train a multilayer perceptron (MLP) that can predict color and transparency for each point in space given a viewing direction [133]. In just over a year since its publication, NeRF has inspired a number of follow-up works that tackle issues such as improved quality [239], performance [120, 185], or compositionality [141, 164].

**Dynamic NVS.** The problem of NVS is even more challenging for dynamic scenes because input views across time may capture differing scene geometry. A dynamic scene can be "frozen" into a static one by using synchronized cameras [253]. Single-image depth estimation also effectively allows one to freeze a scene. Luo *et al*. use it to compensate for small motion [125], while Yoon *et al*. use it to learn a full 4D representation of the scene, which they use to perform spatio-temporal image synthesis [234].

A number of pre-prints that augment NeRF for dynamic content have also recently appeared in parallel to our proposed research, which we report here for completeness. Many make simplifying assumptions, such as the availability of a depth estimate [226], or that dynamic objects are distractors to remove [129]. Other works target specific applications, such as self-portraits [52, 149]. Two closely related works are the works of Li *et al*., who

use information from neighboring frames by learning a warping field [116], and Tretschk *et al*. who learn to warp input rays into a canonical volume [195].

The central idea for these methods, including ours, is that information must be warped between different time instants to combine information from multiple frames. Both Tretschk *et al*. [195] and Gao *et al*. [149] define a canonical space-time volume and warp the information from all the frames to this template, either by bending the light rays [195], or by applying a warping field to the values from the non-canonical frames [149]. Instead of defining a canonical volume, for each frame in the sequence Li *et al*. warp the information from the neighboring frames (*i.e*., $t \pm 1$, $t \pm 2$). In a way these are opposite extremes: the former two have a single "centralized" representation of the scene, where all the input frames are mapped to the same volume, and the latter can only gather and leverage local information. Our approach captures the advantages of both. Because we use trajectories, we can estimate dense correspondences and combine information from any two frames in the sequence (see Section 6.3.1). Figure 6.3(b) shows an illustration of the different strategies to combine dynamic information.

**Non-rigid structure from motion.** NRSfM concerns the problem of recovering deforming 3D shapes using 2D point correspondences from multiple images. Bregler *et al*. introduce this problem and propose the classical low-rank factorization approach[19]. Since then, two lines of work have emerged: shape-based and trajectory-based approaches. These two types of approaches are often complementary, and recent state-of-the-art employs both of them. Readers are referred to Jensen *et al*. [85] for a comprehensive evaluation of NRSfM methods. We focus our discussion on trajectory-based approaches for their relevance to our paper.

Torresani *et al*. utilize temporal information by imposing a linear motion model on top of a low-rank shape model, and show improvement on robustness against noise [194]. Since then, different trajectory priors have been explored. The seminal work of Akhteret *et al*. introduces a pure trajectory-based factorization method, which assumes that trajectories of different points in space can be represented as a linear combination of a small number of DCT trajectory bases [10]. Other methods have explored the convolutional structure of trajectories [252] as well as the assumption that different trajectories can be clustered into a small number of linear subspaces [105].

Taking inspiration from Akhteret *et al*., our approach represents continuous trajectories with DCT bases. A geometric analysis on how trajectories can be reconstructed using this

representation is given by Park *et al*. [148]. One limitation to the low-rank approach of Akhteret *et al*. is that the number of DCT bases needs to be tuned per sequence. Valmadre *et al*. later argue that minimizing the response of high-pass filters (*e.g*., first- and second-difference filters) on trajectories eliminates the need to tune the number of bases, and also leads to more stable result for long sequences [201]. Following this insight, we minimize the difference between neighboring time steps on the trajectories without tuning the number of bases.

**Occupancy flow.** Niemeyer *et al*. propose a temporally and spatially continuous vector field to represent velocities [142]. To get the displacement of a point between different time steps, they need to integrate over the flow fields w.r.t. time, which leads to the number of queries of the network being proportional to time. In comparison, our trajectory field can output displacements between any time steps using only a single network query.

## 6.3 Neural Trajectory Fields for Dynamic Novel View Synthesis

We build on NeRF, an implicit neural representation that predicts the color $\mathbf{c}$ and the transparency $\sigma$ of any scene point, given a viewing direction [133]: $(\mathbf{c}, \sigma) = \Psi(\mathbf{p}, \mathbf{d})$. NeRF estimates the color of a pixel in the desired view via volumetric rendering along the corresponding ray. Given a number of posed views of a scene, NeRF can be trained by rendering pixels and comparing them with the corresponding ground-truth pixels. This training strategy, however, requires the scene to be static, as a dynamic point $\mathbf{p}$ maps to different pixels in the input views.

To account for the point's motion, we propose to reconstruct its trajectory *across the entire sequence*. Inspired by classical works in the trajectory-based non-rigid structure from motion [10, 201], we parameterize the trajectory as a combination of sinusoidal bases, *i.e*., with a discrete cosine transform (DCT). A few observations are in order. First, a DCT trajectory is regular and smooth by construction, and has been shown to be a compact representation for the motion of deforming, dynamic objects [128, 241]. Second, all points in spacetime $(\mathbf{p}, t) \in \mathbb{R}^4$ are associated with a DCT trajectory $\mathcal{T}_{\mathbf{p}}^t(\cdot)$, and trajectories associated with any two spacetime points, *e.g*., $(\mathbf{p}_0, t_0)$ and $(\mathbf{p}_1, t_1)$, are expected to be the same if they correspond to the same physical point, *e.g*., the tip of the bunny's paw

shown in Figure 6.3. This can be enforced by penalizing the difference between $\mathcal{T}_{\mathbf{p}_0}^{t_0}$ and the trajectory associated with $(\mathcal{T}_{\mathbf{p}_0}^{t_0}(t_1), t_1)$ for any $\mathbf{p}_0$, $t_0$, and $t_1$. Finally, the ability to "move" the point in time following the trajectory allows us to compute the photometric error against the appropriate input pixels.

In the following we first formalize the description of our neural trajectory field, dubbed DCT-NeRF (see Section 6.3.1). We then show how the proposed representation is learned by enforcing photometric consistency through rendering with trajectories (see Section 6.3.2). The additional regularization terms we use are described in Section 6.3.3.

### 6.3.1 DCT-NeRF

Given a spacetime location $(\mathbf{p}, t)$, our neural field outputs the DCT coefficients (or frequency values) $\boldsymbol{\varphi}$ for the trajectory, the embedding $\boldsymbol{\omega}$ for the temporal-view-dependent RGB color, and the transparency value $\sigma$:

$$(\boldsymbol{\varphi}_{\mathbf{p}}^{t}, \boldsymbol{\omega}_{\mathbf{p}}^{t}, \sigma_{\mathbf{p}}^{t}) = \Psi(\mathbf{p}, t), \tag{6.1}$$

where superscript $t$ and subscript $\mathbf{p}$ are added to each output to denote that they are associated to the spacetime location $(\mathbf{p}, t)$. The outputs $\boldsymbol{\varphi}_{\mathbf{p}}^{t}$, $\boldsymbol{\omega}_{\mathbf{p}}^{t}$ are then used to condition the functional representation of the trajectory $\mathcal{T}_{\mathbf{p}}^{t} : \mathbb{R} \rightarrow \mathbb{R}^3$ and the color output $\mathbf{c}_{\mathbf{p}}^{t} : \mathbb{R} \times \mathbb{R}^3 \rightarrow \mathbb{R}^3$, *i.e.*,

$$\mathcal{T}_{\mathbf{p}}^{t}(t') = f_{\text{DCT}^{-1}}(t', \boldsymbol{\varphi}_{\mathbf{p}}^{t}), \qquad \mathbf{c}_{\mathbf{p}}^{t}(t', \mathbf{d}) = f_{\text{color}}(t', \mathbf{d}, \boldsymbol{\omega}_{\mathbf{p}}^{t}), \tag{6.2}$$

where $t'$ is the time input to the functions, $f_{\text{DCT}^{-1}}$ is the inverse DCT transform of the DCT coefficients $\boldsymbol{\varphi}$ (details see Section 6.3.1), and $\mathbf{c}_{\mathbf{p}}^{t}$ is modeled as a neural network $f_{\text{color}}$. The details of the network architecture are given in Section 6.3.4. We note that, unlike the original NeRF, our color output for a spacetime location $(\mathbf{p}, t)$ can vary over different time input $t'$ to account for specularities or illumination changes, even with fixed $\mathbf{d}$. This is crucial when enforcing photometric consistency across large temporal distance in Section 6.3.2. In Figure 6.6, we show animations of images rendered by $\mathbf{c}_{\mathbf{p}}^{t}(t', \mathbf{d})$ with varying time input $t'$.

**DCT Trajectory Representation**

Given a sequence of $T$ frames, and the DCT coefficients $\boldsymbol{\varphi} \in \mathbb{R}^{3K}$ ($K < T$) associated with the trajectory $\mathcal{T} = [x(t), y(t), z(t)]$, we can compute the motion along the $x$ axis as:

$$x(t) = \sqrt{2/T} \sum_{k=1}^{K} \varphi_{x,k} \cos\left(\frac{\pi}{2T}(2t+1)k\right), \tag{6.3}$$

where $\varphi_{x,k}$ denotes the $k$-th coefficient for the $x$ axis. The $y$ and $z$ components of the trajectory have the same form. Equation 6.3 discards the $1^{\text{st}}$ component (*i.e.*, $k = 0$) of the inverse DCT transform. We choose this formulation to remove the global offset, since we only care about the relative displacement between points on the trajectory, *i.e.* $\Delta\mathbf{p} = \mathcal{T}(t_1) - \mathcal{T}(t_0)$. In our experiments we set $K = T - 1$ for short sequences. For longer sequences, we use smaller $K$ for computational efficiency. Note that reducing the number of coefficients is a classical approach to enforce smoothness of the trajectory, but it requires knowing the complexity of the motion as *a priori*.

Training our network to directly predict the coefficients $\boldsymbol{\varphi}$ for each sampled spacetime location $(\mathbf{p}, t)$ offers a convenient mechanism to enforce a single trajectory for each object point: the coefficients $\boldsymbol{\varphi}$ predicted for the same point at time $t_0$ and at time $t_1$ should be the same. Moreover, parameterizing a single trajectory for a point allows to predict the location of that point at any time $t$, whether that time instant was observed or not, see Figures 7.1 and 6.2. This enables enforcing photometric consistency between any two frames, as discussed below.

### 6.3.2 Photometric Loss

**Frame-wise photometric consistency ($\mathcal{L}_{t_0}^{\text{photo}}$)**

Consider the color $\mathcal{C}^{t_0}$ of a pixel with ray direction $\mathbf{d}$ at time $t_0 \in \{0, 1, \cdots, T-1\}$. The contribution of a point along the ray $\mathbf{d}$ at position $\mathbf{p}$ to the color $\mathcal{C}^{t_0}$ on the camera plane is

$$\delta\mathcal{C}_{\mathbf{p}}^{t_0} = \sigma_{\mathbf{p}}^{t_0} \mathbf{c}_{\mathbf{p}}^{t_0}(t_0, \mathbf{d}), \tag{6.4}$$

where we make the dependence on $\mathbf{p}$, the sampled time $t_0$, and $\mathbf{d}$ explicit. To render the corresponding pixel's color, we swipe all the points along a ray, and perform volumetric

rendering:

$$\mathcal{C}^{t_0} = \int_{\xi_n}^{\xi_f} A(\xi)\delta\mathcal{C}^{t_0}_{\mathbf{o}+\xi\mathbf{d}}d\xi, \tag{6.5}$$

where $A(\xi) = \exp\big(-\int_{\xi_n}^{\xi}\sigma^{t_0}_{\mathbf{o}+\nu\mathbf{d}}d\nu\big)$ accounts for the attenuation along the ray, and $\mathbf{o}$ denotes the origin of the ray.
Given the color $\mathcal{C}^{t_0}_{\text{GT}}$ of a pixel from the input image at time $t_0$, we use the rendered color from Equation 6.5 to compute the photometric loss:

$$\mathcal{L}^{\text{photo}}_{t_0} = \left\|\mathcal{C}^{t_0} - \mathcal{C}^{t_0}_{\text{GT}}\right\|^2_2. \tag{6.6}$$

**Temporal photometric consistency ($\mathcal{L}^{\text{photo}}_{(t_0;t_1)}$)**

Because we also estimate trajectories, we can integrate information from any other times, say time $t_1 \in [0, T-1]$, in the same way that NeRF can compute the loss for any input frame *despite the scene being dynamic*. First, given a spacetime location $(\mathbf{p}, t_0)$, we compute its point correspondence at time $t_1$ as $\mathbf{p}' = \mathbf{p} + \mathcal{T}^{t_0}_{\mathbf{p}}(t_1) - \mathcal{T}^{t_0}_{\mathbf{p}}(t_0)$. We can then follow Equation 6.4 and 6.5 to render pixels at time $t_0$ using the warped radiance field from time $t_1$:

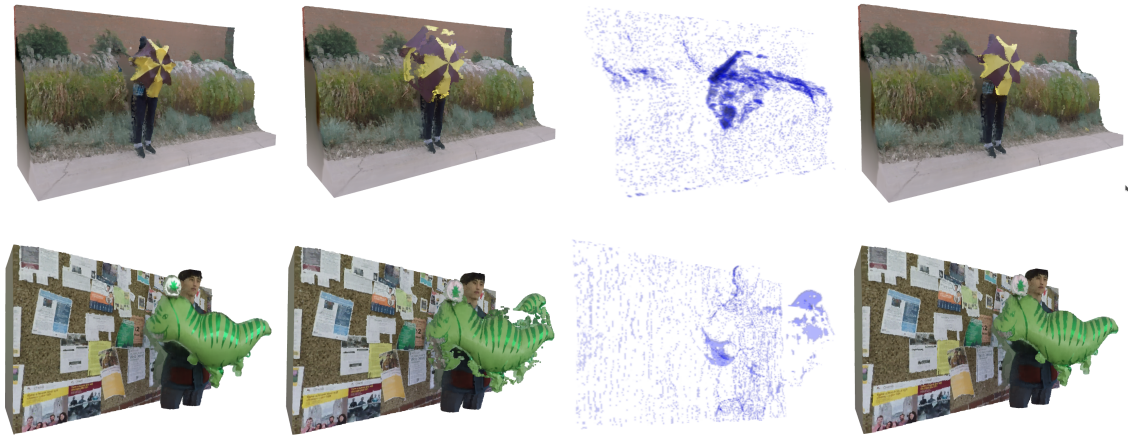
$$\delta\mathcal{C}^{(t_0;t_1)}_{\mathbf{p}} = \sigma^{t_1}_{\mathbf{p}'}\mathbf{c}^{t_1}_{\mathbf{p}'}(t_0, \mathbf{d}). \tag{6.7}$$

The final color $\mathcal{C}^{(t_0;t_1)}$ is computed with volumetric rendering by integrating $\delta\mathcal{C}^{(t_0;t_1)}_{\mathbf{p}}$ as in Equation 6.5. Temporal photometric consistency can then enforced by

$$\mathcal{L}^{\text{photo}}_{(t_0;t_1)} = \left\|\mathcal{C}^{(t_0;t_1)} - \mathcal{C}^{t_0}_{\text{GT}}\right\|^2_2. \tag{6.8}$$

**Handling Temporal Occlusions.** Equation 6.7 does not account for temporal occlusions. Note that these regions are different from the typical spatial occlusions induced by the camera displacement. Consider a point $\mathbf{p}$ sampled in an empty area of the scene at time $t_0$, in other words $\sigma^{t_0}_{\mathbf{p}} = 0$ and consequently $\delta\mathcal{C}^{t_0}_{\mathbf{p}} = 0$. It is also expected that $\mathcal{T}^{t_0}_{\mathbf{p}}(t_1) = 0, \forall t_1$ since empty regions do not move, and thus $\mathbf{p}' = \mathbf{p}$. Then suppose at some time $t_1$ an object moves and occupies position $\mathbf{p}$, which implies $\sigma^{t_1}_{\mathbf{p}'} = \sigma^{t_1}_{\mathbf{p}} \neq 0$. Applying Equation 6.7 to this scenario yields

$$\delta\mathcal{C}^{(t_0;t_1)}_{\mathbf{p}} = \sigma^{t_1}_{\mathbf{p}}\mathbf{c}^{t_1}_{\mathbf{p}}(t_0, \mathbf{d}) \approx \delta\mathcal{C}^{t_1}_{\mathbf{p}} \not\approx \delta\mathcal{C}^{t_0}_{\mathbf{p}}. \tag{6.9}$$



Rend. at $t_0$ | Rend. w/o $p_{\text{occ}}$ | $p_{\text{occ}}$ | Rend. w/ $p_{\text{occ}}$

Figure 6.4: Space that is empty at $t_0$ and becomes occupied at $t_1$ causes rendering issues (Rend. w/o $p_{\text{occ}}$). We predict the probability of a region to cause such issues ($p_{\text{occ}}$) and downweight their contribution to the rendering process accordingly (Rend. w/o $p_{\text{occ}}$).

That is, even though $(\mathbf{p}, t_0)$'s trajectory is correctly estimated (zero motion), we use the wrong color when integrating Equation 6.5. One could identify and exclude these regions by learning a mask, as suggested by Li *et al*. [116]. However, comparing any two time stamps in the input sequence, as our method does, requires a combinatorial number of such maps. We observe that, barring interpenetration of objects, and neglecting objects that only appear in a subset of frames, this issue only manifests in space that is empty at $t_0$ and becomes occupied at another time $t_1$. Conveniently, our network predicts $\sigma$, which we use to estimate a probability of "empty-ness":

$$p_{\text{e}}(\mathbf{p}, t) = \text{sigmoid}(-k_1 \sigma^t_{\mathbf{p}} + k_2), \tag{6.10}$$

where we empirically set $k_1 = 2$, and $k_2 = 3$. Treating $k_1$ and $k_2$ as learnable parameters results in equal or degraded quality. The probability that empty space at $(\mathbf{p}, t_0)$ becomes occluded at time $t_1$ can then be estimated as

$$p_{\text{occ}}(\mathbf{p}, t_0, t_1) = p_{\text{e}}(\mathbf{p}, t_0)\big(1 - p_e(\mathbf{p}', t_1)\big), \tag{6.11}$$

which we can use to modify Equation 6.7 to downweigh the contribution of occluded

regions:

$$\delta\mathcal{C}_{\mathbf{p}}^{(t_0;t_1)} = p_{\text{occ}}(\mathbf{p}, t_0, t_1) \cdot \delta\mathcal{C}_{\mathbf{p}}^{t_0} + (1 - p_{\text{occ}}(\mathbf{p}, t_0, t_1)) \cdot \sigma_{\mathbf{p}'}^{t_1}\mathbf{c}_{\mathbf{p}'}^{t_1}(t_0, \mathbf{d}). \quad (6.12)$$

Figure 6.4 shows a rendering of a frame using the warped radiance field from a different time, with and without $p_{\text{occ}}$ to handle occlusions. It also shows the probability $p_{\text{occ}}$ over the whole space.

