High-Fidelity 3D Mesh Reconstruction of Humans and Objects

Shubham Goel

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2023-254
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2023/EECS-2023-254.html

December 1, 2023

High-Fidelity 3D Mesh Reconstruction of Humans and Objects

by

Shubham Goel

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Jitendra Malik, Co-chair
Professor Angjoo Kanazawa, Co-chair
Professor Ken Goldberg

Spring 2023

High-Fidelity 3D Mesh Reconstruction of Humans and Objects

Abstract

High-Fidelity 3D Mesh Reconstruction of Humans and Objects

by

Shubham Goel

Doctor of Philosophy in Computer Sciences

University of California, Berkeley

Professor Jitendra Malik, Co-chair

Professor Angjoo Kanazawa, Co-chair

Humans perceive the world through their eyes – where the images formed on the retina are two-dimensional projections of the underlying three-dimensional world. Akin to human vision, the goal of computer vision, is to extract information about the 3D world from 2D images. A fundamental problem in computer vision is to extract the 3D structure underlying such 2D images. Even though this problem is mathematically ill-posed, the ambiguity can be resolved, either using multiple 2D views, or using priors about how the world is structured.

In this thesis, I present my work on high-fidelity 3D mesh reconstruction of humans and objects from 2D images. I discuss the more classical setting of optimizing a shape/texture using multiple image inputs, as well as how we can learn priors that enable mesh reconstruction even from a single image. Specifically, I first present work on multi-view 3D reconstruction, where we reconstruct meshes of an object given few images with noisy camera poses. Then, I continue with 3D reconstruction from single images, enabled by learning category-specific shape priors from natural image datasets. Finally, I focus on learning single-view 3D human reconstruction using big models and big data. Such robust 3D reconstruction of humans enables downstream applications like 3D tracking and action recognition.

To Guruji, Mom, Dad, and Naaz,
for your unconditional love and support

Contents

Acknowledgments

This thesis has come to fruition only because of the unwavering support of incredible individuals who have accompanied me on this journey. The acknowledgments that follow can only begin to express my immense gratitude for their invaluable contributions.

I am deeply indebted to my advisors, Jitendra Malik and Angjoo Kanazawa. Without their guidance and support, this journey would not have been possible. Jitendra taught me the art of picking research problems and emphasized "what" to solve is always more important than "how" to solve it. He showed me how to think of long-term research goals, and consistently take small steps toward them. His academic history lessons not only gave me context around seminal research in the field, but also made it a lot more fun. In addition to shaping me as a researcher, conversations with him have had a profound effect on my life, sometimes with the most wonderful quotes. "If it's not worth doing, it's not worth doing fast" lives in my mind rent-free.

Angjoo took a chance on me and mentored me, even when she had no reason to do so, and for that, I'm eternally grateful. She taught me how to do vision research, how to go about solving a problem, how to make pretty figures, and the importance of presenting it well. Her critically honest feedback has been instrumental in my growth, and I sincerely hope to continue receiving it. Her unparalleled excitement and energy also kept me motivated through hard times. How she works so hard while maintaining a healthy lifestyle, never ceases to amaze me, and I will always be inspired by her.

Most importantly, Jitendra and Angjoo are kind at heart and empathetic. To have found advisors like them who put my needs above all else, I feel truly blessed.

With the uniquely open and welcoming culture at Berkeley, I always found it easy to interact with our exceptional faculty members. I thank Alexei (Alyosha) Efros for introducing me to the breathtaking beauty of the California coast through our numerous hikes and camping trips. Your mild skepticism of 3D has continually challenged me, keeping me on my toes, and pushing me to refine and improve my ideas. I'm grateful to Sanjit Seshia for his invaluable guidance during my first year at Berkeley. He challenged me to deeply contemplate my research interests and aspirations, which led me to realize I want to pursue research in 3D Computer Vision. I also thank Trevor Darrell, Ken Goldberg, and Ren Ng, for many thought-provoking research discussions, and for often going out of their way to help me.

In addition to the guidance from these mentors, I had the privilege of learning from some amazing collaborators. The research presented in this thesis wouldn't be possible without their tireless efforts and constant motivation. My deepest thanks go to each of them for not only tolerating my shortcomings but also consistently encouraging me to strive for better. Navigating research during the pandemic was particularly hard for me, yet, Georgia Gkioxari made pandemic research not just possible, but enjoyable! Among many things, I learned from her, that the paper text can also be made very pretty. Georgios Pavlakos has been a mentor, collaborator, friend, and occasionally, even an in-house therapist. I am thankful for his unwavering support during my final years. His work ethic is so incredible, and his dedication so intense, that I suspect he has a twin that takes over the night shift! Jathushan Rajasegaran has selflessly helped me with research, sometimes staying long hours for moral support as a core member of the late-night crew, and sometimes as a formidable

gully-cricket rival when we needed to de-stress. Jasmine Collins was a constant source of calm and composure in our collaborations, even when our research projects seemed to be engulfed in flames, and I am very grateful for that.

Working on research projects was only a part of my journey at Berkeley. The day-to-day excitement and energy I experienced came from the interactions and discussions I had with my incredible labmates and other folks at Berkeley. I'd like to thank Jitendra, Angjoo, Alyosha, and Trevor's amazing research groups, many of whom I interacted with regularly. I want to thank Sasha for consistently grounding me through my highs and lows, and to Ashish for helping me find my footing at Berkeley. I also want to thank Aleks, Allan, Alvin, Anastasios, Andrea, Antonio, Assaf, Bill, Boyi, Dave, Ethan, Evonne, Hang, Haozhi, Ilija, Kartik, Lea, Matt, Medhini, Neerja, Ruilong, Shiry, Taesung, Tete, Tim, Toru, Utkarsh, Vickie, Vongani, Yossi, Yu, and Zhe. From going climbing/hiking/backpacking, to acting as a sounding board for my research ideas, to finding good mangoes in the bay area, to burning the midnight oil together before a deadline, to stocking the lab with my favorite snacks, you've supported me in an immeasurable number of ways. I'd also like to thank folks from Sanjit's group: Eddie, Kevin, Shromona, and Daniel, for guiding me through my first year at Berkeley. Thanks to the BAIR Admin team - Angie, Ami, Roxana, and Lena, for making BWW life pleasant and shielding me from bureaucracy and logistics, often without me even knowing it.

I've been blessed to have a wonderful set of friends who made the journey through graduate school an unforgettable experience. For this, I thank Aditi, Akshit, Anurag, Aviral, Ayush, Bhuvnesh, Chitraang, Divya, Karan, Siddharth, Srajan, Surbhi, Teena, Utkarsha, and Yeshwanth[1]. A special shout out to Sumith Kulal, who has been a lifesaver more times than I can count. I'd also like to thank all my friends outside Berkeley: in Stanford, south bay, the rest of the US, and India. Your companionship and camaraderie have been a source of joy and comfort. Thank you all for being by my side through thick and thin, and transforming Berkeley into a home away from home.

My journey to this point began long before my time at Berkeley, and there are countless individuals from my past who have played a vital role in shaping me both personally and professionally. I'm indebted to my friends, teachers, and professors from IIT Bombay and my high school, who instilled in me a love for learning and a curiosity to explore the world.

To my extended family and family friends, your unwavering faith in me has been a constant source of strength. I am grateful for your enduring love and encouragement that has spurred me on even in the most challenging of times.

To my parents, words fall short to express my gratitude. Your sacrifices, love, and relentless belief in me have been the bedrock of my journey. You have taught me resilience, humility, and the value of hard work, lessons that extend far beyond the realm of academics.

Last but certainly not least, my sister, Naaz. You are a shining beacon of light in my life. Your infectious enthusiasm, relentless optimism, and unwavering support have been invaluable. You have always been there to listen, to advise, and to celebrate each small victory with me. Your tenacity and spirit inspire me every day, and I am immensely grateful for your love and companionship.

[1]Sorry if I forgot to mention someone. Knowing me, this may not surprise you, but I'm writing this a little last-minute :)

Chapter 1

Introduction

We reside in a complex 3D world, and rely on our senses to perceive and interact with our surroundings. One of the most critical senses, vision, is essentially our ability to interpret the images formed in our eyes – 2D projections of the underlying 3D world – to richly perceive the environment around us. This innate capability allows us to navigate and engage with the world effectively. The field of computer vision is aimed at enabling machines to extract and understand information from such visual data, which is predominantly in the form of such 2D images. By teaching machines to interpret and analyze images akin to how humans do, we can equip them to intelligently understand the world and perform tasks that require visual comprehension.

Computer vision can also be perceived as inverting the image formation process – given a 2D image, recover the underlying 3D world. In the first-ever Ph.D. thesis in Computer Vision, Lawrence Roberts [139] tried to lift images of simple block shapes to their 3D structure. Unfortunately, this task of 3D reconstructing from a single image is under-constrained. Each 2D image can be mapped to a multitude of 3D structures, introducing significant ambiguity [39, 146].

Classically, Structure from Motion (SfM) [161] mitigates this ambiguity by leveraging *multiple* 2D images of the *same* 3D structure. Multiple viewpoints provide additional constraints and with enough corresponding points across views, the system becomes over-constrained and we can resolve the ambiguity, enabling extraction of depth and 3D spatial information, up to a scale.

When only a single image is available, the ambiguity remains unresolved. However, note that not all possible 3D reconstructions of a 2D image are equally plausible. There is an inherent structure in the world, in both natural and man-made objects. Natural scenes get structure from the physics of the universe. For example, because of gravity, trees are vertical, the horizon is horizontal, and mountains are thinner at the top. Similarly, man-made objects have structure, as a result of intended functionality, aesthetics, and ease of manufacturing. This includes low-level structure like surfaces being smooth and flat, and high-level structure like all chairs having similar shape and size. This structure imposes priors on both natural and man-made objects, making some 3D reconstructions more likely than others. Humans, through experience, develop an intuition for these priors, which they use to perceive depth and volume from images [149]. Understanding and effectively incorporating these priors into computational models is a key challenge in learning to perceive images in 3D.

In this thesis, I develop techniques that can lift 2D images to the 3D shapes of the underlying entities. The work spans two main areas: multi-view and single-view 3D reconstruction. First, I will delve into the multi-view setting where we leverage multiple images without relying on data-driven priors. Then, I will transition into exploring single-view 3D reconstruction, focusing on the use of learned priors to enable 3D reconstruction from single images, for specific categories such as humans, birds, and objects.

Multi-View 3D Reconstruction First, in Chapter 2, we consider the classical setting where multiple views of the same 3D shape are available. Classical SfM [141] works in this setting but often requires hundreds of views in practice, which have excessive visual overlap across them, leading to a lot of redundancy in the input information. We explore how to operate with a very limited number of images without any data-driven priors. We propose Differentiable Stereopsis (DS), a multi-view stereo approach that reconstructs shape and texture from few input views and noisy cameras. DS pairs traditional stereopsis and modern differentiable rendering to build an end-to-end approach that optimizes and extracts textured 3D meshes of objects with varying topologies and shape. The key ingredient of the approach is to compare images using underlying shape and camera estimates. We optimize shape and cameras to minimize a texture loss associated with un-projecting one image onto the shape and re-rendering from another viewpoint. We show compelling reconstructions on challenging real-world scenes and for an abundance of object types with complex shape, topology and texture.

Shape optimization is typically slow and we cannot always have multiple images of a scene. For example, there are many dynamic entities in our world like humans and animals, and their shape changes every time they move. The only way to capture multiple images of the same underlying shape is synchronized multi-view capture. This is possible in a limited lab setting [66], but not in the wild. Therefore, we need to develop methods that quickly reconstruct 3D structures from single images.

Single-View 3D Reconstruction In Chapters 3 and 4, we show how to use data to learn priors that enable reconstructing shape from single images. In Chapter 3, we learn category-specific priors by leveraging an image collection of a particular category. Since different instances across the category have similar shape, the shape deformations lie on a low-dimensional manifold [154, 12]. We exploit this to learn to recover the 3D shape, pose, and texture from a single image. Notably, we do so without any ground truth 3D shape, multi-view, camera viewpoints, or keypoint supervision, which makes scaling to new categories easier. Our particular contribution in this work is a representation of the distribution over cameras, which we call “camera-multiplex". Instead of picking a point estimate, we maintain a set of camera hypotheses that are optimized during training to best explain the image given the current shape and texture. We call our approach Unsupervised Category-Specific Mesh Reconstruction (U-CMR), and show results on multiple categories like birds, cars, motorcycles, shoes, *etc*. Our state-of-the-art results show that we can learn to predict diverse shapes and textures across objects using an image collection without any keypoint annotations or 3D ground truth. Followup work in the community [2] shows that our approach can be scaled to 150+ categories.

Even though U-CMR allows building shape models for many categories, some categories are special. For example, for humans, we already have shape models available [100, 125]. These models have been built painstakingly in a lab setting by recording humans from multiple viewpoints, often by placing markers on the body. These models of 3D shape and articulation are parametric, and model the full mesh using pose parameters (akin to a skeleton), and shape parameters.

Finally, in Chapter 4, I discuss how to leverage existing parametric human shape models to learn to 3D reconstruct humans in images and track them over video. At the core of our approach, we propose a fully "transformerized" version of a network for human mesh recovery – *i.e*. single-view 3D reconstruction of human shape and pose. To train this network, we scale up dataset size by an order of magnitude using unlabelled image and video datasets, and find pseudo-ground truth via an optimization process involving off-the-shelf 2D keypoint predictors. This model, HMR 2.0, advances the state of the art on reconstructing humans from single images. To analyze video, we use 3D reconstructions from HMR 2.0 as input to a tracking system that operates in 3D. This enables us to deal with multiple people and maintain identities through occlusion events. Our complete approach, 4DHumans, achieves state-of-the-art results for tracking people from monocular video. Furthermore, we demonstrate the effectiveness of HMR 2.0 on the downstream task of action recognition, achieving significant improvements over previous pose-based action recognition approaches.

This thesis makes significant strides in the 3D reconstruction of different entities in the world around us, but a lot of open problems remain to be tackled. For example, even though we know how to learn priors for certain sets of object categories, we still do not know how to learn 3D priors over the entire world. In the concluding Chapter 5, I discuss the current state of affairs in 3D computer vision, and how the ideas presented in this thesis might be used for tackling open problems.

Chapter 2

Differentiable Stereopsis: Meshes from Multiple Views

2.1 Introduction

Binocular stereopsis [169], and its multi-view cousin, Structure from Motion [52, 152], has traditionally been formulated as a two stage process:

1. Find corresponding 2D points across views, which are the 2D projections of the *same* 3D scene point.

2. Recover relative orientations of cameras, and the depths of these points by triangulation.

In this work, we bypass the first stage of finding point correspondences across images and directly estimate 3D shape and cameras given multiple 2D views with noisy cameras. We formulate this as an optimization problem that we solve using newly developed differentiable rendering tools. We name our approach *Differentiable Stereopsis*.

Our approach is linked to old work in multi-view geometry, and in particular model-based stereopsis which was explored by Debevec *et al*. [26] and related ideas in plane plus parallax by Irani *et al*. [60]. The key observation in model-based stereo is simple: two images of the same scene which appear different become similar after projection onto an approximate 3D model of the scene. Projecting the texture from one image onto the 3D model produces a warped version of that view which when transformed from a second view is directly comparable to the second image. Initially, the 3D model and the estimated relative camera orientation are inaccurate. But as shape and camera predictions improve, the two images will start to look more similar and will eventually become identical – in the idealized case of Lambertian surfaces and no imaging noise. Upon convergence, the shape is expected to be an accurate representation of the scene.

This chapter is based on joint work with Georgia Gkioxari and Jitendra Malik [46], and is presented much as it appeared in the CVPR 2022 proceedings.

Figure 2.1: Reconstructions with Differentiable Stereopsis (DS) from few input views and noisy cameras. We show input views (left) and novel views (right) of reconstructions for two instances.

Algorithm 1: Differentiable Stereopsis (2-view)

Input: $I_{1,2}$, $\pi_{1,2}$;
$S \leftarrow$ Sphere();
while *not converged* **do**
  points $\leftarrow$ `rasterize`(S, π_1);
  texels $\leftarrow$ `sample`(I_2, π_2(points));
  $I_1^r \leftarrow$ `blend`(texels);
  loss $\leftarrow$ `compute_loss`(I_1^r, I_1);
  $S, \pi_{1,2} \leftarrow S, \pi_{1,2}-$ lr * `gradient`(loss);
end
Outputs S, $\pi_{1,2}$

An important step in traditional stereopsis is finding 2D correspondences across views. We bypass this and directly recover shape and cameras using modern optimization techniques. We frame stereopsis as an optimization problem by minimizing a differentiable objective with respect to shape and cameras. To this end, we exploit advances in *differentiable rendering* [101, 75, 98, 18, 135] to project shape and texture onto image planes which we compare to scene views. We update shape and cameras via gradient descent. Algorithm 1 illustrates our proposed Differentiable Stereopsis (DS) for the case of 2 views. We rely on (i) object masks to isolate and refine object topology; and (ii) noisy camera pose initializations, which may still come from a correspondence matching algorithm. Fig. 2.1 shows shape reconstructions with DS when noisily posed views are given as input.

At the core of our approach is a novel and differentiable texture transfer method which pairs rendering with the key insight of texture warping via 3D unprojections. Our texture transfer learns to sample texture from the input views based on the shape estimate and noisy cameras. To allow for differentiation, it composites the final texture in a soft manner by weighing texture samples

proportionally to their visibility and direction from each view.

We test our approach on challenging datasets and on a large variety of objects with complex and varying shapes. Unlike prior works that assume several dozens of object views, our experimental settings follow real-world practical scenarios where only a few views are available. For example, Amazon, eBay or Facebook Marketplace only contain a handful of views for each listed item, and any 3D reconstruction has to originate from 10 views or less. We emphasize on this harder, yet more realistic, setting and show empirical results with real product images from Amazon [24]. On Google's Scanned Objects [137] we perform an extensive quantitative and qualitative analysis and compare to competing approaches under settings similar to ours. We also show results on Tanks and Temples [79] which contains RGB views of complex scenes, as shown on the right in Fig. 2.1.

2.2 Related Work

Extracting 3D structure from 2D views of a scene is a long standing goal of computer vision. Classical multi-view stereo methods and Structure from Motion techniques [52, 152, 26, 40, 60] find correspondences across images and triangulate them into points in 3D space. The resulting point clouds, if dense enough, can be meshed into surfaces [10, 77]. The culmination of a long line of classical SfM and stereo approaches is COLMAP [141, 142] – a widely used tool for estimating camera poses and reconstructing dense point clouds from 2D input views. All aforementioned techniques assume calibrated and accurate cameras and thus are not very robust to camera noise.

Finding point correspondences, the first stage of stereopsis, is challenging especially in the case of sparse widely-separated views. Debevec *et al*. [26] tackle this by proposing model-based stereopsis wherein a coarse scene geometry allows views to be placed in a common reference frame, making the correspondence problem easier. We draw inspiration from this work and pair it with new learning tools to reconstruct textured 3D meshes from sparse views. We frame stereopsis as an optimization problem and minimize a differentiable objective which allows both shape and cameras to self-correct. This increases robustness to camera noise, in antithesis to classical techniques.

Recent work on multi-view stereo [182, 179] train deep neural nets (DNNs) with depth supervision. As expected, these methods outperform COLMAP for point cloud reconstruction but are limited as they need ground truth. We rely solely on image re-projection losses and no true depth information.

Work on unsupervised depth prediction [198, 92, 186, 196] estimate depth via DNNs trained on monocular videos and without ground truth depth. They exploit photometric and depth consistency across multiple views, much like classical stereo. However, they focus on forward-facing scenes like KITTI [42] and do not reconstruct high-fidelity shape.

There is extensive work on recovering shape from images using differentiable rendering [75, 101, 98, 18, 135, 71, 47, 90, 158, 86, 185]. These approaches focus on extracting object priors by training on large datasets and test on images of seen categories. We also use differentiable rendering [75, 101, 98, 18, 135] to frame stereopsis as a differentiable optimization problem. Differentiating with respect to shape and camera allows for both to self-correct during optimization.

Most relevant to our work are methods that learn shape by fitting to a set of images. Early work on extracting shape from silhouettes used a visual hull [88]. Gadelha *et al*. [41] reconstruct voxels from silhouettes and noisy camera poses via differentiable projection but don't use any texture information. However, shape details such as concavities cannot be captured by silhouettes. We show in Fig. 2.5 in our experiments (Sec. 2.4) that optimizing for shape without texture information fails to reconstruct creases in shape. Some variational approaches for MVS [36, 130, 56, 28] exploit photometric consistency to refine shape via gradient descent but they require many images, initial shapes or accurate cameras. Recently, IDR [183] and DVR [115] recover shape from multiple posed images and masks using implicit volumetric representations. IDR shows superior results to DVR and claims to work with few input views and slightly noisy camera poses; a setting similar to ours. We compare to IDR in Sec. 2.4.

Recent novel-view-synthesis approaches [197, 99, 110] encode volumetric occupancy information in their internal representations for the task of image synthesis from novel viewpoints. While they don't explicitly learn shape, their representation can be processed to extract geometry. NeRF [110] is one such approach which takes posed multiple views as input and encodes occupancy and color for points in 3D space as an implicit function. In Sec. 2.4, we compare to NeRF and extend it with a variant that optimizes for noisy cameras by enabling backpropagation to its parameters.

2.3 Approach

We tackle the problem of stereopsis using modern differentiable rendering techniques. Our approach takes N image views $I_{1..N}$ of an object with corresponding masks $A_{1..N}$ and *noisy* camera poses $\pi_{1..N}$ as input, and outputs the shape of the object as a textured mesh. We frame stereopsis as an optimization problem, outlined in Fig. 2.2. We iteratively render a shape estimate from multiple cameras using differentiable textured rendering and update the shape and cameras by minimizing image reprojection losses.

We first provide some background on differentiable rendering and then describe our approach in detail.

Background

We define a textured mesh $M = (V, F, T)$ as a set of vertices V, faces F and a texture map T. Under a camera viewpoint π, mesh M is rendered to image $I^r = R_T(M, \pi)$ and mask $A^r = R_S(M, \pi)$, where R_T denotes textured rendering and R_S silhouette rendering.

Both R_S and R_T perform mesh rasterization. Rasterization computes which parts of the mesh are projected to a pixel on the image plane. For each pixel p, we find the K nearest faces that intersect with a ray originated at p [135].

In the case of silhouette rendering R_S, rasterization is followed by a soft silhouette shader. This shader assigns each pixel an occupancy probability by blending the euclidean distance of the pixel to each of the K faces [98, 135].