### 6.3.3 Regularization

To encourage our system to converge to the correct solution, we follow the common practice of adding regularizers to our optimization.

**Cycle consistency ($\mathcal{L}^{\text{cycle}}$).** To enforce the network to make temporally consistent estimates, we add an $L_1$ loss on the difference of $(\mathcal{T}, \mathbf{c}, \sigma)$ for corresponding points, where $\mathcal{T}$ is represented by the DCT coefficients and the value of $\mathbf{c}$ is queried with the same $(\mathbf{d}, t)$ input. To account for disocclusions, we weigh $\mathcal{L}^{\text{cycle}}$ by $1 - p_{\text{occ}}$.

**Single visible surface ($\mathcal{L}^{\text{SVS}}$).** We want to encourage the source of radiance to be concentrated on a single surface area visible to the current view point. This means that most of the energy of the attenuation coefficient $A(\xi)$ in Equation 6.5 along the ray should be concentrated in a small window of size $r$:

$$\mathcal{L}^{\text{SVS}} = 1 - \max_{z_0} \int_{z=z_0}^{z=z_0+r} \tilde{A}(z)dz, \quad (6.13)$$

where $\tilde{A}$ denotes that $A$ is normalized to sum to one along the ray.

**Trajectory regularizers ($\mathcal{L}^{\text{traj}}$).** Our trajectory field is regularized by the combination of the following terms: (i) the spatial smoothness term, which calculates the $L_1$ norm of the spatial derivative of scene flow; (ii) an as-rigid-as-possible deformation loss [104], which is used to encourage the local deformation in occupied regions (measured by $p_{\text{e}}$) to be isometric; (iii) the temporal smoothness term which penalizes the $L_1$ norm of the scene flow.

**Depth & flow.** We use precomputed optical flow [188] to provide local supervision for the projection of $\mathcal{T}$ between neighboring frames. We also employ single-image depth

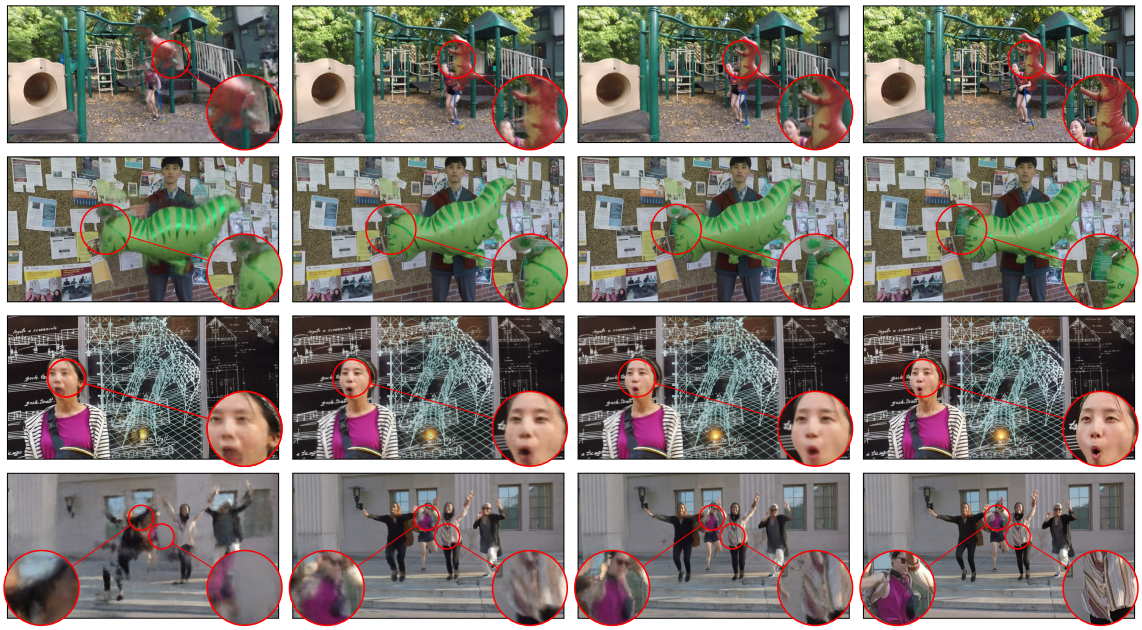



Figure 6.5: Comparisons. Our trajectory-based representation allows our method to perform well—particularly in dynamic regions.

estimation [161] to regularize $\sigma$ with a loss invariant to affine transformations [161, 213].

### 6.3.4 Implementation

**Optimization objective.** Given a pixel sampled at time $t_0$, we compute the loss as a combination of photometric losses and regularization terms. The photometric loss is evaluated as the sum of $\mathcal{L}^{\text{photo}}_{t_0} + \mathcal{L}^{\text{photo}}_{(t_0;t_0\pm1)} + \mathcal{L}^{\text{photo}}_{(t_0;t_1)}$, which enforces the rendering consistency between neighboring time steps, as well as a randomly sampled time step $t_1$. We employ a local-to-global strategy to sample $t_1$. That is, during the initial optimization iterations, $t_1$ is limited to $[t_0 - 2, t_0 + 2]$. We then exponentially increase the temporal radius during optimization, until the sample range of $t_1$ covers the whole sequence.

We find that adjusting the weights for certain regularization terms during the different stages of the optimization is critical for their success. First, when the weights for the depth and optical flow loss are not linearly decreased, as proposed by Li *et al*. [116], the network may be forced to overfit to the noise in the pre-computed depth and optical flow maps. Second, gradually increasing the weight of $\mathcal{L}^{\text{svs}}$ from a small value, *e.g.*, $1e-5$, helps to

Figure 6.6: For photometric consistency across time in the presence of dynamic specularities and shadows, our approach allows the predicted radiance to be rendered with respect to any other reference time across the sequence. Given the leftmost column's time in the sequence as a reference, the rightmost two columns show two different points in spacetime but rendered using the predicted radiance values of the reference time. **To view this animation, please open this document in a media-enabled PDF viewer, such as Adobe Reader.**

prevent getting stuck in poor local minima in the early stage of training. We use the same hyperparameters for all our experiments, and the details are included in the supplementary.

**Static background.** The consistency of the background across time is crucial for the viewing experience of dynamic NVS. Martin-Brualla *et al*. show success in learning a rigid NeRF for static background from images with dynamic objects [129]. Their approach is to modify the rigid NeRF model to output an additional 3D blending field so as to exclude interference from the dynamic regions of input images. Similarly, Li *et al*. [116] also include a rigid NeRF and blend the rigid and dynamic radiance fields to render an additional image when evaluating the photometric loss. We follow the approach of Li *et al*. and observe an improvement in the stability of the background.

**Network architecture.** We adapt the architecture from the original NeRF [133] implementation, with the following changes: (i) we concatenate an additional positional encoding of time to the input; (ii) parallel to the linear layers that predict $\sigma$, we add another linear layer to output the DCT coefficients; (iii) we concatenate the directional inputs with the positional encoding of time when calculating the color.

## 6.4 Evaluation and Results

**Datasets.** We use eight diverse scenes from the NVIDIA Dynamic Scene Dataset [234] to validate our method with both numerical and visual comparisons. For each scene, the dataset provides 12 sequences captured by synchronized cameras at fixed positions. Then camera motion is synthesized by choosing frames from different cameras at each time step.

We extract 24 frames per scene as input sequences for training, and report results on the held-out 11 images for each frame.

**Trajectories.** Figure 7.1 shows the trajectory estimated at the time of the first frame traced out over the whole duration of the sequence. Notice that our trajectories are stable and smooth across the entire sequences, even when the motion changes direction, as is the case of the scene on the right. The trajectories can be sampled as densely or as sparsely as needed. Figure 6.2 shows trajectories for the same sequences of Figure 7.1 and two more overlaid to a 3D visualization of the radiance field computed by our method.

**Dynamic specularities and shadows.** Because we allow the color $\mathbf{c}$ to vary with time, we can render one frame with the radiance of another. Figure 6.6 shows an example of this (please view the animation in a media-enabled reader). In this experiment, we apply the radiance of a reference frame (the leftmost in the figure) to two different target frames, animated over the entire sequence. Notice how the dynamically changing specularities and shadows correctly correspond to the reference image. This is possible because our method correctly estimates both radiance and trajectories over time.

**Ablation study.** We analyze the impact of the regularization terms we describe in Section 6.3.3 on the overall quality of the results. We use our model trained without the static background loss as a baseline. Table 6.1 shows the numerical results when using only the neighboring frames for $\mathcal{L}^{\text{photo}}$ ("local only"), removing the occlusion weights ($p_{\text{occ}}$), and removing other regularization terms one at a time. Through our numerical analysis, as well as a thorough visual inspection of the results, we observe that these terms are statistically beneficial. Removing any of them from our formulation introduces *some* artifacts in *some* of the results. However, they are not central to our results. We refer the reader to the supplementary material for visual examples.

**Comparisons.** The two most closely related methods are the method by Li *et al*. [116] and that of Tretschk *et al*. [195]. Both of these methods are pre-prints and we compare against them to offer context to the reader.

Table 6.1 shows a numerical comparison against these two methods, and two more [125, 176] whose numbers were reported by Li *et al*. [116]. The numbers show that we perform significantly better than the first three methods in the table. We perform roughly on par with Li *et al*., with their method performing slightly better overall, and ours performing slightly better on the dynamic regions. This is consistent with the fact that our main contribution is a global representation of the motion of the dynamic regions of the scene. We offer a

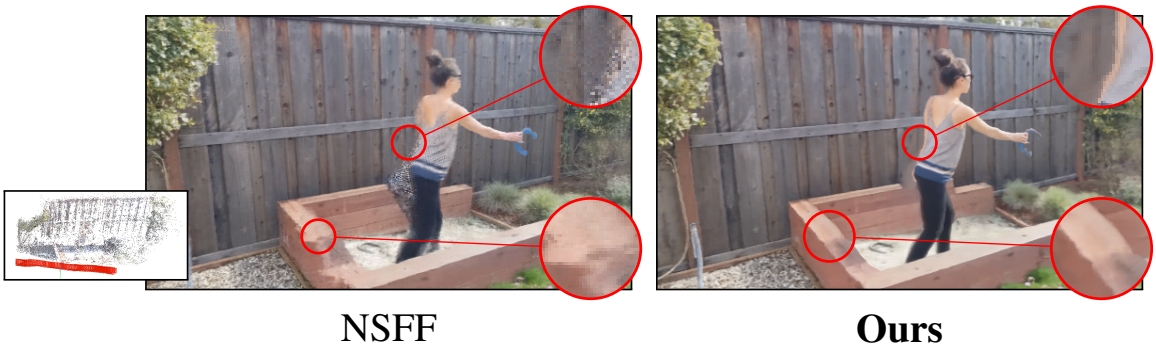



Figure 6.7: Comparison with Li *et al*. [116] on a challenging NVS case where we render the view on one end of the trajectory using the camera on the opposite end. The cameras are shown in the inset. Our method produces fewer artifacts in particular in the temporal occlusion regions.

per-sequence breakdown of the comparison with the two most closely related methods in Table 6.2.

Figure 6.5 shows visual comparisons with Li *et al*. and Tretschk *et al*. Note how our method produces sharper results in many of the dynamic regions. We also show a comparison with the method by Li *et al*. on two additional sequences in Figure 6.7. These are two challenging cases in which the rendered view is at the location of the camera at one end of the capture sequence, but the values are from the time of the camera at the opposite end of the sequence. Thanks to the fact that we model long-term trajectories, we can better reconstruct the scene.

Another approach for NVS of dynamic scenes is the method by Yoon *et al*. [234]. Note that their method requires a mask for the dynamic regions to be given. As their code is not publicly available we compare against their work by selecting a few frames from their own videos and synthesize a reasonably close view in space and time with our method. Their results are overall sharper, which is consistent with the general observation that NeRF-based methods can be a bit blurrier. However, several artifacts can be seen in their results around depth discontinuity regions.

Some of the images we use in the paper are courtesy of Li *et al*. [116] and the Nvidia Dynamic Scene Dataset [234].

| Method | Full scene | | Dynamic parts only | |
|---|---|---|---|---|
| | SSIM (↑) | LPIPS (↓) | SSIM (↑) | LPIPS (↓) |
| 3D Photo [176] | 0.614 | 0.215 | 0.486 | 0.217 |
| Luo *et al*. [125] | 0.746 | 0.141 | 0.530 | 0.207 |
| NR-Nerf [195] | 0.526 | 0.307 | 0.40 | 0.400 |
| NSFF [116] | 0.928 | 0.045 | 0.758 | 0.097 |
| Ours w/o static | 0.885 | 0.077 | 0.701 | 0.092 |
| - local only | 0.878 | 0.087 | 0.689 | 0.103 |
| - w/o $p_{\text{occ}}$ | 0.889 | 0.078 | 0.707 | 0.097 |
| - w/o $\mathcal{L}^{\text{cycle}}$ | 0.881 | 0.082 | 0.718 | 0.103 |
| - w/o $\mathcal{L}^{\text{svs}}$ | 0.885 | 0.082 | 0.711 | 0.096 |
| - w/o $\mathcal{L}^{\text{traj}}$ | 0.879 | 0.085 | 0.701 | 0.101 |
| - w/o $\mathcal{L}^{\text{depth}}$ | 0.892 | 0.081 | 0.697 | 0.108 |
| Ours (w static) | 0.915 | 0.049 | 0.704 | 0.089 |

Table 6.1: Numerical results and ablation study for our method. We perform better than existing methods and on par with the concurrent method by Li *et al*. [116].

| Sequence | NR-NeRF [195] full / dyn. | NSFF [116] full / dyn. | Ours full / dyn. |
|---|---|---|---|
| Balloon1 | 0.278 /0.398 | 0.062/0.175 | **0.051/ 0.146** |
| Balloon2 | 0.301 /0.395 | **0.030/0.065** | 0.054/ 0.074 |
| DynamicFace | 0.164 /0.175 | 0.025/0.034 | **0.021/ 0.022** |
| Jumping | 0.415 /0.518 | **0.046/0.111** | 0.056/ 0.116 |
| Playground | 0.363 /0.467 | 0.066/0.124 | **0.055/ 0.092** |
| Skating | 0.349 /0.575 | **0.017/0.061** | 0.036/ 0.088 |
| Truck | 0.268 /0.346 | 0.026/0.052 | **0.025/ 0.036** |
| Umbrella | 0.315 /0.322 | **0.089**/0.157 | 0.091/ **0.134** |

Table 6.2: Breakdown of LPIPS metric per scene from the Nvidia Dynamic Scene Dataset.

Yoon *et al*. [234] **Ours**

Figure 6.8: The code from Yoon *et al*. is not available. We estimate the location and time of a frame from their results and synthesize it with our method. Note the artifacts at disocclusion regions, *e.g*., the front-most green pole in the top row, or the right side of the woman in the bottom.

Figure 6.9: Failure cases for our method. Top: confuse background as moving region. Bottom: artifacts due to fast motion.

## 6.5 Appendix

### 6.5.1 Implementation details

We use 10 frequencies for the spatial-temporal positional encoding, and use 4 frequencies for the directional and temporal input to the color output layer. Both the radiance field and the DCT trajectories are defined in normalized device coordinates (NDC). We linearly sample 128 depth values along the rays when performing volumetric rendering. Due to the intense computational cost of the current method, we do not perform additional depth sampling and train an extra "fine" network as done by Mildenhall *et al*. [133].

#### Regularization details

**$\mathcal{L}^{\text{cycle}}$ details.** Given a point $\mathbf{p}_{t_0}$ at time $t_0$, by query the DCT trajectory $\mathcal{T}^{t_0}_{\mathbf{p}_{t_0}}$ outputted by $\Psi(\mathbf{p}_{t_0}, t_0)$, we get its correspondences at time $t_1$, *i.e.* $\mathbf{p}_{t_1} = \mathcal{T}^{t_0}_{\mathbf{p}_{t_0}}(t_1)$. Denote the occupancy, color and DCT coefficients associated to the points $\mathbf{p}_{t_i}$ as $(\sigma_{t_i}, \mathbf{c}_{t_i}, \boldsymbol{\varphi}_{t_i})$, and with particular note that $\mathbf{c}_{t_0} = \mathbf{c}[\mathbf{p}_{t_0}, t_0](t_0)$ and $\mathbf{c}_{t_1} = \mathbf{c}[\mathbf{p}_{t_1}, t_1](t_0)$, the cycle consistency loss is evaluated

as:

$$\mathcal{L}^{\text{cycle}} = (1 - p_{\text{occ}})(\|\boldsymbol{\varphi}_{t_0} - \boldsymbol{\varphi}_{t_1}\|_1 + 0.1\|\sigma_{t_0} - \sigma_{t_1}\|_1 + 0.1\|\mathbf{c}_{t_0} - \mathbf{c}_{t_1}\|_1), \quad (6.14)$$

where $p_{\text{occ}}$ is the probability of $\mathbf{p}_{t_0}$ being disoccluded by $\mathbf{p}_{t_1}$ as an empty region.

**$\mathcal{L}^{\text{traj}}$ details.** Given neighboring points $\mathbf{p}_{t_0}$ and $\mathbf{p}'_{t_0}$ (converted from NDC to Euclidean coordinates) along a ray at time $t_0$, the trajectory regularization loss is evaluated as:

$$\begin{aligned}\mathcal{L}^{\text{traj}} = &\underbrace{\|(\mathbf{p}_{t_0} - \mathbf{p}_{t_0\pm1}) - (\mathbf{p}'_{t_0} - \mathbf{p}'_{t_0\pm1})\|_1}_{\text{spatial smoothness of scene flow.}} \\ &+ \underbrace{(1 - p_{\text{e}})(\|(\mathbf{p}_{t_0} - \mathbf{p}_{t_0})' - (\mathbf{p}_{t_1} - \mathbf{p}'_{t_1})\|_1)}_{\text{as-rigid-as-possible.}} \\ &+ \underbrace{\|(\mathbf{p}_{t_0} - \mathbf{p}_{t_0\pm1}) - (\mathbf{p}'_{t_0} - \mathbf{p}'_{t_0\pm1})\|_1}_{\text{temporal smoothness of trajectory / small scene flow.}} \cdot\end{aligned} \quad (6.15)$$

**Hyperparameters.** The full regularization loss is linearly combined as:

$$\mathcal{L}^{\text{cycle}} + 0.1\mathcal{L}^{\text{traj}} + w_{\text{SVS}}\mathcal{L}^{\text{SVS}} + w_{\text{depth}}\mathcal{L}^{\text{depth}} + w_{\text{flow}}\mathcal{L}^{\text{flow}}, \quad (6.16)$$

where the initial $w_{\text{depth}}$ and $w_{\text{flow}}$ are set as 0.04 and 0.02, and then decreased by a factor of 0.1 every 7 epochs. $w_{\text{SVS}}$ is increased by a factor of 10 every 7 epochs from the initial value of $1e-5$, until it reaches $1e-2$.

### Optimization details

We use Adam optimizer with default parameters, and set the learning rate as $5e-4$ for the first 70 epochs, then decrease the learning rate by 0.1 for the next 10 epochs for fine-tuning. For the temporal local-to-global procedure, we increase the temporal radius by 2 times every 10 epochs.

We use a batch size of $1024 + 512$ during training, with the first 1024 pixels uniformly sampled from training images, while the last 512 pixels are sampled uniformly only from the regions marked by a motion segmentation method. Sampling extra pixels from moving regions is also done by Li *et al*. [116], and we note that this is helpful to speed up

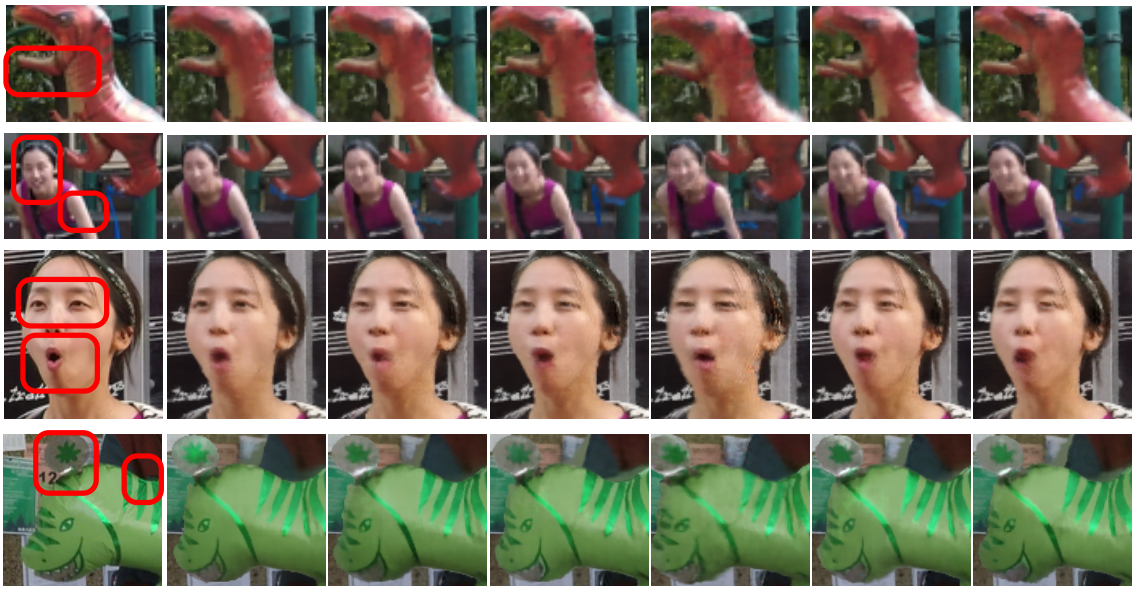



Figure 6.10: Visual comparison for ablation studies. The regions which show major difference between different versions of the method are marked with red rectangles.

convergence for small non-rigid regions, but not crucial to the end result of our method.

### 6.5.2 Visual comparison for the ablation study

The visual comparison of different versions of the method for the ablation study is provided in Figure 6.10.

## 6.6 Challenges to investigate

Dynamic NVS from single camera is still a very challenging task, and the quality of our result is sensitive to the context and motion of the input scene. We list the following limitations of our work which we are considering to address.

### 6.6.1 Tracking failure

Our method estimates long-term trajectories by integrating pairwise optical flows and minimizing photometric rendering loss. However, the robustness of such approach is limited

under challenging scenarios such as fast moving thin objects and occlusions. Figure 6.9 shows two typical examples of wrong reconstruction due to tracking failures. Tracking failures is especially common due to self-occlusion and disocclusion. For example, for a video capturing a side view of human sliding her both arms. One of her arm is periodically occluded or disoccluded by her body and the other arm. Our current method tends to have confusion in differentiating the arms from each other, and consequently leads to wrong estimate of trajectories as well as reconstructions.

There has been many attempts in literature to address such issues, but mostly from the perspective of body pose tracking with learned trackers or body pose estimators [67]. Recent works by Yang *et al*. [230, 231] is able to perform self-supervised tracking and reconstruction from monocular videos and achieve impressive results. But their method is mostly design for a single articulated object and relies on image segmentation, pose estimator and embedding layers pretrained for several animal categories. This prevents their success on our problem setup where we care about holistic reconstruction of both the foreground and background, and we do not want to put too much restriction on the category of objects we can work with.

We find a recent work on particle videos by Harley *et al*. [72] potentially can be relevant to our problem. They show particles can be tracked through occlusions with a RAFT-like neural network. The potential benefit of their work is that it is a generic long-term tracking method without assuming object categories or scene types. We would be interested to see if the occlusion handling ability learned from their customized sythetic dataset can be used to improve our method.

### 6.6.2 More efficient deformation representation

We currently represent deformation using neural trajectory field. While trajectory field is advantageous compared to scene flow fields, it is unclear how it compares to other typical deformation representation such as soft blend skinning and se(3) deformation field [149]. In our initial experiment with the code from Nerfies [149], we find that se(3) deformation field is not suitable to constrain large and rapid motion. On the other hand, it seems soft blend skinning can be a viable choice, and it is currently widely used for articulated object modeling. We will investigate how to apply soft blend skinning for scene reconstruction, and we foresee a hybrid of different deformation model can be an interesting topic to

research.

### 6.6.3 Sensitive to input camera poses

A lot of struggles in getting NeRF works on real data is to get accurate camera poses. However this can be challenging or laborious at least for dealing with dynamic scenes. Directly running SfM systems (*e.g*. colmap) over images with significant dynamic regions would result in failure or noisy camera estimation. The common practice in literature is to use either semantic segmentation or motion segmentation to exclude dynamic regions in input images before feeding into the SfM pipeline. This however would require the user to manually specify which object categories are considered as foregrounds and motion segmentation can also be erroneous. Another challenge in real world videos is that many background regions are low texture, thus even if the motion segmentation is successful, the camera pose estimation can still be unreliable in some cases.

Motivated by recent works [119, 221] which show that camera poses can be jointly optimized with neural radiance field, we would like to see if we can also achieve this for dynamic NeRF. We note that joint estimation of camera poses and object motions have been extensively investigated in non-rigid structure from motion and scene flow communities. We will review the relevant literature and draw some inspirations.

# Chapter 7

# Flow supervision for Deformable NeRF

## 7.1 Introduction

Reconstructing dynamic scenes from monocular videos is a significantly more challenging task compared to its static-scene counterparts, due to lack of epipolar constraints for finding correspondences and ambiguities between motion and structure. Recent advances in differentiable rendering have lead to various solutions using an analysis-by-synthesis strategy – solving the non-rigid deformation and structure by minimizing the difference between synthesized images and input video frames. Among those, deformable neural radiance fields [123, 149, 159, 195] has been a notable technique to represent dynamic scenes and shows plausible space-time view synthesis results. However, the current implementations only warrant success on teleporting-like videos whose camera motions are significantly more rapid than object motions. Quality of their results significantly decrease on videos with more rapid object motions [53].

In this work, we conjecture the deficiency of these deformable NeRF-based methods is mainly due to lack of temporal regularization. As they represent deformation as *backward* warping from the sampled frames to some canonical space, the motions or scene flows between temporally adjacent frames is not directly modeled nor supervised. Another deficiency is that these methods minimize photometric error alone, which is insufficient for gradient descent to overcome poor local minima when the canonical and other frames has little spatial overlap. This deficiency is severe for non-object-centric videos with large translations.

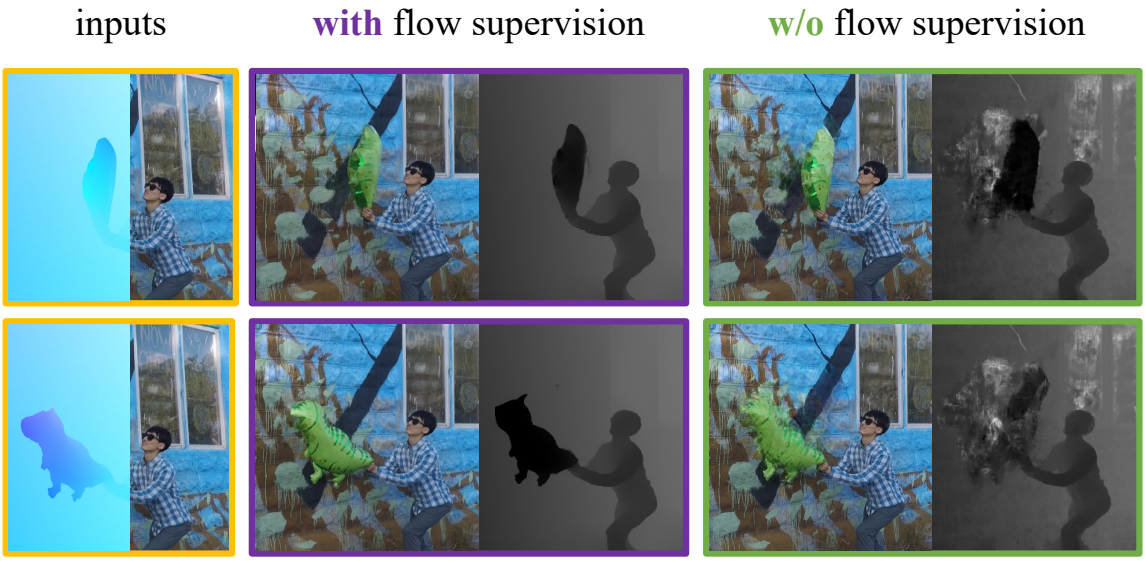



Figure 7.1: We propose a method to use optical flow supervision for deformable NeRF. It noticeably improves novel view synthesis for monocular videos with rapid object motions. In the figure, we visualize rendered novel view images and depth maps for the first and last frame of the input video.

The community has explored optical flow as an additional cue to help supervise the temporal transitions of other motion representations, such as scene flow fields [46, 51, 116, 242] and blend skinning fields [231]. However, enforcing flow constraints with respect to a generic *backward* warping field as in Nerfies [149] is non-trivial. Intuitively, to compute scene flows, it requires inverting the *backward* warp by having a *forward* warp which maps points from canonical space to other frames. Then scene flows can be computed either by taking time derivative of the forward warp or predicting the next position of a point through evaluating the forward warp. But this can be problematic since analytical inverse of a complicated non-bijective function (*e.g.* neural networks) is impossible, and an approximate solution by having an auxiliary network to represent the forward warp will introduce computational overhead and is theoretically ill-posed. Counter to this intuition, we will show that inverting the backward warping function is actually not needed for computing scene flows between frames.

The main **contribution** of this paper is: we derive an analytical solution to compute velocities of objects directly from the *backward* warping field of the deformable NeRF. The velocities are then used to compute scene flows through temporal integration, which

allows us to supervise the deformable NeRF through optical flow. This leads to significant improvement on videos with more rapid motions, as shown in Fig. 7.1.

The advantage of our approach is twofold: (i) Our method applies to all kinds of backward warping function, thanks to the weak assumptions required by the inverse function theorem which our derivation is based on. Thus our method is more general compared to other works using invertible normalizing flows [112] or blend skinning [30, 231]. (ii) Our method is also computationally more tractable compared to neural scene flow fields [46, 114, 116], which would require integrating flows over long period of time to reach some canonical frame.

## 7.2 Related works

**Deformable NeRF.** One common approach to model the dynamic scene is to represent it as the deformation of a static unknown template, and reconstruction is then done by fitting the model to the input 2D observations. Such approach has been implemented using techniques from recent advances in differentiable rendering. Most noticeably is deformable neural radiance field [123, 149, 159, 195], which models the deformation and the template radiance field using coordinate-based neural networks, and employs volumetric rendering to synthesis images under different viewing directions.

More concretely, to synthesize the color of a pixel at time $t$, it first samples points along the line of sight. The sampled points $\mathbf{p}$'s are then fed into a *backward* deformation field , *i.e*.

$$w_{c\leftarrow}(\mathbf{p}; t) \longrightarrow \mathbf{p}_c, \tag{7.1}$$

which maps the input spacetime point $(\mathbf{p}, t)$ to its corresponding 3D position $\mathbf{p}_c \in \mathbb{R}^3$ in the canonical space. Colors $\mathbf{c}$ and densities $\sigma$ are then queried from the radiance field network, *i.e*.

$$f(\mathbf{p}_c, \mathbf{d}, \boldsymbol{\lambda}) \longrightarrow \mathbf{c}, \sigma, \tag{7.2}$$

where $\mathbf{d} \in S^2$ is the viewing direction and $\boldsymbol{\lambda}$ is an additional frame-wise code to allow the template to vary per frame so as to cope with topological and appearance changes [129, 149]. $\boldsymbol{\lambda}$ can also be extended to vary spatially as ambient embeddings to enable greater flexibility to topological changes [150]. With the computed colors and densities of points along the viewing ray, RGB intensities of a pixel are computed using the volumetric rendering

equations proposed in NeRF [133]. Finally, the optimization objective is to minimize the difference between rendered pixels and the input observations.

**Backward deformation fields.** Different ways exist for representing the backward deformation field. The most straightforward design is to use a neural network to output displacement between the input point and its canonical position [123, 164, 195]. Park *et al*. shows that having the neural network outputting SE(3) transformations leads to improvement in reconstructing rotational motions [149].

For articulated objects such as animals and humans, blend skinning is widely used in literature [89, 124, 245, 255]. However, it is mainly designed for *forward* deformation, thus adjustment is needed to adapt it for *backward* warping field. Yang *et al*. [230, 231] uses mixture of gaussians to model the blending weights. This enables them to approximate the inverse of a forward blend skinning by simply inverting the SE(3) transformation of each deformation nodes. On the other hand, instead of explicitly define a deformation function, Chen *et al*. [30] solves for canonical correspondences of any deformed point using iterative root finding.

For homeomorphic deformation where the mappings between any frames are bijective and continuous, recent works explored the use of invertible normalizing flows [21, 112] where the forward and backward deformation are computed with the same network parameters. The downside is the network architecture is restrictive, and in practice requires more compute due to having more coupling layers to enumerate different axis partitions.

**Other dynamic NeRF representations.** Instead of having a static template NeRF, other works [51, 116, 207] choose to use a time-modulated NeRF to directly represent warped radiance fields. To enforce temporal consistency, they optimize neural scene flow fields to regularize pairwise motion between adjacent frames. This is only suitable for enforcing short-term consistency, but intractable for long-term consistency due to the expensive computational cost for performing scene flow integration. To improve computational efficiency, Wang *et al*. [207, 210] propose neural trajectory field which directly outputs trajectories for all space-time locations. This allows enforcing long-term consistency without the need for scene flow integration.

**Optical flow supervision.** Using optical flows to assist view interpolation [13, 81, 219] and 3D reconstruction [97, 125, 187] has been a common practise. Several recent dynamic NeRF works [46, 51, 116, 207] also use optical flow supervision, but none of them is based on backward deformation fields. Yang *et al*. [231] apply optical flow to supervise

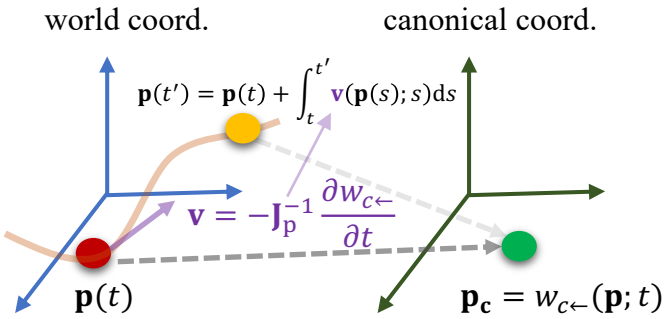



Figure 7.2: Given a point at $\mathbf{p}(t)$ and a *backward* warping field $w_{c\leftarrow}$, we want to compute the 3D scene flow by predicting the next position of the point at time $t'$. We achieve this by deriving the velocity field $v$ as a differentiable function of $w_{c\leftarrow}$, and then perform time integration.

a blend skinning deformation field for object-centric reconstruction. Their result focuses on articulated objects and requires amodel segmentation mask as input. Under the context of view synthesis for dynamic scenes, we are unaware of any deformable NeRF-based approach using flow supervision. Thus our result provides a useful tool for future related research.