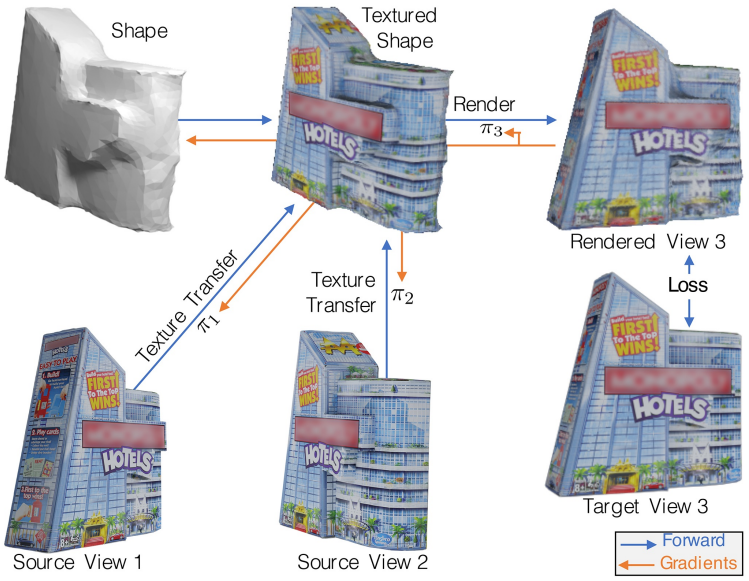

Figure 2.2: Overview of Differentiable Stereopsis (DS). The shape estimate is textured using source views, rendered from a target view's camera and compared against the target view. The loss is backpropagated to update shape and cameras.

For textured rendering R_T, we use a texture shader which computes the RGB color for each pixel p in the image. This shader blends the colors from the top K faces for each pixel, as computed by the rasterizer. For the k-th face, the color $c_k = T(x)$ is computed by sampling the texture map T at the point of intersection x of the ray originating at p and the k-th face. The set of colors $c_{1..K}$, also known as *texels*, are composited to get the final color for the pixel.

Texture Transfer

The goal of our approach is to find $M = (V, F, T)$ that represents the object as seen from the noisily posed input views. For each shape hypothesis (V, F) we need to find the optimal texture T. We introduce a *novel* texture shader which relies on texture transfer from the inputs $I_{1...N}$.

Our shader computes the texture map T as a function of the shape hypothesis (V, F) and the posed input views $I_{1...N}$. The texture map $T : x \rightarrow (r, g, b)$ assigns an RGB color for each point x on the mesh surface. The color is directly sampled from one or more input views. We build on a key insight: for a correct shape (V, F) and correct cameras $\pi_{1...N}$, there exists one (or many) view i in which x is unoccluded, or in other words, there is a clear line-of-sight to x. For all such views, the projections $\pi_i(x)$ in the images correspond to the same 3D point x and for Lambertian surfaces, all these points will share the same color $I_i(\pi_i(x))$. The color $T(x)$ assigned to point x is composited from the colors $I_i(\pi_i(x))$ for all views with a clear line-of-sight to x. Formally, we define the texture transfer as follows:

$$T(x) = \sum_i w_i I_i(\pi_i(x)) \tag{2.1}$$

where weights are unit-normalized and defined as $w_i = \sigma_i \gamma_i$.

σ encodes whether x has a clear line-of-sight from the corresponding view. Formally, we compare the z-distance of the camera transformed point $\pi_i(x)$ to the rendered depth map D_i at $\pi_i(x)$ as follows

$$\sigma_i = \exp(-(\pi_i(x)_z - D_i(\pi_i(x)))/\tau_{\text{vis}}) \tag{2.2}$$

If there is a clear line-of-sight to x, then $\pi_i(x)_z \approx D_i(\pi_i(x))$ and thus $\sigma_i \approx 1.0$. If x is obstructed by other parts of the shape, then $\pi_i(x)_z > D_i(\pi_i(x))$ and $\sigma_i < 1.0$. We set the temperature τ_{vis} to 10^{-4}.

γ is a heuristic that favours views that look at x fronto-parallel with minimal foreshortening. If $\hat{n}_i(x)$ is the outward surface-normal at x in i-th view coordinates, then

$$\gamma_i = \mathbb{1}[\hat{n}_i(x)_z < 0] \exp(-(1 + \hat{n}_i(x)_z)/\tau_{\text{cos}}) \tag{2.3}$$

γ_i is highest when the normal points opposite the camera's z-axis, or $\hat{n}_i(x)_z = -1$. γ_i decreases exponentially as $\hat{n}_i(x)_z$ increases. We set the temperature τ_{cos} to 0.1. In addition, we cull backward-facing normals ($\hat{n}_i(x)_z > 0$) to correctly sample texture in thin surfaces where σ fails to capture visibility information of points on the two sides of the surface.

Texture Rendering We described how to sample texture for a point x on the mesh surface. To render the texture under a viewpoint π, for each pixel p we sample texels $c_{1..K}$ for all points $x_{1..K}$ with $c_k = T(x_k)$, where x_k is the point on the k-th face that intersects the ray originating at p. We use softmax blending [98] to composite the final color at p.

Optimization

We have explained how to define the texture map T for an object shape (V, F) given posed input views $I_{1..N}$ and we have described how to render $M = (V, F, T)$ to images and silhouettes. We now describe our objective and how we optimize it w.r.t. vertices V and cameras $\pi_{1..N}$.

Parametrization We parametrize a camera $\pi = (r, t, f)$ as rotation via an axis-angle representation $r \in \mathbb{R}^3$ (magnitude $|\mathbf{r}|$ is angle, normalization $\mathbf{r}/|\mathbf{r}|$ is axis), translation as $t \in \mathbb{R}^3$ and focal length f as half field-of-view.

We parametrize geometry as $V = V_0 + \Delta V$ where $\Delta V \in \mathbb{R}^{|V| \times 3}$ is the deformation being optimized and V_0 are initial mesh vertices which remain constant.

Objective Given a shape hypothesis $M = (V, F, T)$, cameras $\pi_{1..N}$ and input views $I_{1..N}$, we render silhouette $A_i^r = R_S(M, \pi_i)$ and image $I_i^r = R_T(M, \pi_i)$ for each view $i = 1, ..., N$. We define our total loss to be

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{tex}} + \mathcal{L}_{\text{mask}} + \mathcal{L}_{\text{edge}} + \mathcal{L}_{\text{lap}} \tag{2.4}$$

The texture reconstructions loss $\mathcal{L}_{\text{tex}}$ is defined as the sum of an $\mathcal{L}_1$ loss and perceptual distance metric $\mathcal{L}_{\text{perc}}$ [194]:

$$\mathcal{L}_{\text{tex}} = \sum_i |I_i^r - I_i| + \mathcal{L}_{\text{perc}}(I_i^r, I_i) \tag{2.5}$$

The mask reconstruction loss combines an MSE loss and a bi-directional distance transform loss (see details in Appendix).

$$\mathcal{L}_{\text{mask}} = \sum_i ||A_i^r - A_i||_2^2 + \mathcal{L}_{\text{bi-dt}}(A_i^r, A_i) \tag{2.6}$$

In addition to reprojection losses in Eq. 2.5 & 2.6, we employ smoothness reguralizers on the mesh: $\mathcal{L}_{\text{edge}} = ||E - l||_2^2$ is an MSE loss penalizing edge lengths that deviate from the mean initial edge length l, while $\mathcal{L}_{\text{lap}} = ||L_{\text{cot}}V||_2$ is a cotangent-laplacian loss that minimizes mean curvature [113].

Initialization and Warmup We initialize cameras with the noisy input cameras, V_0 with an ico-sphere and ΔV with zeros. During an initial warmup phase of 500 iterations, we freeze cameras and optimize shape without the texture loss. We start with a very low-resolution sphere and subdivide it twice during warmup, at 100 and 300 iterations respectively.

Texture Sampling We compute the texture map T after each shape update during optimization. For each training view i, and for each pixel p, we find the K closest faces intersecting a ray originating at p and the corresponding points of intersection $x_{1..K}$. We compute texels $c_k = T(x_k)$, described in Sec. 2.3, and set $w_i = 0$ in Eq. 2.1 so that image I_i does not contribute to the texture for pixel p in the rendered i-th view. This ensures that image I_i^r for camera π_i is generated by sampling colors from all images $I_{1..N}$ but I_i to encourage photometric consistency.

Handling Variable Topology Each gradient descent step updates the vertex positions of the mesh and the camera parameters. However, the topology of the shape is left intact. To handle objects with varying topology and to deviate from shapes homeomorphic to spheres, we update the topology of our shapes during optimization. At intermediate steps during training, we voxelize our mesh [111, 116], project voxels onto the view plans and check for occupancy by comparing to the ground truth silhouettes $A_{1...N}$. We remove voxels that project to an unoccupied area in any mask. We re-mesh the remaining voxels using marching cubes, reset all shape-optimization parameters and resume optimization.

2.4 Experiments

We test our differentiable stereopsis approach, which we call DS, on three datasets: Google's Scanned Objects [137], Tanks and Temples [79] and the Amazon-Berkeley Objects [24]. We additionally evaluate on DTU MVS [63] in the Appendix. We run extensive quantitative analysis on objects of varying topology and shapes, for which 3D ground truth is available. We also show qualitative results on real objects and challenging real-world scenes.

Experiments on Google's Scanned Objects

Google's Scanned Objects (CC-BY 4.0) [137] consists of 1032 common household objects that have been 3D scanned to produce high-resolution Lambertian textured 3D meshes. From these, we pick 50 object instances with varying shape, topology and texture for quantitative analysis including

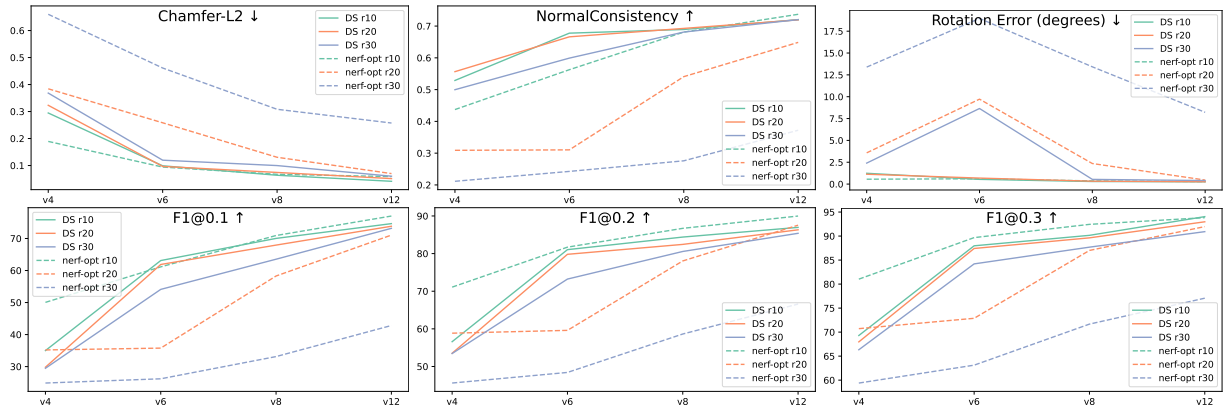

Figure 2.3: Performance of DS and nerf-opt on Google's Scanned Objects with varying number of views $\{4, 6, 8, 12\}$ (x-axis) and camera noise $\{10°, 20°, 30°\}$. Each plot reports the median across the 50 objects in our evaluation set. We report shape reconstruction metrics (Chamfer, F1), normal consistency, and camera error. For Chamfer and camera error, lower is better. For everything else, higher is better.

toys, electronics, instruments, appliances, cutlery and many more. For each object, we render 2048×2048 RGBA images from 12 random camera viewpoints. Camera rotation Euler angles and field-of-view are uniformly sampled in $[0°, 360°]$ and $[20°, 50°]$ respectively. To the cameras, we add rotation noise $\theta \sim \mathcal{N}(0, \sigma^2)$ about a uniformly sampled axis with varying $\sigma = \{10°, 20°, 30°\}$.