## 7.3 Supervise deformable NeRF by flow

**Problem setup.** We concern the problem of fitting the deformable NeRF given a monocular video input. We precomputed the camera intrinsics and extrinsics using off-the-shelf structure-from-motion methods *e.g.* Colmap [173]. Optical flows between neighboring frames $o_{t\to t\pm\nabla t}$ are also computed using RAFT [188]. Then we want to find optimal parameters for the backward deformation field $w_{c\leftarrow}$ and radiance field $f$ such that the synthesised images $c'_t$ and optical flow maps $o'_{t\pm t'}$ match the input video frames $c_t$ and precomputed optical flow maps $o_{t\pm t'}$,

$$\min \sum_t \left[ \underbrace{\|c'_t - c_t\|_2^2}_{\text{image loss}} + \beta \sum_{t'\in\{t\pm\Delta t\}} \underbrace{\|M_{t\to t'} \odot (o'_{t\to t'} - o_{t\to t'})\|_1}_{\text{optical flow loss}} \right] \tag{7.3}$$

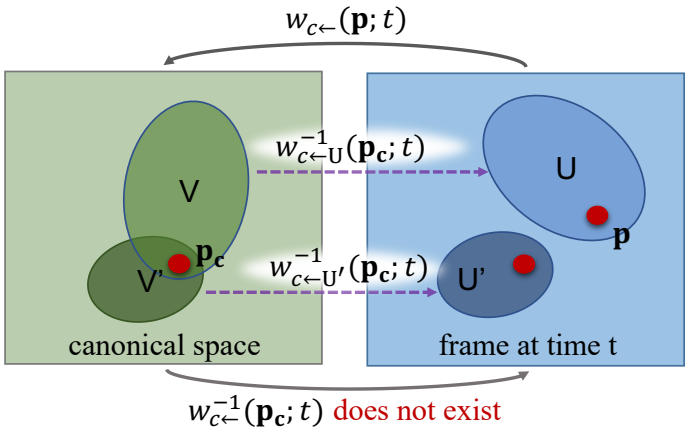



Figure 7.3: Illustration of the invertibility for *backward* deformation field $w_{c\leftarrow}$. Modeled by coordinate-based MLP, $w_{c\leftarrow}(\mathbf{p};t)$ is not invertible on the whole input domain. But local homeomorphism exists. we can have a bijective mapping $w_{c\leftarrow U}(\mathbf{p};t)$ from an open set $U$ to another open set $V$ in the canonical space. Although $w^{-1}_{c\leftarrow U}(\mathbf{p}_c;t)$ exists locally from $V$ to $U$, but it is not possible to analytically derive it in closed form. Fortunately, inverting $w_{c\leftarrow U}$ is not needed for predicting scene flow due to equation (7.5).

To prevent errors in the precomputed optical flow maps from misleading the reconstruction, we use binary masks $M_{t\rightarrow t'}$ to turn off losses for pixels which fail the forward-backward flow consistency test. Moreover, we follow the trick proposed by Li *et al*. [116] to gradually decay the weighting $\beta$ during optimization, so that the reconstruction is able to correct mistakes of the input optical flows.

The key question is then how to synthesize optical flows $o'_{t\rightarrow t'}$ from $w_{c\leftarrow}$ and $f$?

### 7.3.1 Velocity fields from $w_{c\leftarrow}(p;t)$

Velocity fields $v(\mathbf{p};t) : \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}^3$ describe the velocity of an object if placed at position $\mathbf{p}$ in the world coordinate at time $t$. Velocity fields is straightforward to compute if the *forward* deformation field $w_{c\rightarrow}(\mathbf{p}_c;t) : \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}^3$ exists, *i.e.*:

$$v(\mathbf{p};t) = \left.\frac{\partial w_{c\rightarrow}}{\partial t}\right|_{(w_{c\leftarrow}(\mathbf{p};t);t)}. \tag{7.4}$$

However, this intuitive solution is flawed if $w_{c\rightarrow}$ is not strictly an inverse function of $w_{c\leftarrow}$. Unfortunately having an invertible $w_{c\leftarrow}$ is theoretically unwarranted since most deformation

representations such as neural networks and blend skinnings are not bijective.

Since seeking a *global* inverse function of $w_{c\leftarrow}$ on the whole domain is impractical, we propose to consider local regions of $w_{c\leftarrow}$'s domain where *local* inverse function may exist. And based on the inverse function theorem, we find that we can analytically compute $v(\mathbf{p};t)$ for any positions $\mathbf{p}$ without actually inverting $w_{c\leftarrow}$, as long as $w_{c\leftarrow}$ is bijective in an open set includes $\mathbf{p}$. This leads to the main theoretical result of the paper.

**Proposition.** If the warp Jacobian matrix $\mathbf{J}_{\mathbf{p}}(\mathbf{p},t) = [\frac{\partial w_{c\leftarrow}(\mathbf{p};t)}{\partial \mathbf{p}}]$ is non-singular at some position $\mathbf{p}$ at time $t$, and there exists an open set including $\mathbf{p}$ where $w_{c\leftarrow}$ is continuously differentiable, then the velocity at $(\mathbf{p},t)$ is computed as:

$$v(\mathbf{p};t) = -\mathbf{J}_{\mathbf{p}}^{-1}(\mathbf{p},t)\frac{\partial w_{c\leftarrow}(\mathbf{p};t)}{\partial t}. \tag{7.5}$$

*Proof.* First, we upgrade $w_{c\leftarrow}$ to have the same input and output dimension, *i.e.* $\phi(\mathbf{p},t) = (w_{c\leftarrow}(\mathbf{p};t),t)$. Then given the assumptions made by the proposition, and according to the inverse function theorem [137], $\phi$ is a local diffeomorphism. In other words, there is some open set $U$ containing $(\mathbf{p},t)$ and an open set $V$ containing $\phi(\mathbf{p},t)$ such that $\phi_U := \phi : U \to V$ has a continuous and differentiable inverse $\phi_U^{-1} : V \to U$; in particular, for $(\mathbf{p}_c,t) = \phi(\mathbf{p},t)$, we have $(J\phi_U^{-1})(\mathbf{p}_c,t) = [(J\phi_U)(\mathbf{p},t)]^{-1}$. Re-expressing $\phi_U = (w_{c\leftarrow},t)$ and denoting the inverse of $w_{c\leftarrow}$ inside the open set $U$ as $w_{c\leftarrow U}^{-1}$ (see Fig. 7.3) yields :

$$\begin{bmatrix} J_{\mathbf{p}} w_{c\leftarrow U}^{-1} & \frac{\partial w_{c\leftarrow U}^{-1}}{\partial t} \\ \mathbf{0}_3^\top & 1 \end{bmatrix}\Bigg|_{(\mathbf{p}_c,t)} = \begin{bmatrix} J_{\mathbf{p}} w_{c\leftarrow} & \frac{\partial w_{c\leftarrow}}{\partial t} \\ \mathbf{0}_3^\top & 1 \end{bmatrix}^{-1}\Bigg|_{(\mathbf{p},t)} \tag{7.6}$$

By moving the matrix inverse inside the block matrix on the righthand side of (7.6) through Schur complement, we have $\frac{\partial w_{c\leftarrow U}^{-1}(\mathbf{p}_c,t)}{\partial t} = -[(J_{\mathbf{p}} w_{c\leftarrow})(\mathbf{p},t)]^{-1}\frac{\partial w_{c\leftarrow}(\mathbf{p},t)}{\partial t}$. Finally by noticing that $v(\mathbf{p};t) = \frac{\partial w_{c\leftarrow U}^{-1}(\mathbf{p}_c,t)}{\partial t}$, we have the proof. □

We note that the sufficient condition of the proposition is weak and satisfied for most domain of deformation fields. For rare cases where $\det(\mathbf{J}_{\mathbf{p}}) < \epsilon$ for some space-time positions $(\mathbf{p},t)$, we choose to exclude it from evaluating the loss so as to avoid numerical instability. Moreover, we implement equation (7.5) as a differentiable operator so that gradients can be back propagated during optimization.

### 7.3.2 Scene flow from time integration of velocity

With the velocity fields $v(\mathbf{p}; t)$ computed by equation (7.5), we estimate 3D scene flows *i.e.* the displacement between $\mathbf{p}(t)$ and $\mathbf{p}(t + \Delta t)$ by time integration (see Fig. 7.2),

$$\mathbf{p}(t + \nabla t) - \mathbf{p}(t) = \int_{t}^{t+\Delta t} v(\mathbf{p}(s); s) ds. \tag{7.7}$$

We implement the time integration though differentiable numerical ODE solvers. Because in our problem we are dealing with scene flows between small time intervals $\nabla t$, which usually is the duration of 1 or 2 frames, we find unrolling the fourth order Runge-Kutta method [70] for two iterations gives stable results and achieves good trade-off between accuracy and computational cost. To prevent overfitting to a fixed step size, we randomly perturbed the step size by adding a Gaussian noise.

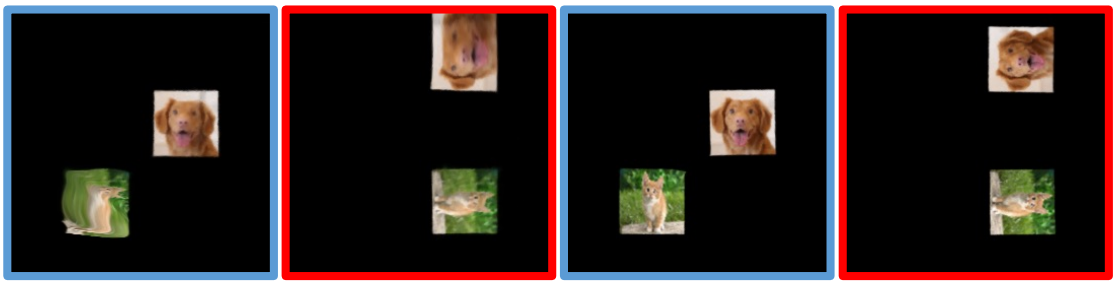
**Rendering optical flow.** For every viewing ray at time $t$, we first calculate the next position $\mathbf{p}(t + \Delta t)$ for each sampled points along the ray by equation (7.7). Then the expected next position $\bar{\mathbf{p}}(t + \Delta t)$ for the visible points is estimated by weighted averaging $\mathbf{p}(t + \Delta t)$'s using NeRF's volumetric rendering equation. Finally, the optical flow is estimated by taking the difference between the 2D projection of $\bar{\mathbf{p}}(t + \Delta t)$ and the pixel location of the viewing ray.

### 7.3.3 Removing Gauge freedom

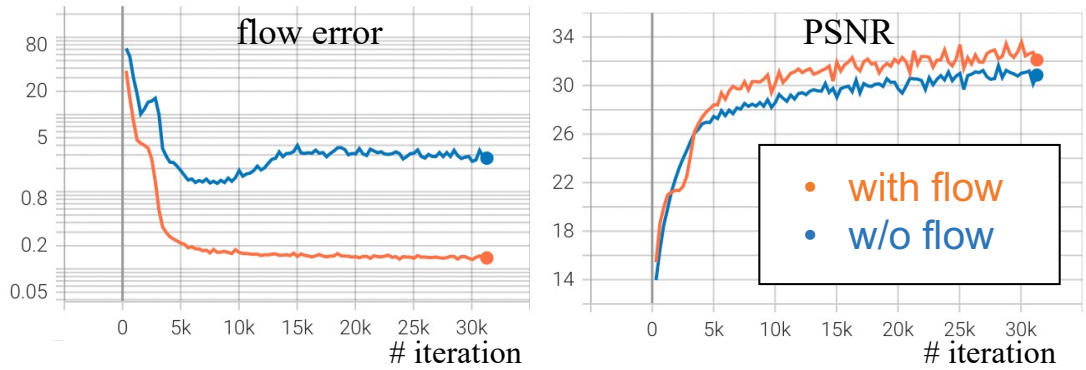
One issue for the deformable NeRF is the recovered background tends to be not static. Deformation of the apparent static scene regions severely affects viewing experience. We find that a main cause for the jittering background is due to the Gauge ambiguity in deformable NeRF, *i.e.* the optimization objective in (7.3) does not specify which canonical coordinate the deformation field is defined at. In theory, any deformation of a canonical space can also be a valid one. This causes confusion for the deformable NeRF during optimization, which tends to trap it in the wrong factorization of motion and shapes.

Therefore, we propose to remove the Gauge freedom by specifying that the canonical coordinate is attached to a key frame $t_0$ from the input sequence. This is enforced through

(a) Input video with rapid SE(2) motions.



(b) w/o flow (c) with flow




(d) convergence during optimization.

Figure 7.4: A 2D toy experiment showing flow supervision is crucial for reconstructing rapid object motions. (a) 4 frames evenly sampled from the synthetic video. Two image patches are moving with large linear and angular velocity and has no overlap between their starting and ending positions. (b) fitting a SE(2) deformation field fails to recover correct motions for the first and last frame. (c) adding flow supervision correctly reconstruct the video frames.(d) we monitor the flow error and PSNR of reconstructed image during optimization. The flows are estimated using equation (7.5,7.7). Our method (with flow) converges significantly faster than w/o flow, and is able to fit the input optical flows with low error.

Figure 7.5: 3D visualizing of trajectories computed by integrating velocity fields $v(\mathbf{p}; t)$ from equation (7.5). Trajectories are colored to represent temporal order, and overlayed with colored point clouds extracted from the optimized deformable NeRF. Bottom row shows two frames from each of the input videos.

adding a loss to penalize the magnitude of deformation at time $t_0$, *i.e.*

$$\mathcal{L}_{\text{Gauge}} = \sum_{\mathbf{p}} \|w_{c\leftarrow}(\mathbf{p}; t_0) - \mathbf{p}\|_2. \tag{7.8}$$

Through our experiments, we choose the middle frame of the input sequence as the canonical frame. This is based on the assumption that the average deformation to other frames is likely to be minimal in the middle frame. Nevertheless, more sophisticated algorithms of choosing canonical frames may further improve the robustness of the method.

We note that prior works would introduce extra regularization to enforce static background. Compared to training the density field to align with sparse points estimated by structure from motion [149, 226], our approach is self-contained without external modules. Compared to prior works having separate NeRFs for static and dynamic regions [51, 116, 129, 224], our implementation is computationally more friendly. Though all the above methods could be included as supplementary.

## 7.4 Experiment

### 7.4.1 Implementation details

**Deformation field.** We follow Nerfies [149] by using a 6 layer 128-width MLP which outputs SE(3) transformations. The deformation $\text{MLP}_g$ takes input $\mathbf{p} \in \mathbb{R}^3$ and an 8-dim deformation code. Instead of treating the codes as independent variables, we use another 2 layer 32-width $\text{MLP}_\alpha$ to map time $t$ to the deformation code. This helps improve

convergence under smaller batch size. We also choose to use softplus as the activation function, since it produces smoother outputs than ReLU. With these, mathematically $w_{c\leftarrow}(\mathbf{p}; t) = \text{MLP}_g(\mathbf{p}, \text{MLP}_\alpha(t)) \circ \mathbf{p}$, where $\circ$ denotes the action of SE(3) transform.

**Radiance field.** We use the original architecture from Nefies with minor modification. Instead of having independent 8-dim appearance codes to optimize, we use a 2 layer 32-width MLP to produce the codes conditioned on time $t$.

**Optimization hyperparameters.** We set the initial weighting $\beta$ for the optical flow loss as 0.04 and anneal it to 0.0001. We set a fixed weighting for $\mathcal{L}_{\text{Gauge}}$ as 1. The initial learning rate of Adam optimizer is 0.001, and decayed 1/10 every 50 epochs. We train a total of 120 epochs or roughly 150k iterations. Each batch samples 4096 rays from 8 frames, and samples 256 points per ray. The optimization takes 4 GTX-3090 GPUs approximately 13hrs.

### 7.4.2 Effectiveness of flow supervision

In experiments, we evaluate the effectiveness of flow supervision using the velocity fields derived in equation (7.5). We aim to answer two key qustions:

- *Does our method help improve convergence for rapid motions?*
- *Does the flow supervision help disambiguate dynamic motion and structure in monocular deformable NeRF?*

#### Flow supervision for fitting rapid motions

To clearly address the first question, we conduct a 2D toy experiment. As shown in Fig. 7.4, we created a 25-frame video, consists of two rapidly moving image patches. Unlike the synthetic experiment in Nerfies [149], the image patches have large translational motion in addition to rotations, and the motions are different for each patches. It turns out that fitting an SE(2) deformation field using only image intensity loss is slow to converge, and results in distorted images. Applying optical flow loss by our method significantly speeds up convergence, and gives correct image reconstruction. This result indicates that our flow calculation algorithm *i.e.* equation (7.5,7.7) is effective and also flow supervision is necessary for handling rapid motions.

### Monocular dynamic view synthesis

**Dataset.** State-of-the-art deformable NeRF, *e.g*. Nerfies and HyperNeRF [149, 150] have shown satisfactory view synthesis result on the data they captured. However as discussed by Gao *et al*. [53], Nerfies and HyperNeRF's data have high effective multi-view factors (EMFs). In other words, the objects are either quasi static or the camera motions are significantly larger than object motions. In this work, we evaluate on datesets with less EMFs.

We first report results on the NVIDIA dynamic view synthesis dataset (NDVS) [234] which has significantly lower EMFs. We follow the preprocessing steps of NSFF [116] which extracted 24 frames per sequence from the raw multi-view videos in NDVS. Neighboring frames are extracted from different cameras to simulate a monocular moving camera. For fair comparison with NSFF, we also downsize the images to have 288 pixels in height. For evaluation, we compare synthesized images to all images captured by 12 cameras, and report metrics such as PSNR, SSIM [220] and LPIPS [240].

We next compare view synthesis results on sequences collected by the authors of Nerfies [149]. To ensure the inputs appear as if they were casually recorded in real life, we use video frames only from the left camera from the stereo rig, as opposed to teleporting between the left and right cameras in the original paper of Nerfies.

We also test our method on one DAVIS sequence [157] and two casual videos captured by Wang *et al*. [207].

**Trajectories by velocity integration.** To visually inspect the quality of our optimized deformation field and the derived velocity fields, we sparsely sample points on the surface of the reconstructed scene, and perform time integration with the velocity fields to create trajectories. As visualized in Fig. 7.5, the recovered trajectories are smooth and closely follow the object motions.