Metrics We report a variety of metrics to quantitatively compare the predicted with the ground-truth meshes. We use $\mathcal{L}_2$-Chamfer distance, normal consistency and F1 score at different thresholds, following [45]. Since predicted shapes don't lie in the same coordinate frame as the ground-truth, we align predictions to ground truth before benchmarking via the iterative-closest-point (ICP) algorithm [11]. See Appendix for more details. Lastly, we report the rotation error (in degrees) between the ground truth and the output cameras from DS optimization.

Comparison with baselines We extensively compare to NeRF [110], as the state-of-the-art volumetric method, which learns an implicit function from accurately posed input views. While NeRF doesn't explicitly output shape, we extract geometry from its implicit representation via voxelization and run marching cubes to get a mesh. We compare to two NeRF variants which use additional mask information: (a) *nerf* - the original NeRF approach with an additional MSE loss on rendered masks, and (b) *nerf-opt* - which is the same as (a) but optimizes camera poses with gradients from the reprojection loss. nerf-opt uses the same camera parametrization as our approach. To prevent NeRF from collapsing due to large areas of white background in the input views, all NeRF baselines sample 50% of their points inside the mask in every iteration. We also compare qualitatively to IDR [183], a volumetric method with an implicit representation that learns geometry and appearance from sparse wide-baseline images and masks with noisy camera poses. In the Appendix, we also compare to COLMAP [141, 142] as the state-of-the-art

photogrammetry approach. Finally, we compare to variants of our approach: (a) *DS-notex*, which does not use any texture information removing $\mathcal{L}_{\text{tex}}$ from Eq. 2.4; and (b) *DS-naive*, which naively optimizes a UV texture image in addition to shape/camera instead of using our texture-transfer. For texturing, the texture image is mapped to the mesh surface using a fixed UV map [57] that is automatically computed with Blender [13]. Whenever the mesh topology changes, the texture image is re-initialized and the UV-map recomputed.

Fig. 2.3 quantitatively compares DS to nerf-opt, the best performing NeRF variant of the two. We train with varying number of views $N = 4, 6, 8, 12$ (x-axis) and varying camera noise $\{10°, 20°, 30°\}$. Each plot reports the median across the 50 instances selected from the dataset. For small camera noise ($10°$), DS and nerf-opt (green lines) achieve comparable Chamfer and F1, except for $N = 4$ views where nerf-opt achieves higher F1. Undoubtedly, predicting shape from 4 views is challenging for all methods, as indicated by the absolute performance, and is the only setting where nerf-opt performs better than DS. For larger camera noise ($20°$), DS performs better than nerf-opt (orange lines) under all metrics for $N \geq 6$ and on par for $N = 4$. For even larger camera noise ($30°$), DS leads by a significant margin (blue lines) for all N and all metrics. We note that as the number of views N increases, both methods converge roughly to the same performance for $10°$ & $20°$ noise. For $30°$ noise, DS also converges to the above optimum with increasing views N. On the other hand, nerf-opt is unable to recover shape or cameras for $30°$ noise and achieves much lower reconstruction quality. These results prove that DS can learn better shapes and recover cameras even under larger camera noise and fewer views. When given slightly more views, DS reaches the same reconstruction quality as with little camera noise proving its robustness to errors in cameras. See Appendix for quantitative comparisons to IDR, COLMAP, and DS-naive.

Fig. 2.4 qualitatively compares nerf, nerf-opt, IDR and DS with 8 views and $30°$ noise. IDR and both NeRF variants produce shapes with cloudy artifacts, with nerf-opt visibly outperforming nerf. DS captures better shape under the same settings proving its robustness to noisy cameras and few views. We observe that in a few-view wide-baseline setting, like ours, implicit volumetric approaches attempt to explain the few input views without relying on accurate shape geometry and appearance. However, meshes, which explicitly represent surfaces, offer stronger surface regularization and predict more precise geometry. Also, in the first row of Fig. 2.4 we observe that DS is able to reconstruct different animals as disconnected components despite having been initialized to a single sphere.

Fig. 2.5 compares DS to DS without texture (*DS-notex*) and naive texture map optimization (*DS-naive*) with 8 input views and $20°$ camera noise. DS-notex fails to capture shape concavities, which are impossible to capture via just silhouettes. DS-naive results in shapes with some concavities but in the wrong place and of shape quality similar to DS-notex. With naive texture optimization as in DS-naive, texture converges to the mean texture from different images, providing unreliable gradients to improve shape/cameras and leading to suboptimal geometry. In contrast, DS accurately captures creases in object shapes by exploiting texture.

Fig. 2.9 shows qualitative results on Google's Scanned Objects. For each object, we train with 8 input views and $20°$ camera noise. We show the input views (left) and the output shape and texture for two novel views (right).

Fig. 2.7 shows the evolution of shape with time for two examples from Google's Scanned

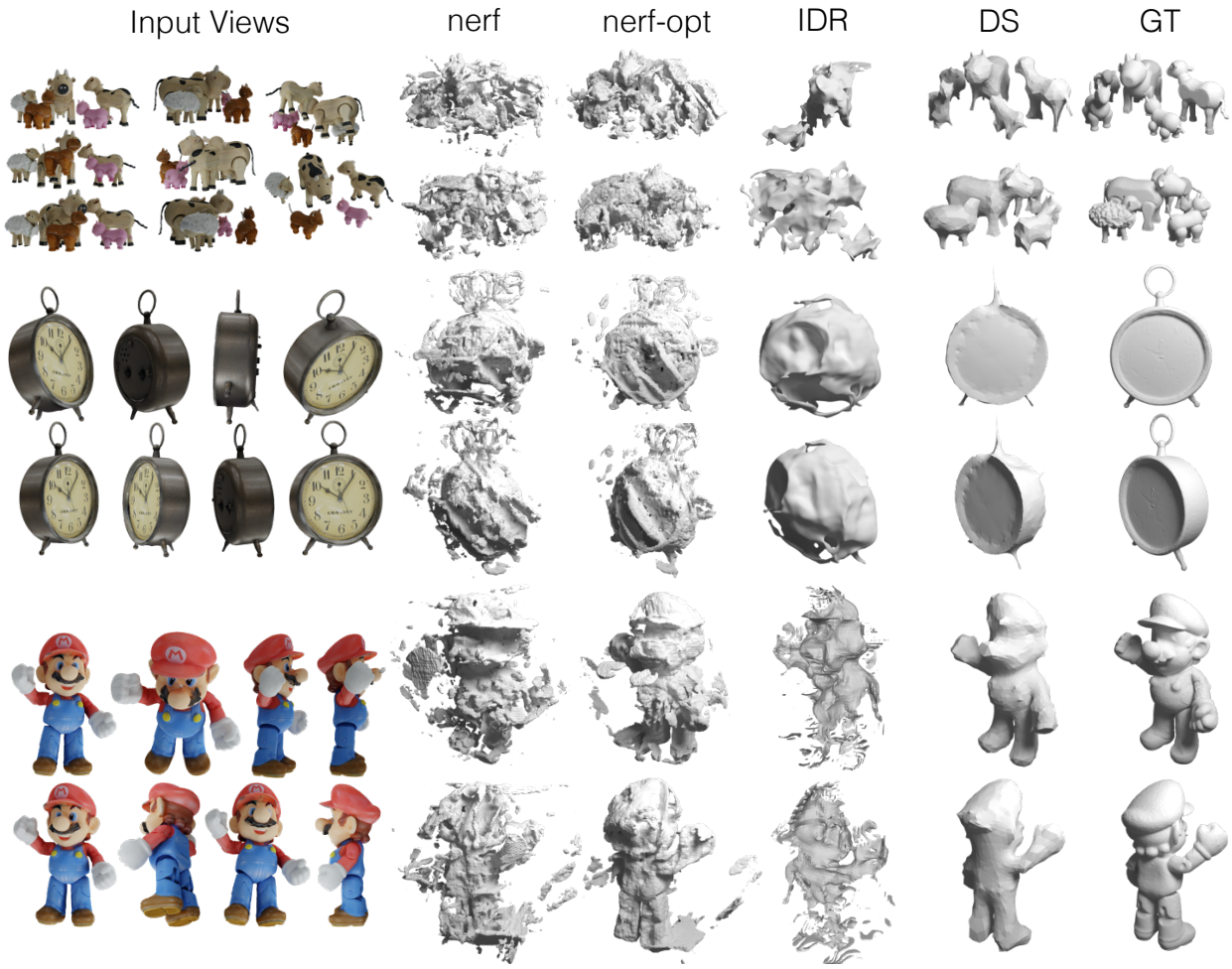

Figure 2.4: Results of nerf, nerf-opt, IDR and DS with 8 views and 30° camera noise. nerf-opt and IDR outperform nerf but they fail to capture good shape. DS captures better geometry illustrating its robustness to high levels of camera noise.

Objects with 8 input views and 20° camera noise. We remesh the shape by updating its topology at three intermediate steps during optimization. This brings the final shape close to the ground truth both in terms of geometry and topology.

We also show failure modes in the Appendix.

Results on Amazon Products

We show results from images of 6 real-world objects from the ABO dataset (CC-BY-NC 4.0) [24]. Pixel-thresholding the white-background images gives masks. Camera poses for these images are unknown and COLMAP fails to give sensible estimates. We get rough initial cameras by manually annotating a set of 40 keypoint correspondences across all images of an object. We estimate the parameters for a weak-perspective camera for each image using a orthographic rigid-body

factorization formulation [107] adopted in [71, 73, 165]. We initialize our perspective cameras using the computed weak-perspective cameras and assume a 30° field-of-view.

Fig. 2.6 shows shape and texture reconstructions. Despite very noisy cameras and few views, ranging from 4 to 9, our approach reconstructs shape and texture reasonably well even for challenging shape topologies like the lawnmower. We also note that DS is able to reconstruct more specular surfaces like the wristwatch in the last row.

Results on Tanks and Temples

Tanks and Temples (CC-BY-NC-SA 3.0) [79] is a 3D reconstruction benchmark consisting of RGB videos of indoor and outdoor scenes with corresponding laser-scanned ground-truth 3D point clouds. The dataset comes with cameras computed by COLMAP's SfM pipeline [141]. We evaluate on 7 scenes using *only 15* input images and corresponding SfM-reconstructed cameras as initialization. For *Barn*, *Ignatius*, *Caterpillar* and *Truck*, we generate masks by rendering the 3D point clouds from SfM-reconstructed cameras. To stress test our approach without relying on 3D point clouds to get masks, for *Horse*, *Family* and *Train* we use an off-the-shelf object detector [78] pretrained on COCO [96].