**Foreground background separation.** Since we removed Gauge freedom by picking one video frame as the canonical frame, the distance of a point $\mathbf{p}$ to its canonical correspondence $\mathbf{p}_c$ now directly indicates whether the point is static or moving. In Fig. 7.6, we visualized the distance $\|\mathbf{p} - \mathbf{p}_c\|_2$ for each frame in a video. To visualize 3D volumes of distances in 2D, we project the distances along a ray by the volumetric rendering equation in NeRF. Thus brightness of the color corresponds to the distance of the visible area. We only observe large distances for the moving balloons and some small distances for the human subjects.

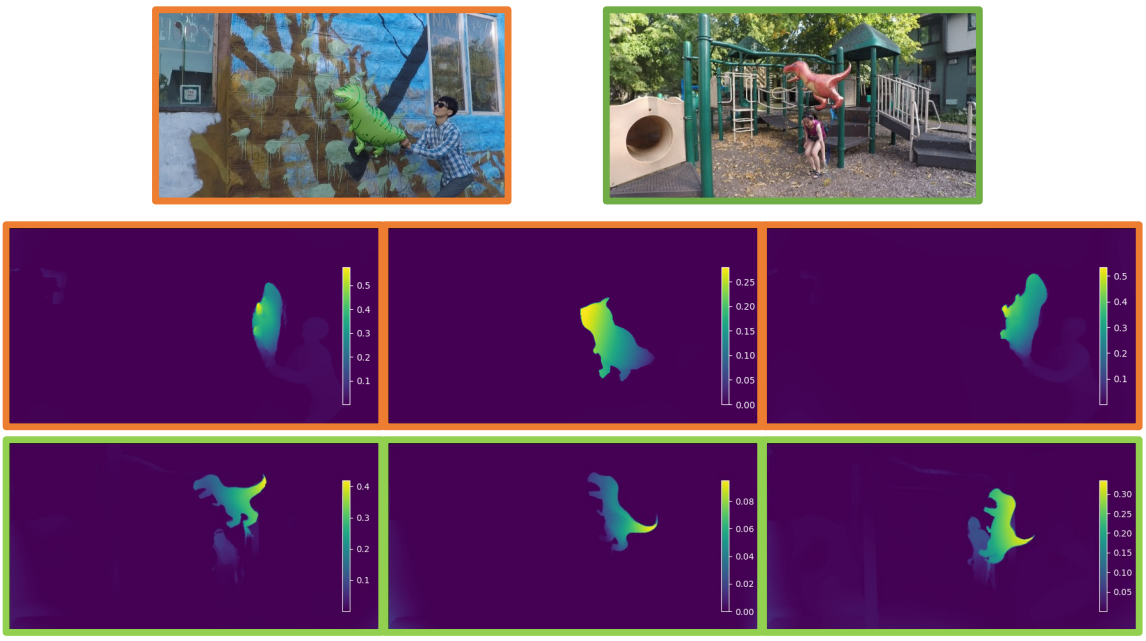


Figure 7.6: Visualization the distance $\|\mathbf{p}-\mathbf{p}_c\|_2$ by volumetric rendering. Since we removed the Gauge freedom by picking the canonical frame to be one of the input frame, large distance only happens on the moving objects. Our result shows clean separation between the moving forground objects and the static background.

The distances on the static background region are correctly rendered as close to 0. This indicates our success on the proposed removal of Gauge freedom.

**Baselines.** We formed a close comparison with the state-of-the-art deformable NeRFs *e.g*. Nerfies [149] on the NDVS dataset. Due to the official code from the authors do not work out of the box for the NDVS dataset, we adapt it by changing its Euclidean coordinates to the NDC coordinates, so as to automatically deal with the increased scene depth in some of the NDVS sequences. Given all the aforementioned implementation and design choices, our own method is essentially applying the proposed flow supervision to the adapted Nerfies implementation. Thus comparison to Nerfies also serves as an ablation showing the effectiveness of the proposed flow supervision.

We also compared with another deformable NeRF approach, *i.e*. NR-NeRF [195], whose deformation network outputs translation rather than SE(3) transformation. Finally, as a reference, we compared to NSFF [116], which optimizes a time-modulated NeRF and scene flow fields, and is supervised not only by optical flows but also by the depth maps from the state-of-the-art monocular depth estimation network [161]. Thus NSFF serves

as a strong reference to check the status of other methods without depth supervision. We summarize the compared methods in Table. 7.1.

| method | representation | | supervision | | |
|---|---|---|---|---|---|
| | motion | NeRF | image | flow | depth |
| NSFF [116] | scene flow | dynamic | ✓ | ✓ | ✓ |
| NR-NeRF [195] | translation | static | ✓ | | |
| Nerfies [149] | SE(3) | static | ✓ | | |
| HyperNeRF [150] | SE(3) + ambient | semi-static | ✓ | | |
| ours | SE(3) | static | ✓ | ✓ | |

Table 7.1: Summary of the compared deformable NeRF methods and neural scene flow field (NSFF).

| method | | Playground | | | Balloon1 | | | Balloon2 | | | Umbrella | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ |
| NSFF [116] | full | 24.69 | 0.889 | 0.065 | 24.36 | 0.891 | 0.061 | 30.59 | 0.953 | 0.030 | 24.40 | 0.847 | 0.088 |
| | dyn. | 19.02 | 0.715 | 0.123 | 18.49 | 0.619 | 0.174 | 24.46 | 0.843 | 0.065 | 16.82 | 0.546 | 0.156 |
| NR-NeRF [195] | full | 14.16 | 0.337 | 0.363 | 15.98 | 0.444 | 0.277 | 20.49 | 0.731 | 0.348 | 20.20 | 0.526 | 0.315 |
| | dyn. | 11.78 | 0.221 | 0.466 | 16.94 | 0.548 | 0.398 | 12.65 | 0.353 | 0.575 | 16.20 | 0.435 | 0.321 |
| w/o flow | full | 22.18 | 0.802 | 0.133 | 23.36 | 0.852 | 0.102 | 24.91 | 0.864 | 0.089 | 24.29 | 0.803 | 0.169 |
| (*Nerfies) | dyn. | 16.33 | 0.535 | 0.244 | 18.66 | 0.613 | 0.215 | 20.50 | 0.717 | 0.141 | 17.57 | 0.581 | 0.202 |
| w flow | full | 22.39 | 0.812 | **0.109** | 24.36 | 0.865 | 0.107 | 25.82 | 0.899 | **0.081** | 24.25 | 0.813 | **0.123** |
| (ours) | dyn. | 16.70 | 0.597 | **0.168** | 19.53 | 0.654 | **0.175** | 20.13 | 0.719 | **0.113** | 18.00 | 0.597 | **0.148** |

Table 7.2: Comparing deformable NeRFs and NSFF on the NVIDIA dynamic view synthesis (NDVS) dataset. Metrics are reported for the full image as well as only for the masked regions containing dynamic motions. Our method shows significant improvement over deformable NeRF methods without flow supervision. Comparison with NSFF is mixed, mainly due to the scale ambiguity without depth supervision. See Fig. 7.9 for further discussion.

**With vs. w/o flow supervision.** In Fig. 7.7 we form a side-by-side comparison with Nerfies, which does not use optical flow supervision. We notice that Nerfies constantly make structural mistakes as indicated by its rendered noisy depth maps. As a result, it has noticeable artifacts concentrated around the dynamic objects. In contrast, with the help from flow supervision, our method renders smoother depth maps and visually more pleasing view synthesis images. These improvements are reflected quantitatively in Tab. 7.2, where we show consistent improvement across all metrics. Our method also produces more plausible results for quasi-static scenes as shown in Fig. 7.8.

**Compare with NSFF.** In table 7.2 we show competitive results on Balloon1 and Umbrella compared to NSFF which is supervised using depth. However we fall behind on the


Figure 7.7: Results on NVIDIA dyanamic view synthesis dataset (NDVS). With flow supervision, our method produces smooth depth map and sharp novel view images. Without flow supervision leads to severe artifacts and noisy depth maps. We note that *Nerfies is our own adaptation of the official code for NDVS dataset.

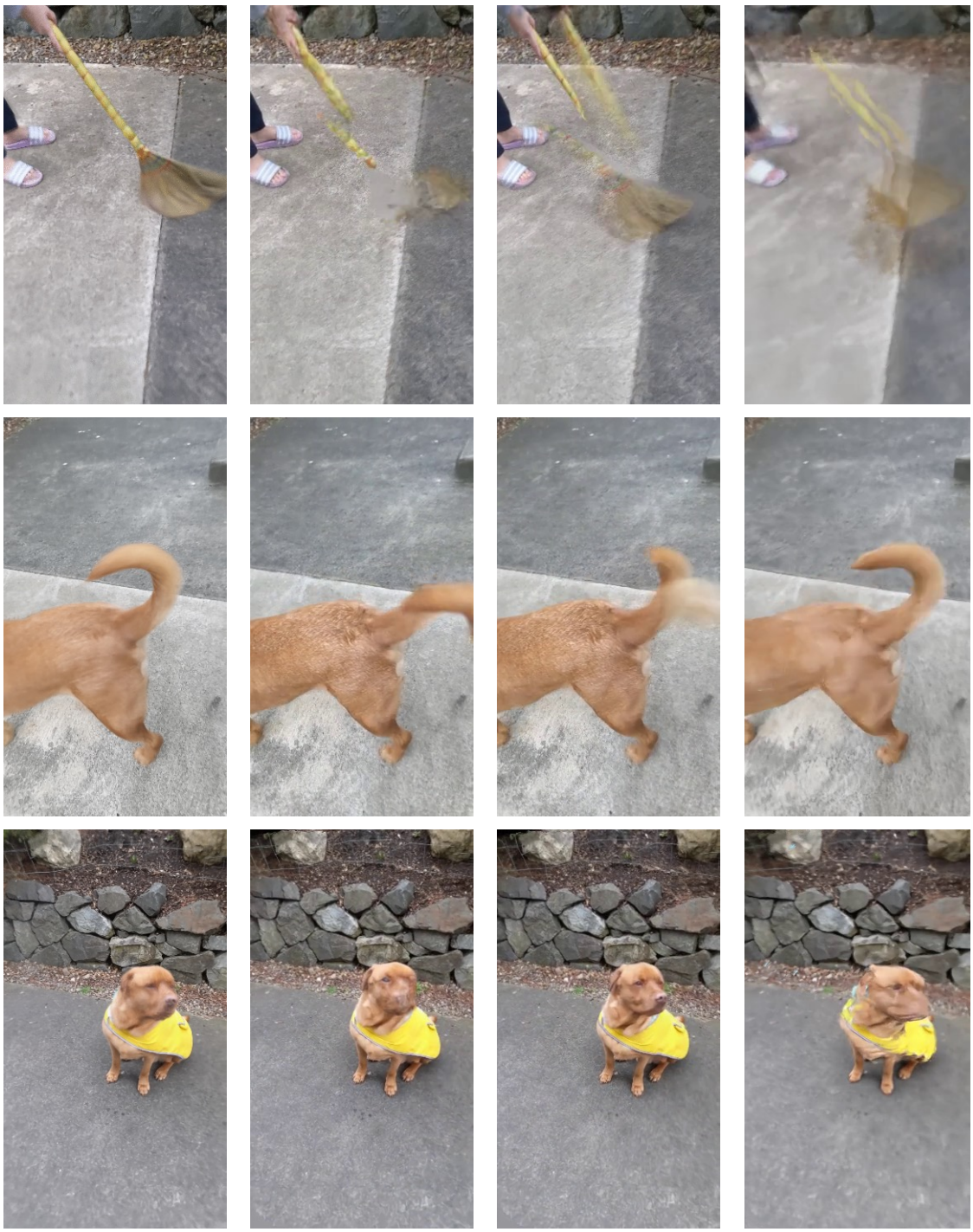
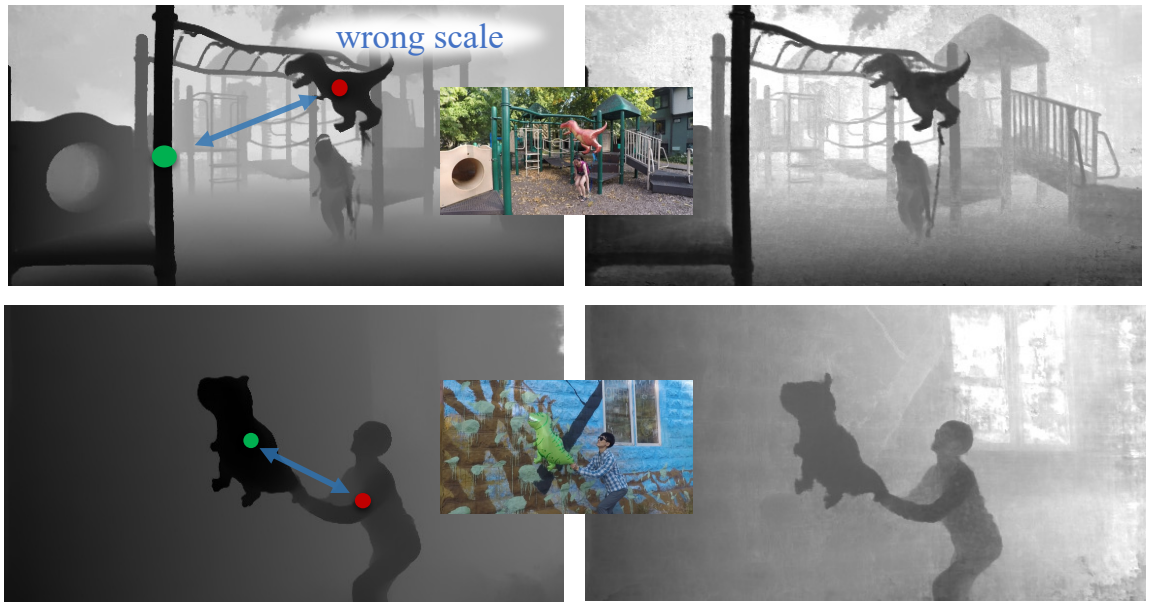



Figure 7.8: View synthesis results on sequences from Nerfies [149]. We only use frames from the left camera for training, instead of teleportation between left and right cameras as in the original paper of Nefies. We find the compared methods make noticeable mistakes in the "broom" sequence (1st row) and some frames in the "tail" sequence (2nd row). In the "toby-sit" sequence (3rd row), HyperNeRF and NSFF produce blurry or distorted dog faces in some frames. In contrast, our method consistently yields a more plausible view synthesis result. This indicates that adding flow supervision by our method is also helpful for quasi-static videos.

Playground and Balloon2 sequence. Closer diagnoses show that this is due to wrong depth scales which our method assigned to the fast moving balloons. As shown in Fig. 7.9, our method actually produces smoother depth maps compared to NSFF, thanks to our stronger temporal constraint due to having a single static template NeRF. However as highlighted by the blue arrows, the depth value of the moving objects are too small in comparison to the reference points on the background. We hypothesize this is due to the small motion bias of the deformation field which tends to explain 2D motions with smaller 3D motions. This causes the rendered foreground objects have significant offsets compared to the groundtruth, and consequently receives large penalties in terms of the image similarity metrics used in Table 7.2, even though our method produces equivalent if not sharper view synthesis result compared to NSFF. This scale ambiguity issue is inherent from the single camera problem setup and should not blame the methods supervised without depth. To recover plausible relative scales of different moving parts of a dynamic scene, mid or higher level reasoning (*e.g.* learning-based depth estimation [161, 213], 2D supervision from image generative models [158]) is required.

## 7.5 Discussion

We presented a method to apply flow supervision for deformable NeRF. We demonstrated that our method significantly improves view synthesis quality of deformable NeRF on videos with lower effective multi-view factors. However, due to the ambiguities of the monocular 3D reconstruction problem, our method has following **limitations:** (i) our method is not able to recover correct relative scale of moving objects (see Fig. 7.9); (ii) our method can be sensitive to the selection of canonical frame if there is large object deformations (see Fig. 7.10); (iii) it requires sufficient motion parallax and does not work for fixed or small camera motion; (iv) Long optimization time is required, but could be sped up using more efficient implementation [136].




(a) ours (b) NSFF

Figure 7.9: Our method suffers from scale ambiguity due to not using depth supervision. This is highlighted on the top left by comparing the points on the balloon and the pole, where the balloon should be behind the pole rather than having similar depth. Similarly, on the bottom left, the arm of the person should be close to the balloon, not behind it. Although we produce much smoother depth maps compared to NSFF, we make more error in the scale of depth, resulting in lower metrics in Table 7.2.

(a) selected canonical frame (b) noisy depth

Figure 7.10: For highly deformable objects such as umbrella, the choice of the canonical frame is sensitive. In this example, we choose the first frame instead of mid frame as the canonical frame, which has a very different topogy compared to other frames when the umbrella is open. This leads to degenerated results as highlighted by the red box.

## 7.6 Appendix

### 7.6.1 Additional implementation details

#### SE(3) transformation in normalized device coordinate

For unbounded frontal scenes in the DVS dataset [234], normalized device coordinate (NDC) [133] is required to squeeze unbounded 3D space into a bounded one with respect to the depth direction. To perform SE(3) transformation of points with the NDC coordinates, naive implementation would be first convert the NDC coordinates to Euclidean coordinates, apply the SE(3) transformation, and then convert it back to the NDC coordinates. However doing so may run into numerical instability issue for points in long distance. Alternatively we derived a formula to directly apply SE(3) transformation in the NDC coordinates. For any NDC coordinates $(x, y, z)$, its corresponding NDC coordinates $(x', y', z')$ after applying the SE(3) transformation with rotation $\mathbf{R} \in SO(3)$ and translation $\mathbf{t} \in \mathbb{R}^3$ is calculated as follow:

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \mathbf{R} \begin{pmatrix} \frac{2f_x x}{W} \\ \frac{2f_y y}{H} \\ 1 \end{pmatrix} + \frac{1-z}{2n}\mathbf{t} \tag{7.9}$$

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \frac{2f_x a}{Wc} \\ \frac{2f_y b}{Hc} \\ \frac{1+c-z}{c} \end{pmatrix}, \tag{7.10}$$

where $f_x, f_y, H, W, n$ are the focal lengths, image sizes and near plane distance as in the definition of NDC coordinates. The above formulation avoids the hassle of dealing with points at infinity in the Euclidean coordinates.