Fig. 2.8 shows reconstructions for scenes from Tanks and Temples with 15 input views and SfM-reconstructed cameras. DS is able to produce good reconstructions and undoubtedly has a harder time for *Barn* due to occlusions by trees. For *Family* and *Train*, detected masks are poor leading to bad reconstructions. In the Appendix, we compare to IDR, NeRF-opt, and COLMAP.

2.5 Discussion

We propose Differentiable Stereopsis (DS) by pairing traditional model-based stereopsis with modern differentiable rendering. We show results on a diverse set of object shapes with noisy cameras and few input views. Even though DS performs well, it has limitations. It assumes Lambertian surfaces and consistent lighting. DS works for objects – extending to complex scenes is

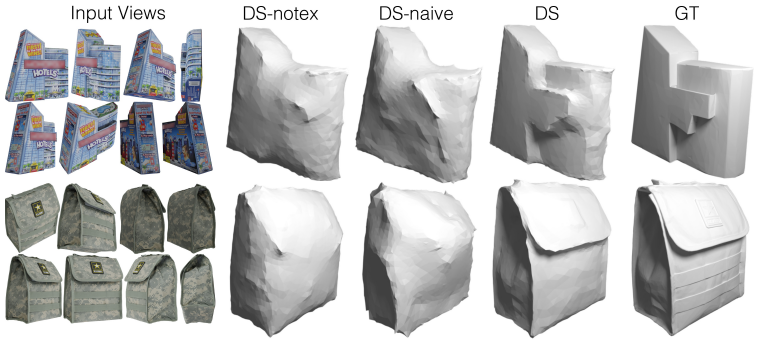

Figure 2.5: DS without texture (DS-notex), DS-naive and DS with 8 views and 20° camera noise. DS-notex fails to capture shape concavities, while DS-naive fails to recover accurate shape and cameras. The ground truth shape (GT) is shown in the last column.

Figure 2.6: DS evaluated on real-world product images from Amazon [24]. For each example, we show input views (left) and reconstructed shape and texture for novel views (right).

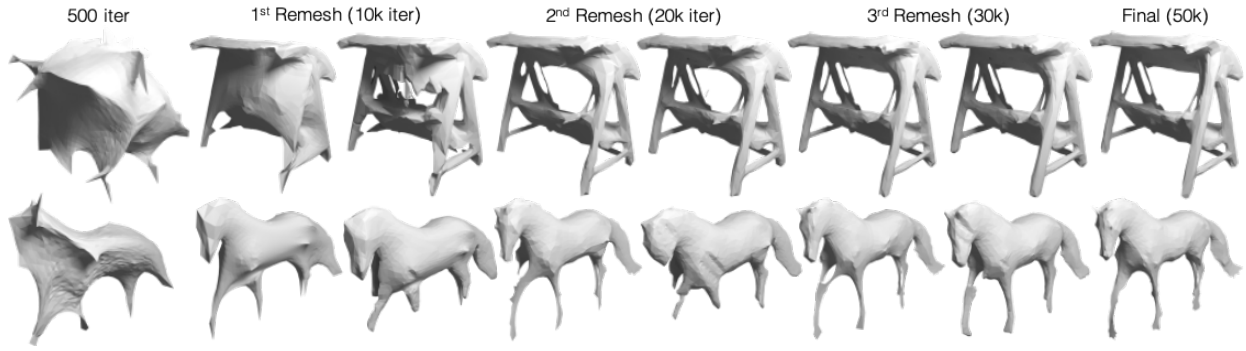

Figure 2.7: Evolution of shape over time for *Garden Swing* (top) and *Breyer Horse* (bottom) from Google's Scanned Objects with 8 views and 20° camera noise. We visualize shape at key optimization steps: at the end of warmup (at 500 iterations), before and after the $1^{\text{st}}/2^{\text{nd}}/3^{\text{rd}}$ remesh (at 10k/20k/30k iterations) and final shape (at 50k iterations).

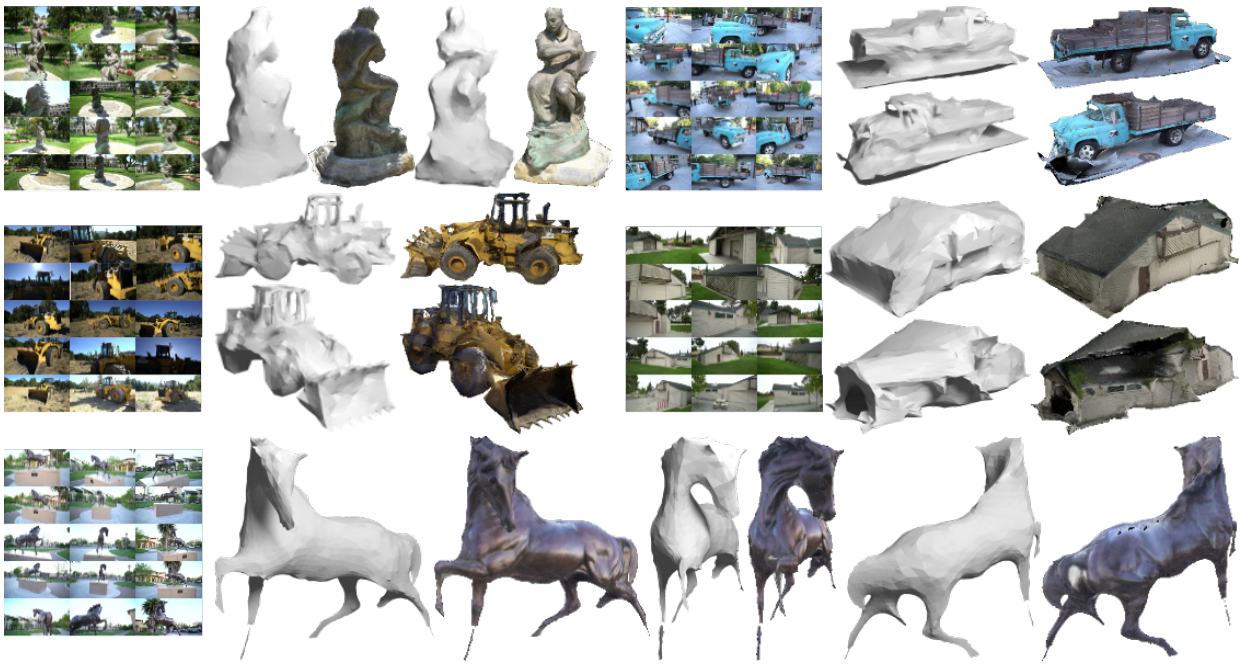

Figure 2.8: Reconstructions of DS on *Ignatius*, *Truck*, *Caterpillar*, *Barn* and *Horse* from Tanks and Temples with 15 input views and SfM-generated camera poses. For each example, we show input views (left), shape and texture reconstructions from two novel views (right). Silhouettes for *Horse* were generated by a pretrained off-the-shelf 2D object detector.

future work. While DS is robust to noisy masks (*e.g.* predictions from Mask R-CNN) and inaccurate cameras provided at input, eliminating them from the input alltogether is important future work.

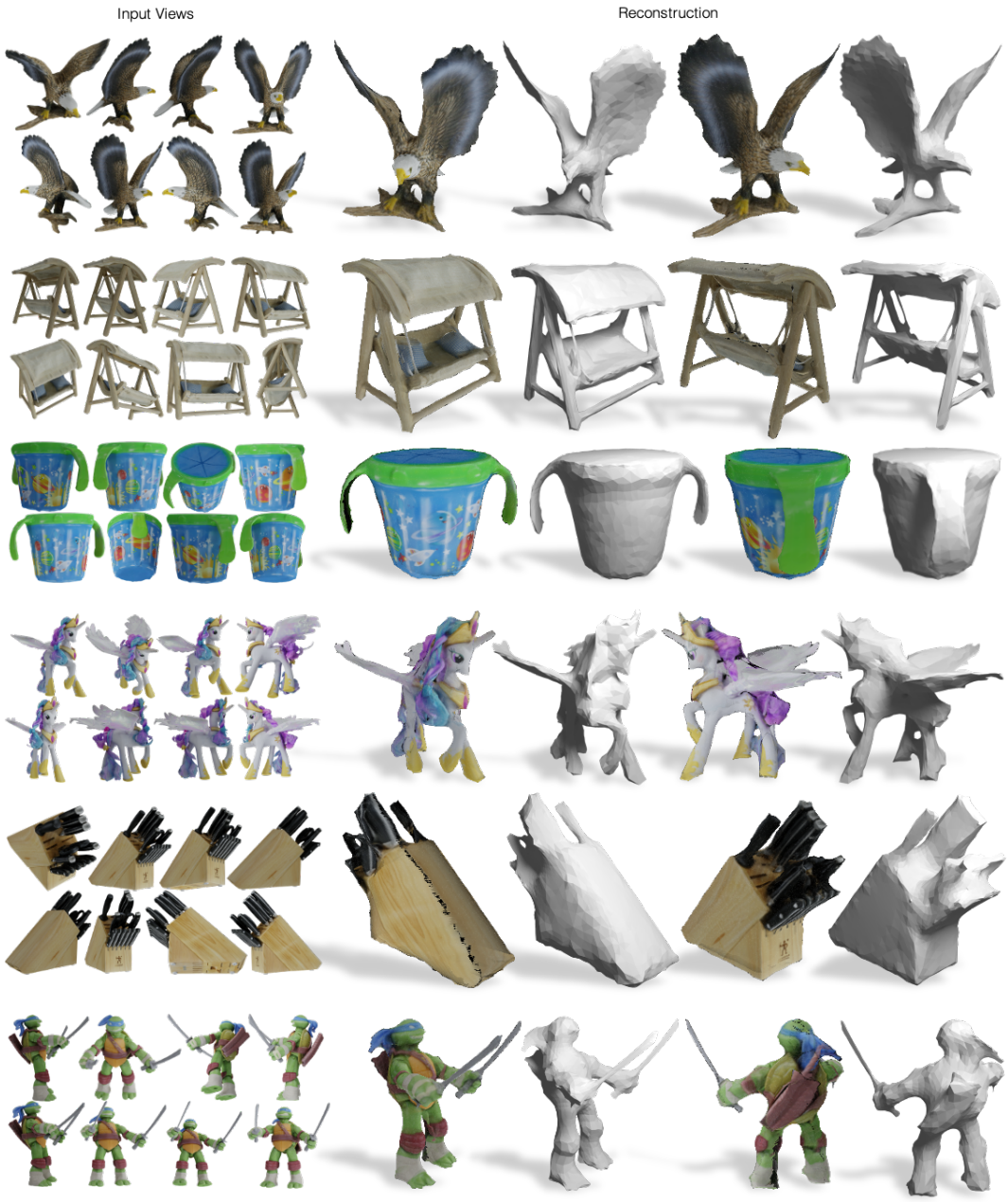

Figure 2.9: Qualitative results of DS on Google's Scanned Objects with 8 input views and 20° camera noise. We show input views (left) and reconstructed shape and texture from two novel views (right).

Chapter 3

Shape and Viewpoint without Keypoints

3.1 Introduction

There has been much progress in recent years in training deep networks to infer 3D shape from 2D images. These approaches fall into two major families based on the supervisory signal used (a) 3D models, as available in collection of CAD models such as ShapeNet or (b) multiple views of the same object which permit deep learning counterparts of classical techniques such as shape carving or structure-from-motion. But do we need all this supervision?

It is easy to find on the internet large collections of images of objects belonging to particular categories, such as birds or cars or chairs. Let us focus on birds, for which datasets like CUB [166], shown in Figure 3.1 (left) exist. Note that this is a “Multiple Instance Single View" setting. For each bird instance we have only a single view, and every bird is a slightly different shape, even though the multiple instances share a family resemblance. Compared to classical SFM, where we have “Single Instance Multiple Views", our goal is to “3Dfy" these birds. From a single image, create a 3D model and its texture map, which can then be rendered from different camera viewpoints as shown in the rows of Figure 3.1 (right). This particular formulation was presented in the “Category-Specific Mesh Reconstruction" work of Kanazawa *et al*. [70], and their algorithm (CMR) is an inspiration for our work. Even earlier the work of Cashman and Fitzgibbon [17] working on analyzing images of dolphins showed how an “analysis by synthesis" paradigm with a deformable template model of a shape category could enable one to infer 3D shapes in an optimization framework.

It is under-appreciated that these approaches, while pioneering, do exploit some supervisory information. This includes (1) knowledge of a mean shape for the category (2) silhouettes for each instance (3) marked keypoints on the various instances (e.g. beak tip for each bird). Of these, the need for labeled keypoints is the most troublesome. Only one mean shape is needed for the entire category and sometimes a very generic initialization such as a sphere is good enough. Silhouettes

This chapter is based on joint work with Angjoo Kanazawa and Jitendra Malik [47], and is presented much as it appeared in the ECCCV 2020 proceedings.

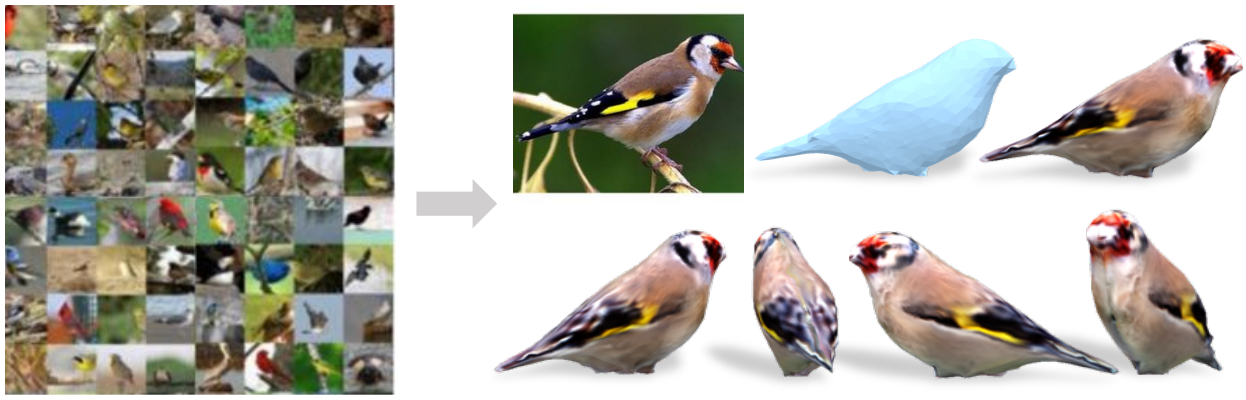

Figure 3.1: Given an image collection of an object category, like birds, we propose a computational framework that given a single image of an object, predicts its 3D shape, viewpoint and texture, without using any 3D shape, viewpoints or keypoint supervision during training. On the right we show the input image and the results obtained by our method, shown from multiple views.

could presumably be marked (perhaps in a category-agnostic way) by an instance segmentation network like Mask R-CNN. But keypoints are tedious to mark on every instance. This effort can be justified for a single important category like humans, but we cannot afford to do for the thousands of categories which we might want to "3Dfy". Yes, we could have the keypoints be marked by a network but to train that would require keypoint labels! There have been recent efforts in unsupervised keypoint-detection [62], however, so far, these methods learn viewpoint-dependent keypoints that often get mixed-up in the presence of 180-degree rotations.

In this chapter, we present an approach, U-CMR (for Unsupervised CMR) which enables us to train a function which can factor an image into a 3D shape, texture and camera, in a roughly similar setting to CMR [70], except that we replace the need for keypoint annotation with a single 3D template shape for the entire category. It turns out that keypoint annotations are needed for recovering cameras (using SFM like techniques) and if we don't have keypoint annotations, we have to solve for the camera simultaneously with shape and texture. Extending the "analysis by synthesis" paradigm to also recover cameras is unfortunately rather hard. Intuitively speaking, this is because of the discontinuous and multi-modal nature of the space of possible camera viewpoints. While shape and texture have a smooth, well behaved optimization surface that is amenable to gradient descent optimization, this does not hold for the space of possible cameras. The two most likely camera explanations might lie on the opposite sides of the viewing sphere, where it is not possible to approach the optimal camera in an iterative manner as done when optimizing the energy landscape of a deep network. This typically causes the camera prediction to be stuck in bad local minima. Our solution to this problem is to maintain a set of possible camera hypotheses for each training instance that we call a **camera-multiplex**. This is reminiscent of particle filtering approaches with the idea being to maintain a distribution rather than prematurely pick a point estimate. We can iteratively refine the shape, texture as well as the camera-multiplex. More details are presented in Section 3.3.

We evaluate our approach on 3D shape, pose, and texture reconstruction on 4 categories: CUB-200-2011 birds [166], PASCAL-3D [174] cars, motorcycles and a new dataset of shoes we scraped from the Internet. We show that naively predicting shape, camera, and texture directly results in a degenerate solution where the shape is flat and the camera collapses to a single mode. We quantitatively evaluate the final camera prediction where the proposed camera-multiplex approach obtains the state of the art results under this weakly-supervised setting of no keypoint annotations. We show that despite the lack of viewpoints and keypoints, we can learn a reasonable 3D shape space, approaching that of the shapes obtained by a previous method that uses keypoint supervision [70].

3.2 Related Work

Recent deep learning based 3D reconstruction methods can be categorized by the required supervisory signal and the output of the system. This is illustrated in Table 1. Earlier methods formulate the problem assuming full 3D shape supervision for an image [23, 44, 34, 159, 45], which is enabled by synthetic datasets such as ShapeNet [173] and SunCG [148]. Some approaches generate synthetic datasets using data gathered from the real world to train their models [19, 162, 200]. However, requiring 3D supervision severely restricts these approaches, since ground truth 3D shape is costly or not possible to acquire, especially at a large scale. As such, follow up methods explore more natural forms of supervision, where multiple-views of the same object are available. Some of these approaches assume known viewpoints [180, 72, 160] akin to the setting of traditional MVS or visual hull. Other approaches explore the problem with unknown viewpoint setting [157, 58, 41]. These approaches assume that multi-view silhouettes, images, and/or depth images are available, and train their models such that the predicted 3D shapes reconstruct the images after projection or differentiable rendering. A variety of differentiable rendering mechanisms have been explored [101, 75, 98].

While multi-view images may be obtained in the real world, the vast amount of available visual data corresponds to the setting of unconstrained collection of single-view images, where no simultaneous multiple views of the same instance are available. This is also the natural setting for non-rigid objects where the shape may change over time. The traditional non-rigid structure from motion [156] also falls under this category, where the input is a tracked set of corresponding points [156, 25] or 2D keypoints [165, 117]. Earlier approaches fit a deformable 3D model [12, 73, 17, 68] to 2D keypoints and silhouettes. Kanazawa *et al*. [70] propose CMR, a learning based framework where 3D shape, texture, and camera pose are predicted from a single image, trained under this setting of single-view image collections with known mask and keypoint annotations. While this is a step in the right direction, the requirement of keypoint annotation is still restrictive. More recently, Kulkarni *et al*. [85] bypass this requirement of keypoints to learn a dense canonical surface mapping of objects from a set of image collections with mask supervision and a template 3D shape. They focus on predicting the surface correspondences on images and learn to predict the camera viewpoints during the training, but do not learn to predict the 3D shape. While we tackle a different problem, we operate under the same required supervision. As such we quantitatively compare with

Approach	Required Supervision per Image					Output			
	3D Shape	Multi-view	Cam	Keypoints	Mask	3D Shape	2.5D	Cam	Texture
MeshRCNN* [45]	✗					✓			
DeepSDF [120]	✗					✓			
Smalst [200]	✗		✗	✗	✗	✓		✓	✓
PTN [180]		✗	✗					✓	
MVC [157]		✗				✓		✓	
CMR [70]			✗	✗	✗	✓		✓	✓
CSM [85]					✗			✓	
Wu *et al*. [172]							✓	✓	✓
U-CMR					✗	✓		✓	✓

Table 3.1: A comparison of different approaches highlighting the differences between the input (during training) and the output (during inference). Our approach (U-CMR) uses only silhouette supervision but predicts full 3D shape, camera and texture. *MeshRCNN predicts shape in camera-coordinates instead of a canonical frame.

CSM on the quality of the camera predicted, where our approach obtains considerably better camera predictions. Note that there are several recent approaches that explore disentangling images into 2.5D surface properties, camera, and texture of the visible regions from a collection of monocular images, without any masks [153, 145, 172]. However these approaches are mainly demonstrated on faces. In this work we recover a full 3D representation and texture from a single image.

3.3 Approach

Preliminaries

Shape Representation. We represent 3D shape as a mesh $M \equiv (V, F)$ with vertices $V \in \mathbb{R}^{|V|\times 3}$ and faces F. The set of faces F defines the connectivity of vertices in the mesh and we assume it remains fixed. We choose a mesh topology that is homeomorphic to a sphere. We model the vertex positions of a deformable object as $V = \Delta_V + \bar{V}$, the summation of an instance-specific deformation Δ_V that is predicted from an image to a learned instance-independent mean shape $\bar{V}$ [70]. We initialize the mean shape with the template 3D mesh. This parameterization allows the model to learn the space of possible deformations for each category.

Texture Representation. As the topology of our mesh is fixed, we can use a UV image I^{uv} representation to model the texture. The values in a UV image get mapped onto the surface via a fixed UV mapping. The UV mapping is either a spherical projection akin to unrolling a globe into a flat map [57], or when that is not good enough, is a distortion-minimizing unwrap of a template mesh along manually defined seams computed using blender [13].