#### Efficient implementation of equation (5)

To be able to backpropagate through equation (5) and make the computation tractable for evaluating on 100k+ points per iteration, we made the following implementation design choices.

First, we choose to implement a 2nd-order differentiable quaternion-based operator using CUDA. This is 40x faster than implementation with native pytorch operators as in

PyTorch3D [163], and 5x faster than matrix multiplication based implementation. We note that LieTorch library [189] also provides fast implementation of SE(3) transformation and computes gradients in the tangent space. However it does not support 2nd-order derivatives which is required for optimization with respect to the velocities estimated by equation (5).

For calculating the $3\times3$ matrix inverse of the Jacobian matrices, me choose to implement an analytical solution of the matrix inverse with CUDA, which is 10x faster than pytorch's generic matrix inverse routine when dealing with 100k $3 \times 3$ small matrices. We use the following equation to calculate the derivative,

$$\frac{\partial \mathbf{H}^{-1}}{\partial h_{ij}} = -\mathbf{H}^{-1}\boldsymbol{\delta}_{ij}\mathbf{H}^{-1}, \tag{7.11}$$

where $\boldsymbol{\delta}_{ij}$ is a binary matrix whose $(i, j)$th entry is the element with value of 1. In our CUDA implementation, we parallalize the compute for each element of $\frac{\partial \mathbf{H}^{-1}}{\partial h_{ij}}$ so as to be significantly faster than the generic matrix inverse operators from pytorch.

### 7.6.2 Comparison to directly inverting the backward deformation field.

In Fig. 7.11, we compare our method against two baselines which directly invert the backward deformation field. The experiment setup is to fit a video with se(2) motions as described in Figure 4 of the main paper. The optimization objective is to minimize the combination of photometric loss and optical flow loss.

The first baseline (3rd column) has a separate MLP which represents the *forward* deformation field $w_{c\rightarrow}$. It has the same architecture as the *backward* deformation field $w_{c\leftarrow}$. Then the optical flow between frame $t$ and $t + \Delta t$ at pixel $\mathbf{p}$ is estimated as $\mathbf{o}_{t\rightarrow t+\Delta t} = w_{c\rightarrow}(w_{c\leftarrow}(\mathbf{p}, t), t + \Delta t) - \mathbf{p}$. As shown in the 3rd column of the following figure, this baseline is not able to reconstruct the input video with high fidelity (PSNR=24.8). This is due to there being no explicit gaurantee that the forward and backward deformation fields are cyclic consistent.

The 2nd baseline applies normalizing flow as in Lei & Daniilidis [112] The deformation field is modeled by Real-NVP, which is a bijective mapping with analytic inverse. We find that due to the network architecture restrictions, Real-NVP is not able to perfectly fit the motions (see distortion of image patches in the 4th column), even when it has significantly

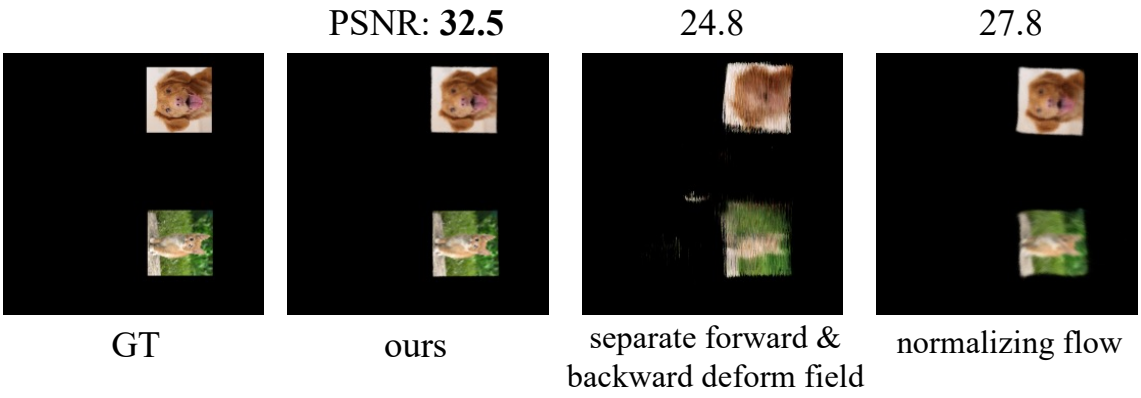



Figure 7.11: Compare flow supervision using different deformation representation. Our method (2nd column) outperforms baselines using separate forward & backward deformation field (3rd column) and bijective normalizing flow (4th column).

more layers than our method (12 vs 4).

In comparison, our method achieves the highest PSNR which indicates its effectiveness against directly inverting the backward deformation field. Finally, we note that it may be possible to make the baselines stronger by carefully tuning loss functions or network architectures, however the main strength of our method is its generality without the need for introducing additional modules or tuning.

### 7.6.3 Other optical flow rendering/loss alternatives

In our preliminary investigation, we also experimented different ways of rendering the optical flow from 3D scene flows and evaluating the flow loss. All of the following alternatives are inferior to the one we described in the main paper, and we have recorded them here for reference.

- The first alternative is instead of evaluating scene flows for all sampled points along the ray and then weighted averaging them, we only evaluate scene flows for a single point on the estimated visible surface. More specifically, we first synthesize the depth as in NeRFs [116, 133, 149] by weighted averaging depth of sampled points along the ray. This gives an estimation of a position $\mathbf{p}(t)$ on the visible surface. Then the next position $\mathbf{p}(t + \Delta t)$ is estimated by equation (7).
- We still evaluate scene flows for all the points along the ray. But instead of aggregate

them together to form a single optical flow, we project all scene flows to 2D flows, and directly evaluate optical flow loss by comparing them to the input optical flow. To account for occlusions, we weighted the loss by the weights from the volumetric rendering equation. More specifically, given the projected 2D flows $\mathbf{o}_i \in \mathbb{R}^2$ for each point $i$ along the ray, we evaluate the optical flow loss as follow:

$$\mathcal{L}_{\text{o.f.}} = \sum_i w_i \|\mathbf{o}_i - \mathbf{o}_{\text{input}}\|_1, \tag{7.12}$$

where $w_i$ denotes the weights from the volumetric rendering and $\mathbf{o}_{\text{input}} \in \mathbb{R}^2$ is the input optical flow.

In our current experiment, we found both above approaches are inferior to the one we used in the main paper. Sometimes they resulted in training instability or floating surfaces. A deeper understanding of why these two approaches do not work may motivate more efficient approaches of enforcing the flow loss.

# Chapter 8

# Discussion and Future Directions

In the previous chapters, considerable effort has been dedicated to tackling the difficult task of reconstructing 3D structures from monocular videos. Despite these efforts, the problem remains challenging to solve. One of the main limitations of the solutions discussed, which rely on a prior-less approach, is the reliance solely on motion parallax and basic geometric constraints like motion smoothness. Although the use of motion parallax and geometric constraints can provide valuable cues to infer the structure of non-rigid objects, however, without prior knowledge or additional cues, accurately determining the relative scale between different objects and distinguishing their boundaries becomes a challenging endeavor. This limitation becomes evident in cases where the proposed deformable NeRF method is applied.

Addressing the challenges in reconstructing accurate 3D structures from monocular videos requires exploring alternative methods that go beyond relying solely on motion parallax and basic geometric constraints. Incorporating additional cues, such as semantic information, or contextual priors, could potentially help overcome the ambiguities between non-rigid motion and structure. Furthermore, leveraging machine learning techniques and larger-scale datasets may also contribute to improving the accuracy and robustness of the reconstruction process. In this regard, we delve into several directions that have been explored in collaboration with other researchers, as well as envision future paths where I believe significant breakthroughs can be achieved.

## 8.1 Learning object shape from segmentation masks


Figure 8.1: Illustration of the problem setup of our SDF-SRN method. SDF-SRN predicts the continuous 3D surface of an object by training on a dataset comprising images capturing various instances from identical object categories. The predicted surface is represented by a signed distance function (SDF). Training is facilitated through the utilization of 2D segmentation masks as supervision.

The process of reconstructing 3D shapes from 2D silhouettes, known as Shape-From-Silhouette (SfS), has long been recognized as a classical approach in 3D reconstruction. The origins of SfS can be traced back to Baumgart's 1974 paper [15], in which he successfully estimated the 3D shapes of a baby doll and a toy horse using four silhouette images. SfS is often favored over other reconstruction methods, primarily because its implementation is comparatively straightforward compared to alternative approaches like multi-view stereo and space carving.

Previously, a major obstacle that hindered the widespread adoption of Shape-From-Silhouette (SfS) as a popular reconstruction method was the difficulty in acquiring accurate 2D silhouettes. Traditionally, estimating 2D silhouettes relied on foreground-background separation assuming a static background and camera setup, excluding the presence of moving shadows. These strong assumptions did not hold up well in real-world scenarios captured in casual images and videos.

However, recent advancements in object segmentation, such as Mask-RCNN [75], have alleviated this limitation. These advancements have enabled the generation of masks for objects belonging to a wide range of categories, surpassing the previous constraints. Remarkably, during the course of writing this thesis, we have witnessed breakthroughs in object segmentation, exemplified by the training of large vision transformer networks that can accurately segment almost any object based on simple text descriptions [92]. These

advancements solidify the notion of using segmentation masks as supervision, in addition to image pixel values, to reconstruct objects.

The incorporation of additional supervision from segmentation proves to be crucial in the reconstruction of non-rigid objects from casually captured videos. This additional supervision provides valuable information regarding the separation of the static background and the non-rigid foreground object. Consequently, different regularization techniques can be applied more effectively to different regions of the scene. In contrast, performing foreground-background separation solely based on motion parallax is inherently ambiguous. This ambiguity arises from the fact that epipolar constraints, which are relied upon by many motion segmentation methods, are insufficient in determining whether the motion of different pixels corresponds to the same rigid motion.

Furthermore, the availability of segmentation masks allows us to tackle an even more challenging problem setup, as illustrated in Figure 8.1, known as shape from category. In this setup, we are provided with images of different instances of objects belonging to the same object category. Our goal in this scenario is to learn a probabilistic generative model of the 3D shape of objects within the category, alongside an inference network capable of predicting the encoding of the 3D shape from a given 2D image. To train these networks effectively, we utilize the segmentation masks to remove the background and supervise the silhouette of the object of interest. For a more comprehensive understanding of the methodology and technical details, we kindly refer interested readers to our papers listed below:

- Lin, C.H., **Wang, C.** and Lucey, S., "Sdf-srn: Learning signed distance 3d object reconstruction from static images", NeurIPS 2020.
- Zhu, R., Kiani Galoogahi, H., **Wang, C.** and Lucey, S., "Rethinking reprojection: Closing the loop for pose-aware shape reconstruction from a single image", ICCV 2017.

## 8.2 Regularize articulated shape with skeleton

In all the methods discussed in the previous chapters, we have intentionally avoided making assumptions about the connectivity of object parts and instead treated deformation as free-form. While this approach offers the advantage of providing a general solution

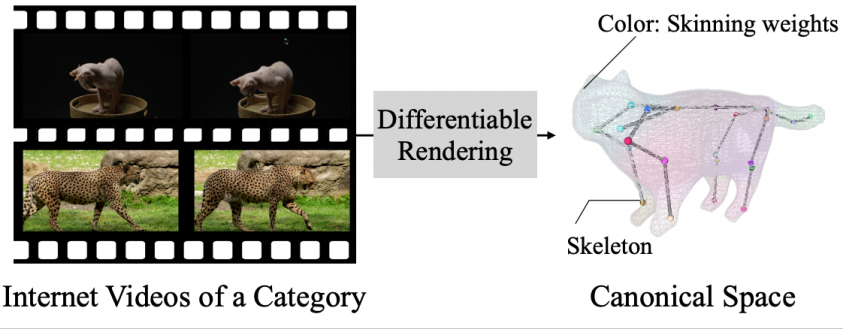



Figure 8.2: Illustration of the problem setup of our RAC (Reconstructing Animatable Categories from Videos) method. Given input videos of different instances of an articulated object category and a predefined skeleton structure, RAC optimizes the canonical shape and skinning weights to create an animatable model.

applicable to various types of non-rigid objects and dynamic scenes, it falls short in directly addressing rapid articulated motions. This is primarily because the lack of regularization and less efficient parameterization of the deformation compromise both the regularization effectiveness and computational efficiency in favor of achieving generality.

Considering the need for an efficient algorithm specifically tailored for reconstructing articulated objects like animals and humans, it becomes advantageous to design the deformation representation based on skeletons. By incorporating skeletons, the method gains valuable information about the interconnections of joints, introduces constraints such as maintaining constant bone lengths across video frames, and enables parameterization of deformation using motion primitives like blend skinning. This motivates our work on reconstructing animatable categories from videos, where we demonstrate the effectiveness of a skeleton-based approach in achieving highly accurate reconstructions of articulated objects. By leveraging skeletons, we enhance the precision and realism of the reconstructed models, enabling them to be easily animated and manipulated. For a more detailed and in-depth understanding of our methodology and the technical intricacies involved, we encourage interested readers to refer to our paper listed below:

- Yang, G., **Wang, C.**, Reddy, N. D., and Ramanan, D. Reconstructing Animatable Categories from Videos. CVPR 2023.

## 8.3 A brave new world with generative models

Deep generative models have demonstrated remarkable success in generating highly realistic images of faces, animals, cars, and buildings. Moreover, the power of Generative Adversarial Networks (GANs) has been harnessed to solve inverse problems, such as reconstructing 3D faces [24]. However, one previous limitation of applying GAN-based approaches to more general reconstruction problems stems from their dependence on structured datasets. GANs have achieved notable results when working with datasets containing aligned faces, cars, specific indoor or outdoor scenes with predefined layouts, or a combination of these elements. Unfortunately, they do not readily adapt to the challenges posed by arbitrarily unstructured scenes encountered in our daily lives. This limitation can be attributed, in part, to the mode-collapse problem inherent in GAN training. Mode collapse occurs when the GAN fails to effectively learn and capture a diverse range of images simultaneously. Consequently, the model may struggle to generalize well to novel, unstructured scenes.

In recent years, significant progress has been made in overcoming these restrictions, thanks to the remarkable advancements in diffusion models. These models offer several advantages, including efficient learning, absence of mode collapse issues, and applicability to various tasks such as text-conditioned generation, editing, and denoising. Leveraging the power of pretrained image diffusion models, a recent method *i.e*. DreamFusion [158], enables the creation of 3D assets based on input texts. DreamFusion utilizes a novel framework, specifically the score-distillation-sampling loss, which successfully distills prior knowledge from the 2D diffuser without requiring backpropagation through the diffusion model. In this setup, the diffusion model acts as an efficient and frozen critic that predicts image-space edits, enabling it to supervise the parameter updates for the 3D shapes. This approach effectively leverages the capabilities of the diffusion model as a guiding force, ensuring accurate and high-quality synthesis of 3D assets.

The distillation of 2D diffusion models for 3D reconstruction is currently a prominent research area, offering a convenient and generalizable strategy to incorporate data priors for regularization, especially in highly ill-posed reconstruction scenarios. Simultaneously, there is ongoing progress in directly learning 3D generative models using diffusion. During the writing of this thesis, we have observed several works, such as DiffRF [135], exploring the learning of 3D diffusion models from large-scale synthetic [42] or real-world 3D datasets [235]. In this direction, we envision future research focusing on expanding both

the scale of training data and the resolution of diffusion models to generate higher-fidelity 3D assets.

# Bibliography


[1] CMU Motion Capture Dataset. available at http://mocap.cs.cmu.edu/. (document), 2.1, 2.5.3, 3.5

[2] Looking Glass Factory. available at https://lookingglassfactory.com/. (document), 1.1

[3] Renderpeople. 1

[4] Pyscendetect. https://pyscenedetect.readthedocs.io, 2008. [Online; accessed 1-July-2018]. 5.3

[5] Antonio Agudo and Francesc Moreno-Noguer. Recovering pose and 3d deformable shape from multi-instance image ensembles. In *Asian Conference on Computer Vision*, pages 291–307. Springer, 2016. 3.1

[6] Antonio Agudo, Melcior Pijoan, and Francesc Moreno-Noguer. Image collection pop-up: 3d reconstruction and clustering of rigid and non-rigid categories. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. (document), 2.2, 2.2, 3.1, 3.2

[7] Antonio Agudo, Melcior Pijoan, and Francesc Moreno-Noguer. Image collection pop-up: 3d reconstruction and clustering of rigid and non-rigid categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2607–2615, 2018. 4.2, 4.5.3

[8] Ijaz Akhter, Yaser Sheikh, and Sohaib Khan. In defense of orthonormality constraints for nonrigid structure from motion. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1534–1541. IEEE, 2009. 2.1, 2.2, 3.1, 3.2

[9] Ijaz Akhter, Yaser Sheikh, Sohaib Khan, and Takeo Kanade. Nonrigid structure from motion in trajectory space. In *Advances in neural information processing systems*, pages 41–48, 2009. 2.5.3, 3.1, 3.5, 5.2

[10] Ijaz Akhter, Yaser Sheikh, Sohaib Khan, and Takeo Kanade. Trajectory space: A dual representation for nonrigid structure from motion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2010. 6.1, 6.2, 6.3

[11] Ijaz Akhter, Yaser Sheikh, Sohaib Khan, and Takeo Kanade. Trajectory space: A dual

representation for nonrigid structure from motion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(7):1442–1456, 2011. 4.2

[12] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 4.5.2

[13] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3703–3712, 2019. 7.2

[14] Adrien Bartoli, Daniel Pizarro, and Marco Loog. Stratified generalized procrustes analysis. *International journal of computer vision*, 101(2):227–253, 2013. 3.1

[15] Bruce Guenther Baumgart. *Geometric modeling for computer vision.* Stanford University, 1974. 8.1

[16] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm with application to wavelet-based image deblurring. Citeseer. 3.4, 3.4.2

[17] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm with application to wavelet-based image deblurring. 2009. 4.3

[18] Adam W Bojanczyk and Adam Lutoborski. The procrustes problem for orthogonal stiefel matrices. *SIAM Journal on Scientific Computing*, 21(4):1291–1304, 1999. 2.4.2