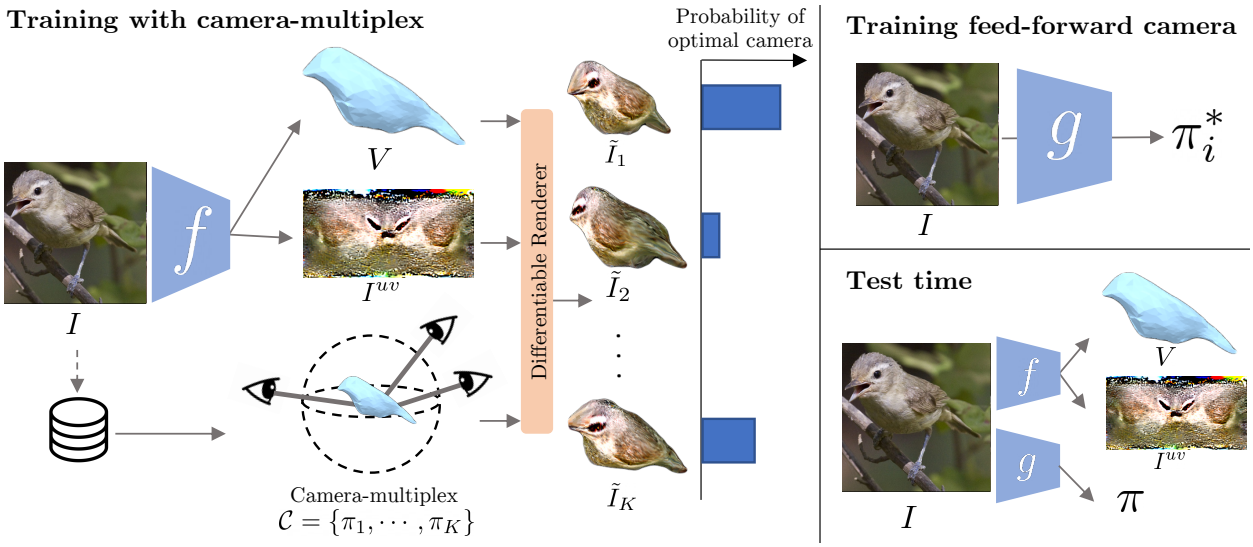

Figure 3.2: **Overview of the proposed framework.** We first train a shape and texture predictor f, while simultaneously optimizing the camera-multiplex $\mathcal{C}$, a set of K possible camera hypotheses maintained for every image. We render the predicted shape and texture from every camera in the multiplex, compute a per-camera reconstruction loss, treat it as negative log-likelihood of the camera, and update the f against the expected loss. We also update each camera in the multiplex against the loss incurred by it. After training f, we train a feed-forward model g to predict the best camera π^* in the multiplex from an image. As such, at test time our approach is able to predict all shape, texture, and camera from a single image.

Camera Projection. We assume a weak-perspective camera projection, parametrized by scale $\mathbf{s} \in \mathbb{R}$, translation $\mathbf{t} \in \mathbb{R}^2$ and rotation $\mathbf{R}$ (captured as Euler angles azimuth, elevation, cyclo-rotation $[az,el,cr] \in \mathbb{R}^3$). We use $\pi(P)$ to denote the projection of a set of 3D points P onto the image coordinates via the weak-perspective projection defined by $\pi \equiv (s, \mathbf{t}, \mathbf{R})$. We denote the image rendered by composing all three factors as $\tilde{I} = \mathcal{R}(V, I^{uv}, \pi)$ and silhouette rendered just from the shape and camera as $\tilde{S} = \mathcal{R}(V, \pi)$ where $\mathcal{R}(\cdot)$ is a differentiable renderer. We denote a set of camera hypotheses kept for each image, a camera-multiplex $\mathcal{C} = \{\pi_1, \cdots, \pi_K\}$. We describe its training details below.

Our Method

Figure 3.2 shows an overview of our approach. During training, we learn a function $f(I)$ to predict the 3D shape and texture of the object underlying image I. We optimize over the camera-multiplex for each instance in the training dataset instead of making a deterministic prediction. For every shape and texture prediction, we compute the loss from every camera in the camera-multiplex, which induces a distribution on the camera poses inside a multiplex. We then use the expected loss over the camera-multiplex to update $f(I)$. When the training of $f(\cdot)$ converges, we identify the optimal camera for each training example in the camera-multiplex. We then train a function $g(I)$

that predicts the optimal camera from a single image, such that at test time we can infer all shape, texture, and camera from a single image. We provide the details for the training process below.

For each training instance I, let $\mathcal{C} = \{\pi_1, \cdots, \pi_K\}$ denote its camera-multiplex with K cameras and S its silhouette. Note that while we omit the subscript on training instances for brevity, *every* instance maintains its own $\mathcal{C}$ independently. For every predicted shape $V = \bar{V} + \Delta V$ and texture I^{uv}, we compute the silhouette and image reconstruction loss against each camera π_k:

$$\mathcal{L}_{\text{mask},k} = ||S - \tilde{S}_k||_2^2 + \texttt{dt}(S) * \tilde{S}_k, \tag{3.1}$$

$$\mathcal{L}_{\text{pixel},k} = \texttt{dist}(\tilde{I}_k \odot S, I \odot S,), \tag{3.2}$$

$\mathcal{L}_{\text{mask},k}$ is the silhouette loss where $\tilde{S}_k = \mathcal{R}(V, \pi_k)$ is the silhouette rendered from camera π_k, and $\texttt{dt}(S)$ is the uni-directional distance transform of the ground truth silhouette. $\mathcal{L}_{\text{pixel},k}$ is the image reconstruction loss computed over the foreground regions where $\mathcal{R}(V, I^{uv}, \pi_k)$ is the rendered image from camera π_k. For this, we use the perceptual distance metric of Zhang *et al*. [194]. To exploit the bilateral symmetry and to ensure symmetric texture prediction, we also render the mesh under a bilaterally symmetric second camera, and compute $\mathcal{L}_{\text{pixel},k}$ as the average pixel loss from the two cameras.

In addition to these losses, we employ a graph-laplacian smoothness prior on our shape $\mathcal{L}_{\text{lap}} = ||V_i - \frac{1}{|N(i)|}\sum_{j \in N(i)} V_j||^2$ that penalizes vertices i that are far away from the centroid of their adjacent vertices $N(i)$. For cars, motorcycles and shoes, we empirically observe better results using $\mathcal{L}_{\text{lap}} = ||LV||_2$ where L is the discrete Laplace-Beltrami operator that minimizes mean curvature [129]. For this, we construct L once using the template mesh at the start of training. Following [12, 17, 73], we also find it beneficial to regularize the deformations as it discourages arbitrarily large deformations and helps learn a meaningful mean shape. The corresponding energy term is expressed as $\mathcal{L}_{\text{def}} = ||\Delta_V||_2$.

Model Update.

For iteratively refining the camera-multiplex, we use the summation of the silhouette and image reconstruction loss $\mathcal{L}_{\pi_k} = \mathcal{L}_{\text{mask},k} + \mathcal{L}_{\text{pixel},k}$ as the loss for each camera π_k in the camera-multiplex. We optimize each camera to minimize L_k every time the training instance is encountered during the training. For updating the shape and the texture, we use the resulting losses over the cameras as a distribution over the most likely camera pose in the camera-multiplex, and minimize the expected loss over all the cameras. Specifically, we compute the probability of π_k being the optimal camera through a softmin function $p_k = \frac{e^{-\mathcal{L}_k/\sigma}}{\sum_j e^{-\mathcal{L}_j/\sigma}}$ and train the shape and texture prediction modules with the final loss:

$$\mathcal{L}_{\text{total}} = \sum_k p_k(\mathcal{L}_{\text{mask},k} + \mathcal{L}_{\text{pixel},k}) + \mathcal{L}_{\text{def}} + \mathcal{L}_{\text{lap}}. \tag{3.3}$$

In practice, the temperature σ changes dynamically while computing p_k by linearly normalizing $\mathcal{L}_k$ to have a fixed range to standardize the peakiness of the probability distribution. We do not

backpropagate through p_k. In summary, we iteratively refine the cameras in the multiplex against loss $\mathcal{L}_{\pi_k}$ and update the parameters of f through $\mathcal{L}_{\text{total}}$ for every training sample.

We implement the camera multiplex for each image $\mathcal{C}_i$ as a variable stored in a dictionary. Every time an image is encountered during training, the corresponding camera multiplex is fetched from this dictionary of variables and used as if it were an input to the rest of the training pipeline. Modern deep learning frameworks such as PyTorch [121] support having such a dictionary of variables.

Training a feed-forward camera predictor.

When the training of f converges, for each training image we select the optimal camera to be the camera that minimizes the silhouette and image reconstruction losses. We then train a new camera prediction module $g(I)$ in a supervised manner such that at inference time our model can predict all 3D shape, texture, and camera at the same time.

Approach at test time.

Given a novel image I at test time, we can use the learnt modules f and g to predict the 3D shape, texture and camera-viewpoint of the object underlying image I. $f(I)$ predicts shape $V = \bar{V} + \Delta_V$ and texture I^{uv} while $g(I)$ predicts the camera-viewpoint π. This is illustrated in Figure 3.2.

3.4 Experiments

In this section we provide quantitative and qualitative evaluation of our approach that learns to predict 3D shape, texture, and camera from a single image without using any keypoint annotations during training. We explore our approach on four object categories: birds, cars, motorcycles and shoes.

Experimental Detail

Datasets. We primarily use the CUB-200-2011 dataset [166], which has 6000 training and test images of 200 species of birds. In addition to this, we train and evaluate U-CMR on multiple categories: car, motorcycles from the Pascal3D+ dataset and shoes scraped from zappos.com. For CUB and Pascal3D, we use the same train-test splits as CMR [70]. For the initial meshes for birds and cars, we use the 3D template meshes used by Kulkarni *et al*. [85]. For others, we download freely available online models, homogenize to a sphere and simplify to reduce the number of vertices. We symmetrize all meshes to exploit bilateral symmetry. We compute masks for the zappos shoes dataset, which contains white background images, via simple threshold-based background subtraction and hole-filling.

Architecture. For all but texture, we use the same architecture as that of CMR [70] and pass Resnet18 features into two modules - one each for predicting shape and texture. The shape prediction module is a set of 2 fully connected layers with $\mathbb{R}^{3|V|}$ outputs that are reshaped into Δ_V following

[70]. For the texture prediction, prior work predicted flow, where the final output is an offset that indicates where to sample pixels from. In this work we directly predict the pixel values of the UV image through a decoder. The texture head is a set of upconvolutional layers that takes the output of Resnet18 preserving the spatial dimensions. We find that this results in a more stable camera, as the decoder network is able to learn a spatial prior over the UV image. We use SoftRas [98] as our renderer. Please see the supplementary material for details and ablation studies.

Camera-multiplex implementation. We use $K = 40$ for camera-multiplex. We initialize the camera multiplex $\mathcal{C}$ for every image in the training set, to a set of K points whose azimuth and elevation are uniformly spaced on the viewing sphere. For cars and motorcycles, we use $K = 8$ cameras - all initialized to zero elevation. We optimize each camera in the multiplex using the silhouette loss $L_{mask,k}$ before training shape and texture. To reduce compute time while training shape and texture, we reduce K from 40 to 4 after 20 epochs by pruning the camera-multiplex and keeping the top 4 cameras. Note that naive data augmentation that scales and/or translates the image without adjusting the camera-multiplex will result in the rendered shape being pixel-unaligned. We handle random crop and scale data augmentation during training by adjusting the scale and translation in the stored camera multiplex with a deterministic affine transformation before using it for rendering the shape.

Baselines. As no other approach predicts 3D shape, texture and pose without relying on keypoints or known camera or multi-view cues during training, as baseline we compare with ablations of our approach that do not use the camera-multiplex. This can be thought of CMR without keypoints, which simultaneously predicts shape, camera and texture and only supervises rendered silhouette and texture. We call this approach CMR-nokp and ensure that the experimental setup is comparable to U-CMR. Additionally, in the supplementary, we compare to two variants of CMR [70] that have more supervision than our setting. For camera prediction, we compare with CMR [70] (which uses additional keypoint supervision), CSM [85] and U-CMR without texture prediction (U-CMR-noTex).

Figure 3.3: **CMR without keypoints.** CMR-nokp, which directly predicts shape, texture, and camera without keypoint supervision or the proposed camera-multiplex, obtains degenerate solutions. The shape from predicted camera viewpoint (centre) explains the silhouette well but an alternate view (right) reveals that the model has learned to output a planar, flat bird shape.