[19] Christoph Bregler. Recovering non-rigid 3d shape from image streams. Citeseer. 2.1, 2.2, 3.1, 3.2, 4.2, 5.2, 6.2

[20] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. 2001. 6.2

[21] Hongrui Cai, Wanquan Feng, Xuetao Feng, Yan Wang, and Juyong Zhang. Neural surface reconstruction of dynamic scenes with monocular rgb-d camera. *arXiv preprint arXiv:2206.15258*, 2022. 7.2

[22] Francesco Camastra and Antonino Staiano. Intrinsic dimension estimation: Advances and open problems. *Information Sciences*, 328:26–41, 2016. 2.6

[23] Geonho Cha, Minsik Lee, and Songhwai Oh. Unsupervised 3d reconstruction networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3849–3858, 2019. 2.1, 2.2, 2.5.2, 3.1, 3.2

[24] 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 *CVPR*, 2022. 8.3

[25] Ching-Hang Chen and Deva Ramanan. 3d human pose estimation = 2d pose es-

timation + matching. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 4.2

[26] Ching-Hang Chen, Ambrish Tyagi, Amit Agrawal, Dylan Drover, Rohith MV, Stefan Stojanov, and James M. Rehg. Unsupervised 3d pose estimation with geometric self-supervision. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2.2, 3.1, 3.2, **??**, 3.5, **??**

[27] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. 1993. 6.1, 6.2

[28] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. Single-image depth perception in the wild. In *Advances in Neural Information Processing Systems*, pages 730–738, 2016. 5.1, 5.4, 5.4.1

[29] Wenzheng Chen, Huan Wang, Yangyan Li, Hao Su, Zhenhua Wang, Changhe Tu, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Synthesizing training images for boosting human 3d pose estimation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 479–488. IEEE, 2016. 4.2

[30] Xu Chen, Yufeng Zheng, Michael J Black, Otmar Hilliges, and Andreas Geiger. Snarf: Differentiable forward skinning for animating non-rigid neural implicit shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11594–11604, 2021. 7.1, 7.2

[31] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7103–7112, 2018. (document), 3.10

[32] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 6.2

[33] Ajad Chhatkuli, Daniel Pizarro, Toby Collins, and Adrien Bartoli. Inextensible non-rigid structure-from-motion by second-order cone programming. *IEEE transactions on pattern analysis and machine intelligence*, 40(10):2428–2441, 2017. 3.1

[34] Nathaniel Chodosh, Chaoyang Wang, and Simon Lucey. Deep convolutional compressed sensing for lidar depth completion. In *Asian Conference on Computer Vision*, pages 499–513. Springer, 2018. 3.4.2

[35] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Int. Conf. Comput. Vis.*, 2019. 6.2

[36] Stéphane Christy and Radu Horaud. Euclidean shape and motion from multiple perspective views by affine iterations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(11):1098–1104, 1996. 3.2

[37] Mosam Dabhi, Chaoyang Wang, Tim Clifford, László Jeni, Ian Fasel, and Simon

Lucey. Mbw: Multi-view bootstrapping in the wild. *Advances in Neural Information Processing Systems*, 35:3039–3051, 2022. (document), 1.1

[38] Rishabh Dabral, Anurag Mundhada, Uday Kusupati, Safeer Afaque, Abhishek Sharma, and Arjun Jain. Learning 3d human pose from structure and motion. In *The European Conference on Computer Vision (ECCV)*, September 2018. 4.2

[39] Yuchao Dai, Hongdong Li, and Mingyi He. A simple prior-free method for non-rigid structure-from-motion factorization. *International Journal of Computer Vision*, 107(2):101–122, 2014. 2.1, 2.2, **??**, 3.1, 3.2, 3.2, 3.3, 3.3, 3.5, 4.2, 4.3, **??**, 5.2

[40] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 57(11):1413–1457, 2004. 4.3

[41] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. 1996. 6.1, 6.2

[42] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023. 8.3

[43] Alessio Del Bue, Fabrizio Smeraldi, and Lourdes Agapito. Non-rigid structure from motion using ranklet-based tracking and non-linear optimization. *Image and Vision Computing*, 25(3):297–310, 2007. (document), 3.2, 3.2, **??**

[44] Dylan Drover, Rohith MV, Ching-Hang Chen, Amit Agrawal, Ambrish Tyagi, and Cong Phuoc Huynh. Can 3d pose be learned from 2d projections alone? In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018. 2.2, 3.2, **??**

[45] Dylan Drover, Rohith MV, Ching-Hang Chen, Amit Agrawal, Ambrish Tyagi, and Cong Phuoc Huynh. Can 3d pose be learned from 2d projections alone? In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 4.1, 4.2, **??**, **??**

[46] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14314. IEEE Computer Society, 2021. 7.1, 7.2

[47] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014. (document), 5.1, 5.2, 5.4, 5.4.1, **??**,

5.1, 5.5.3

[48] Hsiao-Yu Fish Tung, Adam W. Harley, William Seto, and Katerina Fragkiadaki. Adversarial inverse graphics networks: Learning 2d-to-3d lifting and image-to-image translation from unpaired supervision. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. **??**, 4.1, 4.2, **??**, **??**

[49] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. DeepView: View synthesis with learned gradient descent. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 6.2

[50] Katerina Fragkiadaki, Marta Salas, Pablo Arbelaez, and Jitendra Malik. Grouping-based low-rank trajectory completion and 3d reconstruction. In *Advances in Neural Information Processing Systems*, pages 55–63, 2014. 2.2, **??**, 3.1, 3.2, **??**, **??**

[51] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. 7.1, 7.2, 7.3.3

[52] Chen Gao, Yichang Shih, Wei-Sheng Lai, Chia-Kai Liang, and Jia-Bin Huang. Portrait neural radiance fields from a single image. *arXiv preprint arXiv:2012.05903*, 2020. 6.1, 6.2

[53] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. *arXiv preprint arXiv:2210.13445*, 2022. 7.1, 7.4.2

[54] Yuan Gao and Alan L Yuille. Symmetric non-rigid structure from motion for category-specific object structure estimation. In *European Conference on Computer Vision*, pages 408–424. Springer, 2016. 3.1

[55] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer, 2016. 5.2

[56] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3354–3361. IEEE, 2012. 5.1, 5.2

[57] P. Ghosh, M. S. M. Sajjadi, A. Vergari, M. J. Black, and B. Schölkopf. From variational to deterministic autoencoders. In *8th International Conference on Learning Representations (ICLR)*, April 2020. 2.3, 2.3, 2.7.1

[58] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017. 4.4, 5.2, 5.4

[59] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley,

Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2.2, 3.2

[60] Paulo FU Gotardo and Aleix M Martinez. Computing smooth time trajectories for camera and deformable shape in structure from motion with occlusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(10):2051–2065, 2011. 4.2

[61] Paulo FU Gotardo and Aleix M Martinez. Kernel non-rigid structure from motion. In *2011 International Conference on Computer Vision*, pages 802–809. IEEE, 2011. 3.1, 3.2, 3.5, **??**, **??**, **??**

[62] Paulo FU Gotardo and Aleix M Martinez. Non-rigid structure from motion with complementary rank-3 spaces. In *CVPR 2011*, pages 3065–3072. IEEE, 2011. 3.1

[63] Stephen Gould, Richard Hartley, and Dylan Campbell. Deep declarative networks: A new hope. *arXiv preprint arXiv:1909.04866*, 2019. 2.4.2

[64] John C Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975. 3.1

[65] John C Gower, Garmt B Dijksterhuis, et al. *Procrustes problems*, volume 30. Oxford University Press on Demand, 2004. 2.4.2

[66] Ned Greene. Environment mapping and other applications of world projections. 1986. 6.2

[67] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7297–7306, 2018. 6.6.1

[68] Xiaoyang Guo, Hongsheng Li, Shuai Yi, Jimmy Ren, and Xiaogang Wang. Learning monocular depth by distilling cross-domain stereo networks. In *The European Conference on Computer Vision (ECCV)*, September 2018. 5.2

[69] Tewodros Habtegebrial, Varun Jampani, Orazio Gallo, and Didier Stricker. Generative view synthesis: From single-view semantics to novel-view images. 2020. 6.2

[70] Richard Hamming. *Numerical methods for scientists and engineers*. Courier Corporation, 2012. 7.3.2

[71] Onur C Hamsici, Paulo FU Gotardo, and Aleix M Martinez. Learning spatially-smooth mappings in non-rigid structure from motion. In *European Conference on Computer Vision*, pages 260–273. Springer, 2012. 3.1, 3.2, **??**

[72] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle videos revisited: Tracking through occlusions using point trajectories. *arXiv preprint arXiv:2204.04153*, 2022. 6.6.1

[73] Richard Hartley and René Vidal. Perspective nonrigid shape and motion recovery. In *European Conference on Computer Vision*, pages 276–289. Springer, 2008. 3.2

[74] Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85, 2005. 3.4, 3.4.2

[75] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 8.1

[76] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017. 5.3

[77] Anders Heyden, Rikard Berthilsson, and Gunnar Sparr. An iterative factorization method for projective structure and motion from image sequences. *Image and Vision Computing*, 17(13):981–991, 1999. 3.2

[78] Peiyun Hu and Deva Ramanan. Bottom-up and top-down reasoning with hierarchical rectified gaussians. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 3.4.2

[79] Hsin-Ping Huang, Hung-Yu Tseng, Hsin-Ying Lee, and Jia-Bin Huang. Semantic view synthesis. In *Eur. Conf. Comput. Vis.*, 2020. 6.2

[80] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2821–2830, 2018. 5.2

[81] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Rife: Real-time intermediate flow estimation for video frame interpolation. *arXiv preprint arXiv:2011.06294*, 2020. 7.2

[82] 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 *IEEE conference on computer vision and pattern recognition (CVPR)*, volume 2, page 6, 2017. 5.3

[83] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013. (document), 1.1, 2.5.4, 3.5, 4.1, 4.2, 4.5.2

[84] Sebastian Hoppe Nesgaard Jensen, Alessio Del Bue, Mads Emil Brix Doest, and Henrik Aanæs. A benchmark and evaluation of non-rigid structure from motion. *arXiv preprint arXiv:1801.08388*, 2018. 3.5, 3.7.3

[85] Sebastian Hoppe Nesgaard Jensen, Mads Emil Brix Doest, Henrik Aanæs, and

Alessio Del Bue. A benchmark and evaluation of non-rigid structure from motion. *Int. J. Comput. Vis.*, 2021. 6.2

[86] Hanbyul Joo, Hao Liu, Lei Tan, Lin Gui, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic studio: A massively multiview system for social motion capture. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3334–3342, 2015. 1, 4.2

[87] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. 2016. 6.2

[88] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018. **??**

[89] Ladislav Kavan. Part i: direct skinning methods and deformation primitives. In *ACM SIGGRAPH*, volume 2014, pages 1–11, 2014. 7.2

[90] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2.5.1

[91] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *stat*, 1050:1, 2014. 2.3, 2.7.1

[92] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 8.1

[93] Chen Kong and Simon Lucey. Prior-less compressible structure from motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4123–4131, 2016. (document), 2.2, 3.1, 3.2, 3.3, 3.2, 4.2, 4.3, **??**, 5.2

[94] Chen Kong and Simon Lucey. Deep non-rigid structure from motion. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. (document), 1.1, 2.1, 2.2, 2.3, 2.4.2, 2.5.1, **??**, **??**, 2.5.3, 2.7.1, 3.1, 3.2, 3.3, 3.3, 3.3.2, 3.4, 3.4.1, 3.4.2, 3.5, 3.5, 3.2, **??**, **??**, **??**, 4.1, 4.2, 4.3, **??**

[95] Chen Kong, Rui Zhu, Hamed Kiani, and Simon Lucey. Structure from category: A generic and prior-less approach. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 296–304. IEEE, 2016. 2.2, 3.1, 3.2, 3.2, 3.3

[96] Chen Kong, Rui Zhu, Hamed Kiani, and Simon Lucey. Structure from category: a generic and prior-less approach. *International Conference on 3DVision (3DV)*, 2016. 4.2, **??**

[97] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1611–1621, 2021. 7.2

[98] Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina

Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*, 2017. (document), 5.3, 5.4

[99] Yasunori Kudo, Keisuke Ogaki, Yusuke Matsui, and Yuri Odagiri. Unsupervised adversarial learning of 3d human pose from 2d joint locations. *arXiv preprint arXiv:1803.08244*, 2018. 2.2, **??**, 3.1, 3.2, **??**, **??**

[100] Suryansh Kumar. Jumping manifolds: Geometry aware dense non-rigid structure from motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5346–5355, 2019. 3.1

[101] Suryansh Kumar. Non-rigid structure from motion: Prior-free factorization method revisited. In *Winter Conference on Applications of Computer Vision (WACV 2020)*, 2020. 2.2, **??**, 2.5.3, 3.1, 3.2, 3.2, 3.3

[102] Suryansh Kumar, Anoop Cherian, Yuchao Dai, and Hongdong Li. Scalable dense non-rigid structure-from-motion: A grassmannian perspective. *arXiv preprint arXiv:1803.00233*, 2018. 4.2, 5.2

[103] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Multi-body non-rigid structure-from-motion. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 148–156. IEEE, 2016. 2.1, 2.2, 3.1, 3.2, 4.2

[104] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Monocular dense 3d reconstruction of a complex dynamic scene from two perspective frames. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4649–4657, 2017. 3.1, 3.3, 5.2, 5.5.4, 6.3.3

[105] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Spatio-temporal union of subspaces for multi-body non-rigid structure-from-motion. 2017. 6.1, 6.2

[106] Yevhen Kuznietsov, Jörg Stückler, and Bastian Leibe. Semi-supervised deep learning for monocular depth map prediction. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6647–6655, 2017. 5.2

[107] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE, 2016. 5.1

[108] Christoph Lassner, Javier Romero, Martin Kiefel, Federica Bogo, Michael J. Black, and Peter V. Gehler. Unite the people: Closing the loop between 3d and 2d human representations. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2.5.4, 3.5

[109] Minsik Lee, Jungchan Cho, Chong-Ho Choi, and Songhwai Oh. Procrustean normal distribution for non-rigid structure from motion. In *Proceedings of the IEEE Conference on computer vision and pattern recognition*, pages 1280–1287, 2013. 2.2, **??**, 2.5.3, 3.1, 3.2, 3.3.2, 5.2

[110] Minsik Lee, Jungchan Cho, and Songhwai Oh. Consensus of non-rigid reconstructions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4670–4678, 2016. (document), **??**, 2.5.3, 3.1, 3.2, 3.2, **??**, **??**, 3.7.3, **??**

[111] Minsik Lee, Chong-Ho Choi, and Songhwai Oh. A procrustean markov process for non-rigid structure recovery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 2.2

[112] Jiahui Lei and Kostas Daniilidis. Cadex: Learning canonical deformation coordinate space for dynamic surface representation via neural homeomorphism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6624–6634, 2022. 7.1, 7.2, 7.6.2

[113] Marc Levoy and Pat Hanrahan. Light field rendering. 1996. 6.1, 6.2

[114] Xueqian Li, Jhony Kaesemodel Pontes, and Simon Lucey. Neural scene flow prior. *Advances in Neural Information Processing Systems*, 34:7838–7851, 2021. 7.1

[115] Zhengqi Li, Tali Dekel, Forrester Cole, Richard Tucker, Noah Snavely, Ce Liu, and William T Freeman. Learning the depths of moving people by watching frozen people. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4521–4530, 2019. 5.2

[116] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. *arXiv preprint arXiv:2011.13084*, 2020. (document), 1.2.2, 6.1, 6.2, 6.3.2, 6.3.4, 6.4, 6.7, **??**, 6.1, **??**, 6.5.1, 7.1, 7.2, 7.3, 7.3.3, 7.4.2, 7.4.2, **??**, **??**, 7.6.3

[117] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2041–2050, 2018. 5.1, 5.2, 5.4, 5.5.4, **??**, 5.7.3, **??**

[118] Wang Lijun, Shen Xiaohui, Zhang Jianming, Wang Oliver, Hsieh Chih-Yao, Kong Sarah, and Lu Huchuan. Deeplens: Shallow depth of field from a single image. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 37(6):6:1–6:11, 2018. 5.2

[119] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 6.6.3

[120] David B Lindell, Julien NP Martel, and Gordon Wetzstein. AutoInt: Automatic integration for fast neural volume rendering. *arXiv preprint arXiv:2012.01714*, 2020.

6.2

[121] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. 2020. 6.1

[122] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen Change Loy, and Xiaoou Tang. Deep learning markov random field for semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 40(8):1814–1828, 2017. 3.4.2

[123] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, July 2019. 7.1, 7.2, 7.2

[124] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015. 7.2

[125] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Trans. Graph.*, 2020. 6.2, 6.4, **??**, 7.2

[126] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Geometry-based next frame prediction from monocular video. In *IEEE Intelligent Vehicles Symposium, IV 2017, Los Angeles, CA, USA, June 11-14, 2017*, pages 1700–1707, 2017. 5.2

[127] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5667–5675, 2018. 5.2

[128] Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. Learning trajectory dependencies for human motion prediction. 2019. 6.3

[129] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the wild: Neural radiance fields for unconstrained photo collections. 2020. 6.2, 6.3.4, 7.2, 7.3.3

[130] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2640–2649, 2017. 2.3, 2.5.1, **??**

[131] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2640–2649, 2017. 4.2

[132] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning reconstruction in function space. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 6.2

[133] 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 *Eur. Conf. Comput. Vis.*, 2020. 6.1, 6.2, 6.3, 6.3.4, 6.5.1, 7.2, 7.6.1, 7.6.3

[134] Ab Mooijaart and Jacques JF Commandeur. A general solution of the weighted orthonormal procrustes problem. *Psychometrika*, 55(4):657–663, 1990. 2.4.2

[135] Norman Muller, Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulo, Peter Kontschieder, and Matthias Niessner. Diffrf: Rendering-guided 3d radiance field diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4328–4338, 2023. 8.3

[136] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv:2201.05989*, January 2022. 7.5

[137] James R Munkres. *Analysis on manifolds*. CRC Press, 2018. 7.3.1

[138] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. 5.2

[139] Calvin Murdock, MingFang Chang, and Simon Lucey. Deep component analysis via alternating direction neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 820–836, 2018. 3.4.2

[140] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 5.5.1

[141] Michael Niemeyer and Andreas Geiger. GIRAFFE: Representing scenes as compositional generative neural feature fields. *arXiv preprint arXiv:2011.12100*, 2020. 6.2

[142] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 6.2

[143] David Novotny, Nikhila Ravi, Benjamin Graham, Natalia Neverova, and Andrea Vedaldi. C3dpo: Canonical 3d pose networks for non-rigid structure from motion. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 1.1, 2.1, 2.2, 2.3, **??**, **??**, **??**, **??**, 2.5.4, 3.1, 3.2, 3.3.2, 3.5, **??**, **??**, 3.5, **??**, **??**, **??**, **??**, 3.7.4, **??**

[144] John Oliensis and Richard Hartley. Iterative extensions of the sturm/triggs algorithm: Convergence and nonconvergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2217–2233, 2007. 3.2

[145] Marco Paladini, Alessio Del Bue, João Xavier, Lourdes Agapito, Marko Stošić, and

Marija Dodig. Optimal metric projections for deformable and articulated structure-from-motion. *International journal of computer vision*, 96(2):252–276, 2012. 3.2

[146] Vardan Papyan, Yaniv Romano, and Michael Elad. Convolutional neural networks analyzed via convolutional sparse coding. *The Journal of Machine Learning Research*, 18(1):2887–2938, 2017. 3.4.2, 4.3

[147] Shaifali Parashar, Adrien Bartoli, and Daniel Pizarro. Self-calibrating isometric non-rigid structure-from-motion. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 252–267, 2018. 3.1

[148] Hyun Soo Park, Takaaki Shiratori, Iain Matthews, and Yaser Sheikh. 3d reconstruction of a moving point from a series of 2d projections. In *Eur. Conf. Comput. Vis.*, 2010. 6.2

[149] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Deformable neural radiance fields. *arXiv preprint arXiv:2011.12948*, 2020. (document), 1.2.2, 6.2, 6.6.2, 7.1, 7.1, 7.2, 7.2, 7.3.3, 7.4.1, 7.4.2, 7.4.2, 7.4.2, **??**, 7.8, 7.6.3

[150] 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. *ACM Transactions on Graphics (TOG)*, 40(6):1–12, 2021. 7.2, 7.4.2, **??**

[151] Sungheon Park, Minsik Lee, and Nojun Kwak. Procrustean regression networks: Learning 3d structure of non-rigid objects from 2d annotations. In *European Conference on Computer Vision*, pages 1–18. Springer, 2020. 2.1, 2.2, 2.5.2, **??**, **??**, 2.5.4, 2.5.5

[152] Georgios Pavlakos, Xiaowei Zhou, and Kostas Daniilidis. Ordinal depth supervision for 3d human pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 4.2

[153] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Coarse-to-fine volumetric prediction for single-image 3d human pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7025–7034, 2017. 4.2

[154] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Harvesting multiple views for marker-less 3d human pose annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6988–6997, 2017. 4.2

[155] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7753–7762, 2019. (document), 3.10

[156] Eric Penner and Li Zhang. Soft 3D reconstruction for view synthesis. 2017. 6.2

[157] 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 *Computer Vision and Pattern Recognition*, 2016. 7.4.2

[158] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 7.4.2, 8.3

[159] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. *arXiv preprint arXiv:2011.13961*, 2020. 6.1, 7.1, 7.2

[160] Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Reconstructing 3d human pose from 2d image landmarks. In *European Conference on Computer Vision*, pages 573–586. Springer, 2012. 4.2

[161] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020. 6.3.3, 7.4.2, 7.4.2

[162] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4058–4066, 2016. 5.2, 5.5.4

[163] 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. 7.6.1

[164] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. DeRF: Decomposed radiance fields. *arXiv preprint arXiv:2011.12490*, 2020. 6.2, 7.2

[165] Helge Rhodin, Mathieu Salzmann, and Pascal Fua. Unsupervised geometry-aware representation for 3d human pose estimation. In *The European Conference on Computer Vision (ECCV)*, September 2018. 4.1, 4.2, **??**

[166] Helge Rhodin, Jörg Spörri, Isinsu Katircioglu, Victor Constantin, Frédéric Meyer, Erich Müller, Mathieu Salzmann, and Pascal Fua. Learning monocular 3d human pose estimation from multi-view images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 4.1, 4.2

[167] JH Rick Chang, Chun-Liang Li, Barnabas Poczos, BVK Vijaya Kumar, and Aswin C Sankaranarayanan. One network to solve them all–solving linear inverse problems using deep projection models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5888–5897, 2017. 3.4.2

[168] Gernot Riegler and Vladlen Koltun. Stable view synthesis. *arXiv preprint arXiv:2011.07233*, 2020. 6.2

[169] Christopher J Rozell, Don H Johnson, Richard G Baraniuk, and Bruno A Olshausen. Sparse coding via thresholding and local competition in neural circuits. *Neural computation*, 20(10):2526–2563, 2008. 3.4, 3.4.2

[170] Christopher J Rozell, Don H Johnson, Richard G Baraniuk, and Bruno A Olshausen. Sparse coding via thresholding and local competition in neural circuits. *Neural computation*, 20(10):2526–2563, 2008. 4.3

[171] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):824–840, 2009. 5.2

[172] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5.2, 5.5.4

[173] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016. 7.3

[174] 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. 5.1, 5.2, 5.5

[175] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. GRAF: Generative radiance fields for 3D-aware image synthesis. 2020. 6.1

[176] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3D photography using context-aware layered depth inpainting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 6.4, **??**

[177] Vikramjit Sidhu, Edgar Tretschk, Vladislav Golyanik, Antonio Agudo, and Christian Theobalt. Neural dense non-rigid structure from motion with latent space constraints. In *European Conference on Computer Vision (ECCV)*, 2020. 2.1, 2.4.2

[178] Xibin Song, Peng Wang, Dingfu Zhou, Rui Zhu, Chenye Guan, Yuchao Dai, Hao Su, Hongdong Li, and Ruigang Yang. Apollocar3d: A large 3d car instance understanding benchmark for autonomous driving, 2018. 4.1

[179] Xibin Song, Peng Wang, Dingfu Zhou, Rui Zhu, Chenye Guan, Yuchao Dai, Hao Su, Hongdong Li, and Ruigang Yang. Apollocar3d: A large 3d car instance understanding benchmark for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5452–5462, 2019. 3.5

[180] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 6.2

[181] Carsten Steger. Algorithms for the orthographic-n-point problem. *Journal of Mathematical Imaging and Vision*, 60(2):246–266, 2018. 2.4.2, 2.4.2

[182] Peter Sturm and Bill Triggs. A factorization based algorithm for multi-image projective structure and motion. In *European conference on computer vision*, pages 709–720. Springer, 1996. 3.2, 3.2, 3.3

[183] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. *arXiv preprint arXiv:1902.09212*, 2019. (document), 3.5, 3.5

[184] Xiao Sun, Bin Xiao, Fangyin Wei, Shuang Liang, and Yichen Wei. Integral human pose regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 529–545, 2018. 4.2, 4.4.4, 4.5.1, **??**, 4.5.2, 4.5.2, 4.5.4

[185] Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P Srinivasan, Jonathan T Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations. *arXiv preprint arXiv:2012.02189*, 2020. 6.2

[186] Chengzhou Tang and Ping Tan. Ba-net: Dense bundle adjustment network. *arXiv preprint arXiv:1806.04807*, 2018. 3.4.2

[187] Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. *arXiv preprint arXiv:1812.04605*, 2018. 7.2

[188] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *Eur. Conf. Comput. Vis.*, 2020. 6.3.3, 7.3

[189] Zachary Teed and Jia Deng. Tangent space backpropagation for 3d transformation groups. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 7.6.1

[190] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. Nießner, R. Pandey, S. Fanello, G. Wetzstein, J.-Y. Zhu, C. Theobalt, M. Agrawala, E. Shechtman, D. B Goldman, and M. Zollhöfer. State of the Art on Neural Rendering. 2020. 6.1

[191] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. In *Computer Graphics Forum*, 2020. 6.1

[192] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International journal of computer vision*, 9(2):137–154, 1992. 2.2

[193] Denis Tome, Chris Russell, and Lourdes Agapito. Lifting from the deep: Convolutional 3d pose estimation from a single image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. **??**, 4.1, 4.2, **??**

[194] Lorenzo Torresani, Aaron Hertzmann, and Chris Bregler. Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors. *IEEE transactions on*

*pattern analysis and machine intelligence*, 30(5):878–892, 2008. **??**, 2.5.3, **??**, **??**, 6.2

[195] 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 deforming scene from monocular video. *arXiv preprint arXiv:2012.12247*, 2020. 6.1, 6.2, 6.4, **??**, **??**, 7.1, 7.2, 7.2, 7.4.2, **??**, **??**

[196] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 6.2

[197] Hsiao-Yu Tung, Hsiao-Wei Tung, Ersin Yumer, and Katerina Fragkiadaki. Self-supervised learning of motion capture. In *Advances in Neural Information Processing Systems*, pages 5236–5246, 2017. 4.1, 4.2, **??**

[198] Toshio Ueshiba and Fumiaki Tomita. A factorization method for projective and euclidean reconstruction from multiple perspective views via iterative depth estimation. In *European conference on computer vision*, pages 296–310. Springer, 1998. 3.2

[199] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018. 4.4.2

[200] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *IEEE Conference on computer vision and pattern recognition (CVPR)*, volume 5, page 6, 2017. (document), 5.1, 5.2, 5.4.1, 5.5.3, **??**, 5.5.4, 5.4

[201] Jack Valmadre and Simon Lucey. General trajectory prior for non-rigid reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2012. 6.1, 6.2, 6.3

[202] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *CVPR*, 2017. 2.5.5

[203] Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 109–117, 2017. 4.2

[204] Sara Vicente and Lourdes Agapito. Soft inextensibility constraints for template-free non-rigid reconstruction. In *European conference on computer vision*, pages 426–440. Springer, 2012. 3.1

[205] Bastian Wandt and Bodo Rosenhahn. Repnet: Weakly supervised training of an adversarial reprojection network for 3d human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7782–7791,

2019. 2.2, 2.4.2, 3.2, **??**

[206] Bastian Wandt and Bodo Rosenhahn. Repnet: Weakly supervised training of an adversarial reprojection network for 3d human pose estimation. *CoRR*, abs/1902.09868, 2019. 4.2

[207] Chaoyang Wang, Ben Eckart, Simon Lucey, and Orazio Gallo. Neural trajectory fields for dynamic novel view synthesis. *arXiv preprint arXiv:2105.05994*, 2021. 1.2.2, 7.2, 7.4.2

[208] Chaoyang Wang, Hamed Kiani Galoogahi, Chen-Hsuan Lin, and Simon Lucey. Deep-lk for efficient adaptive object tracking. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 627–634. IEEE, 2018. 3.4.2

[209] Chaoyang Wang, Chen Kong, and Simon Lucey. Distill knowledge from nrsfm for weakly supervised 3d pose learning. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 1.2.2, 3.1, **??**

[210] Chaoyang Wang, Xueqian Li, Jhony Kaesemodel Pontes, and Simon Lucey. Neural prior for trajectory estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6532–6542, June 2022. 1.2.2, 7.2

[211] Chaoyang Wang, Chen-Hsuan Lin, and Simon Lucey. Deep nrsfm++: Towards 3d reconstruction in the wild. *arXiv preprint arXiv:2001.10090*, 2020. 1.2.2, 2.2, **??**, 2.5.4, 2.6

[212] Chaoyang Wang and Simon Lucey. Paul: Procrustean autoencoder for unsupervised lifting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 434–443, 2021. 1.2.2

[213] Chaoyang Wang, Simon Lucey, Federico Perazzi, and Oliver Wang. Web stereo video supervision for depth prediction from dynamic scenes. 2019. 1.2.2, 6.3.3, 7.4.2

[214] Chaoyang Wang, Lachlan Ewen MacDonald, Laszlo A Jeni, and Simon Lucey. Flow supervision for deformable nerf. *arXiv preprint arXiv:2303.16333*, 2023. 1.2.2

[215] Chaoyang Wang, José Miguel Buenaposada, Rui Zhu, and Simon Lucey. Learning depth from monocular videos using direct methods. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2022–2030, 2018. 4.4

[216] Chaoyang Wang, José Miguel Buenaposada, Rui Zhu, and Simon Lucey. Learning depth from monocular videos using direct methods. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 5.2, 5.5.4, **??**

[217] Guanghui Wang, Hung-Tat Tsui, and Zhanyi Hu. Structure and motion of nonrigid object under perspective projection. *Pattern recognition letters*, 28(4):507–515, 2007.

3.2, 3.3

[218] Guanghui Wang, Hung-Tat Tsui, and Q. M. Jonathan Wu. Rotation constrained power factorization for structure from motion of nonrigid objects. *Pattern Recognition Letters*, 29:72–80, 2008. 3.2

[219] Qianqian Wang, Zhengqi Li, David Salesin, Noah Snavely, Brian Curless, and Janne Kontkanen. 3d moments from near-duplicate photos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3906–3915, June 2022. 7.2

[220] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 7.4.2

[221] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF−−: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021. 6.6.3

[222] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. SynSin: End-to-end view synthesis from a single image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 6.2

[223] Jiajun Wu, Tianfan Xue, Joseph J Lim, Yuandong Tian, Joshua B Tenenbaum, Antonio Torralba, and William T Freeman. Single image 3d interpreter network. In *European Conference on Computer Vision*, pages 365–382. Springer, 2016. **??**, 4.1, 4.1, 4.2, **??**, **??**

[224] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. D2 nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. *arXiv preprint arXiv:2205.15838*, 2022. 7.3.3

[225] Ke Xian, Chunhua Shen, Zhiguo Cao, Hao Lu, Yang Xiao, Ruibo Li, and Zhenbo Luo. Monocular relative depth perception with web stereo data supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 311–320, 2018. (document), 5.1, 5.2, 5.4, 5.4.1, 5.4.2, **??**, 5.5.1, **??**, 5.5.4, 5.3

[226] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. *arXiv preprint arXiv:2011.12950*, 2020. 6.1, 6.2, 7.3.3

[227] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision*, pages 75–82. IEEE, 2014. 2.5.4, 3.5

[228] Jing Xiao and Takeo Kanade. Uncalibrated perspective reconstruction of deformable structures. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1075–1082. IEEE, 2005. 3.2

[229] Junyuan Xie, Ross Girshick, and Ali Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *European Conference on Computer Vision*, pages 842–857. Springer, 2016. 5.2

[230] 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. *Advances in Neural Information Processing Systems*, 34, 2021. 6.6.1, 7.2

[231] Gengshan Yang, Minh Vo, Natalia Neverova, Deva Ramanan, Andrea Vedaldi, and Hanbyul Joo. Banmo: Building animatable 3d neural models from many casual videos. *arXiv preprint arXiv:2112.12761*, 2021. 6.6.1, 7.1, 7.2

[232] Wei Yang, Wanli Ouyang, Xiaolong Wang, Jimmy Ren, Hongsheng Li, and Xiaogang Wang. 3d human pose estimation in the wild by adversarial learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 4.2

[233] Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, and Ram Nevatia. Lego: Learning edge with geometry all at once by watching videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 225–234, 2018. 5.2

[234] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 6.2, 6.4, 6.4, 6.8e, 7.4.2, 7.6.1

[235] Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Tianyou Liang, Guanying Chen, Shuguang Cui, and Xiaoguang Han. Mvimgnet: A large-scale dataset of multi-view images. In *CVPR*, 2023. 8.3

[236] Amir Zadeh, Yao-Chong Lim, Paul Pu Liang, and Louis-Philippe Morency. Variational auto-decoder. *arXiv preprint arXiv:1903.00840*, 2019. 2.7.1

[237] Andrei Zanfir, Elisabeta Marinoiu, and Cristian Sminchisescu. Monocular 3d pose and shape estimation of multiple people in natural scenes - the importance of multiple scene constraints. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 4.2

[238] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 340–349, 2018. 5.2

[239] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. NeRF++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 6.2

[240] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

7.4.2

[241] Yan Zhang, Michael J Black, and Siyu Tang. We are more than our joints: Predicting how 3D bodies move. *arXiv preprint arXiv:2012.00619*, 2020. 6.3

[242] Zhoutong Zhang, Forrester Cole, Richard Tucker, William T Freeman, and Tali Dekel. Consistent depth of moving objects in video. *ACM Transactions on Graphics (TOG)*, 40(4):1–12, 2021. 7.1

[243] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N. Metaxas. Semantic graph convolutional networks for 3d human pose regression. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. **??**

[244] Yinqiang Zheng, Yubin Kuang, Shigeki Sugimoto, Kalle Astrom, and Masatoshi Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2344–2351, 2013. 2.6, 3.1, 3.2, 3.3

[245] 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 *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pages 492–509. Springer, 2020. 7.2

[246] Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. Deeptam: Deep tracking and mapping. In *European Conference on Computer Vision (ECCV)*, 2018. 5.2

[247] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1851–1858, 2017. 4.4, 5.2, 5.5

[248] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018. 5.2, 6.2

[249] Xiaowei Zhou, Menglong Zhu, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis. Sparseness meets deepness: 3d human pose estimation from monocular video. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4966–4975, 2016. 2.2, 3.2

[250] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue, and Yichen Wei. Towards 3d human pose estimation in the wild: a weakly-supervised approach. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 398–407, 2017. 4.1, 4.2, 4.5.2

[251] Yingying Zhu, Dong Huang, Fernando De La Torre, and Simon Lucey. Complex non-rigid motion 3d reconstruction by union of subspaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1542–1549,

2014. 2.1, 2.2, 3.2, 4.2, 5.2

[252] Yingying Zhu and Simon Lucey. Convolutional sparse coding for trajectory reconstruction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013. 6.1, 6.2

[253] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon A. J. Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. 2004. 6.2

[254] Yuliang Zou, Zelun Luo, and Jia-Bin Huang. Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In *European Conference on Computer Vision*, 2018. 5.2, 5.4.2

[255] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6365–6373, 2017. 7.2