Qualitative Evaluation

The problem when no keypoints and viewpoints are available is that there always exists a planar shape and texture that explains the image and silhouette for any arbitrary camera pose. We first demonstrate this point using CMR-nokp. We observe that, as expected, CMR-nokp results in a

degenerate solution shown in Figure 3.3 where the recovered shape explains the image silhouette well, but when seen from a different viewpoint the shape is planar.

In Figure 3.6, we visualize U-CMR predictions on unseen images from the CUB test set. Our approach, despite not using any viewpoint or keypoint supervision is able to recover a full, plausible 3D shape of the birds and learns to predict their texture from a single image. Our approach captures various types of bird shapes, including shapes of water birds and songbirds. We are able to recover sharp long tails and some protrusion of legs and beaks. Please see supplementary for more results of random samples from test set and comparisons to CMR.

We further analyze the shape space that we learn in Figure 3.4, where we run principal component analysis on all the shapes obtained on the train set. We find directions that capture changes in the body type, the head shapes, and the tail shapes. In Figure 3.4, we also show that the final mean shape deviates significantly from the template mesh it was initialized to, by becoming thinner and developing a more prominent tail. Please see the supplemental for more results.

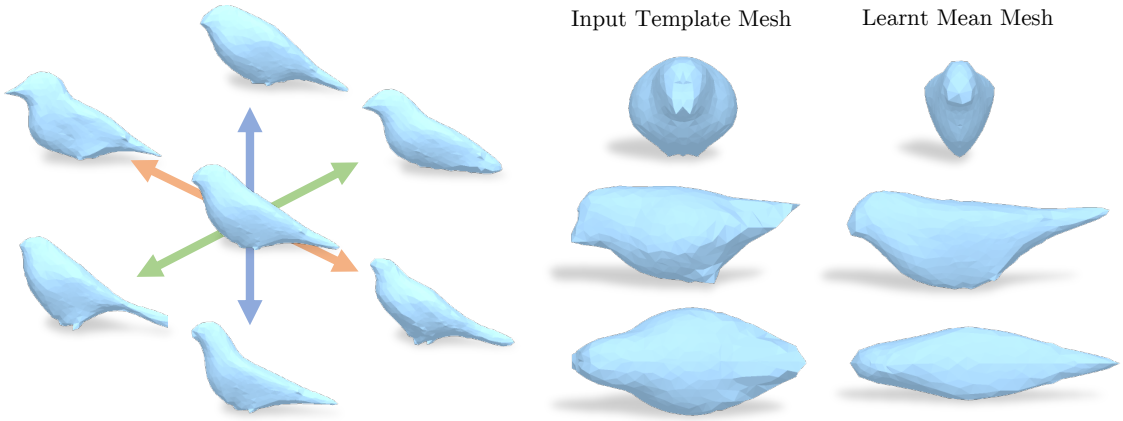

Figure 3.4: **Learned Shape.** On the left, we visualize the space of learned shapes by running PCA. We see that the model has learned to output changes in body type, head shape, and tail types. On the right, we compare the template shape to the final learnt mean mesh. See text for discussion.

Quantitative Evaluation

We conduct quantitative evaluation on the camera poses obtained from our approach, since there are no 3D ground truth shapes on this dataset. For camera evaluation, we compare our approach to CSM [85], which learns to output dense correspondences of the image against a template 3D shape as well as the camera poses from image collections without keypoints. Note that they do not learn to predict 3D shapes. We used the same 3D template mesh as CSM and therefore are comparable to CSM. We evaluate cameras from different approaches on metrics measuring their

	Rotation Error ↓	Entropy (nats) ↑	Wasserstein Dist ↓ Azimuth	Elevation
GT	-	7.44	-	-
CMR [70]	22.94°	7.25	6.03°	4.33°
CMR-nokp	87.52°	5.73	64.66°	12.39°
CSM [85]	61.93°	5.83	27.34°	11.28°
U-CMR (noTex)	61.82°	**7.36**	16.08°	7.90°
U-CMR	45.52°	7.26	8.66°	6.50°

Table 3.2: **Quantitative evaluation of camera pose predictions on the test dataset.** We plot rotation error as the geodesic distance from the ground-truth, the entropy of the azimuth-elevation distribution and the wasserstein distance of marginal Az/El w.r.t. ground-truth. U-CMR outperforms all methods in the absence of keypoints.

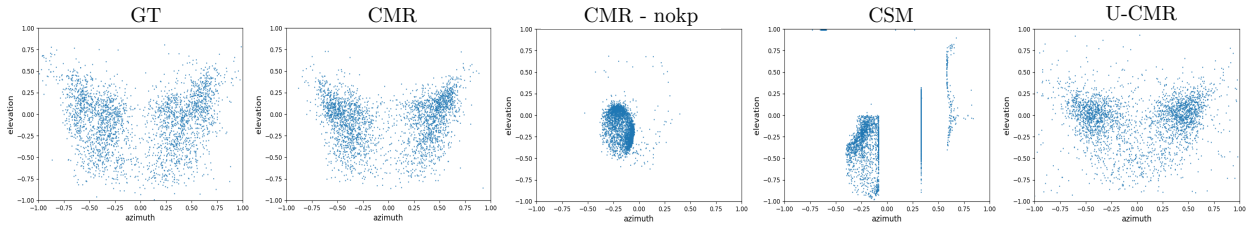

Figure 3.5: **Camera Pose Distributions on CUB Test.** We show azimuth-elevation scatter plots over the entire CUB test set for different approaches. From left to right, we show (i) pseudo ground-truth cameras computed via running SfM on keypoints,(ii) the cameras predicted by CMR which uses the SfM cameras as supervision, (iii) CMR without viewpoint and keypoint supervision (CMR-nokp), (iv) CSM [85] and (v) our approach U-CMR. The last three approaches are weakly-supervised and do not use any keypoint annotation. Notice how the camera pose collapses in CMR-nokp and CSM, while U-CMR with camera-multiplex is able to obtain distribution similar to the ground truth cameras.

accuracy and collapse. We compare our predicted cameras to the pseudo ground-truth cameras computed in CMR [70] using SFM on keypoints. For evaluating accuracy, we compute the rotation error $\text{err}_R = \arccos\left(\frac{\text{Tr}(\tilde{R}^T R^*)-1}{2}\right)$ as the geodesic distance between the predicted camera rotation $\tilde{R}$ and the pseudo ground-truth camera rotation R^*. We report the average rotation error (in degrees) over the entire test dataset. To measure collapse, we analyze the azimuth-rotation distribution and report (i) its entropy (in nats) and (ii) it's Wasserstein distance to the pseudo ground truth azimuth-elevation distribution. Because of computational ease, we only report the Wasserstein distance on the azimuth and elevation marginals. We primarily focus on azimuth and rotation because changes in scale, translation and cyclo-rotation of a camera only warp the image in 2D and don't constitute a "novel viewpoint".

Table 3.2 reports the numbers on all metrics for supervised (CMR) and weakly-supervised

(CMR-nokp, CSM, U-CMR) methods. Observe that all the weakly-supervised baselines incur significant camera pose collapse - as can be seen by the entropy of their distributions. In contrast, U-CMR, despite being weakly-supervised, achieves an entropy that is slightly better than CMR - the supervised baseline. We automatically learn a camera distribution that is almost as close to the ground-truth distribution (in Wasserstein distance) as the supervised baseline (CMR). U-CMR is more accurate than CMR-nokp and CSM and achieves an average rotation error at least 15 degrees better than them. This table also suggests that the texture loss helps with refining camera poses to make them more accurate as U-CMR (noTex) is well-distributed with a very high entropy but is not as accurate as U-CMR.

Figure 3.5 visualizes the azimuth-elevation distributions of different approaches. This figure illustrates that while CSM prevents an extreme mode collapse of the camera, their camera pose distribution still collapses into certain modes. For making this figure, we employ CSM's public model, trained on the same CUB dataset with a fixed template shape. This empirical evidence exemplifies the fact that U-CMR's weakly-supervised camera-multiplex optimization approach learns a camera pose distribution that's much better than other weakly-supervised baselines and almost as good as supervised methods.

Evaluations on other categories

While our primary evaluation is on the CUB Birds dataset, we also run our approach on cars and motorcycles from the Pascal3D+ dataset, and the shoe images we scraped from zappos.com. We show qualitative visualizations of predicted shape, texture and camera for all categories. For Pascal3D cars, we also compare IoU of predicted shapes to CMR, previous deformable model fitting-based [73] and volumetric prediction [160] methods. All three of these approaches leverage segmentation masks and cameras and keypoints to learn 3D shape inference.

Figure 3.7 shows qualitative results on selected images from their respective test set. U-CMR learns accurate camera poses and diverse yet plausible shapes for cars and motorcycles. For shoes, U-CMR shapes are reasonable but not as diverse because of biases in the underlying dataset. We observe some artifacts where the sides of the cars have concave indentations and some parts of the shoes are tapered and pointy. These issues stem from using weak-perspective projections and limitations of the regularization, which is not spatially adaptive. Please see the supplemental for each category's PCA visualizations and the initial template.

Category	CSDM [73]	DRC [160]	CMR [70]	U-CMR
Car	0.60	0.67	0.640	0.646

Table 3.3: **Reconstruction evaluation using PASCAL 3D+.** We report the mean intersection over union (IoU) on PASCAL 3D+ to benchmark the obtained 3D reconstructions (higher is better). We compare to CMR [70], a deformable model fitting approach (CSDM [73]) and a volumetric prediction approach (DRC [160]). CSDM and DRC use image collection supervision in addition to keypoints/cameras.

We report the mean IoU on the test set in Table 3.3 and observe that U-CMR performs comparably to alternate methods that require more supervision.

Limitations

While U-CMR shows promising results in the direction of weakly supervised 3D shape understanding, it has some limitations. Foremost limitation is that we do not model articulation and expect to fail in cases with significant articulation. In Figure 3.8, we demonstrate various modes of failure for shape, texture and camera-viewpoint prediction. Our approach struggles when the bird shape is significantly different from the template mesh and undergoes large articulation, such as the case with flying birds. It is challenging to identify correct camera poses when there's a large deformation like this without keypoints. The data imbalance between flying and not flying birds also exacerbates this problem. The two examples in the top row of the figure show how our shape prediction fails when the bird in the image is flying, or has it's wings open. The example in the bottom right shows an articulated bird with it's head twisted back. Due to the lack of an articulation model, these failure cases are expected. We also fail at predicting good texture sometimes - especially for parts of the object that are not visible. The example on the bottom left of Figure 3.8 and bottom right of Figure 3.7 shows how background colours may leak into the predicted texture.

3.5 Conclusion

In this work, we present a learning framework that can decompose an image of a deformable object into its 3D shape, texture, and camera viewpoint. In order to solve this highly under-constrained problem, we propose a representation for maintaining a distribution over possible camera viewpoints called camera-multiplex. This allows the model to maintain a possible set of camera hypothesis, avoiding the learning process from getting stuck in a bad local minima. We show our approach on four categories, where we show that U-CMR can recover reasonable 3D shape and texture without viewpoints and keypoints.

Acknowledgements. We thank Jasmine Collins for scraping the zappos shoes dataset and members of the BAIR community for helpful discussions. This work was supported in-part by eBay, Stanford MURI and the DARPA MCS program.

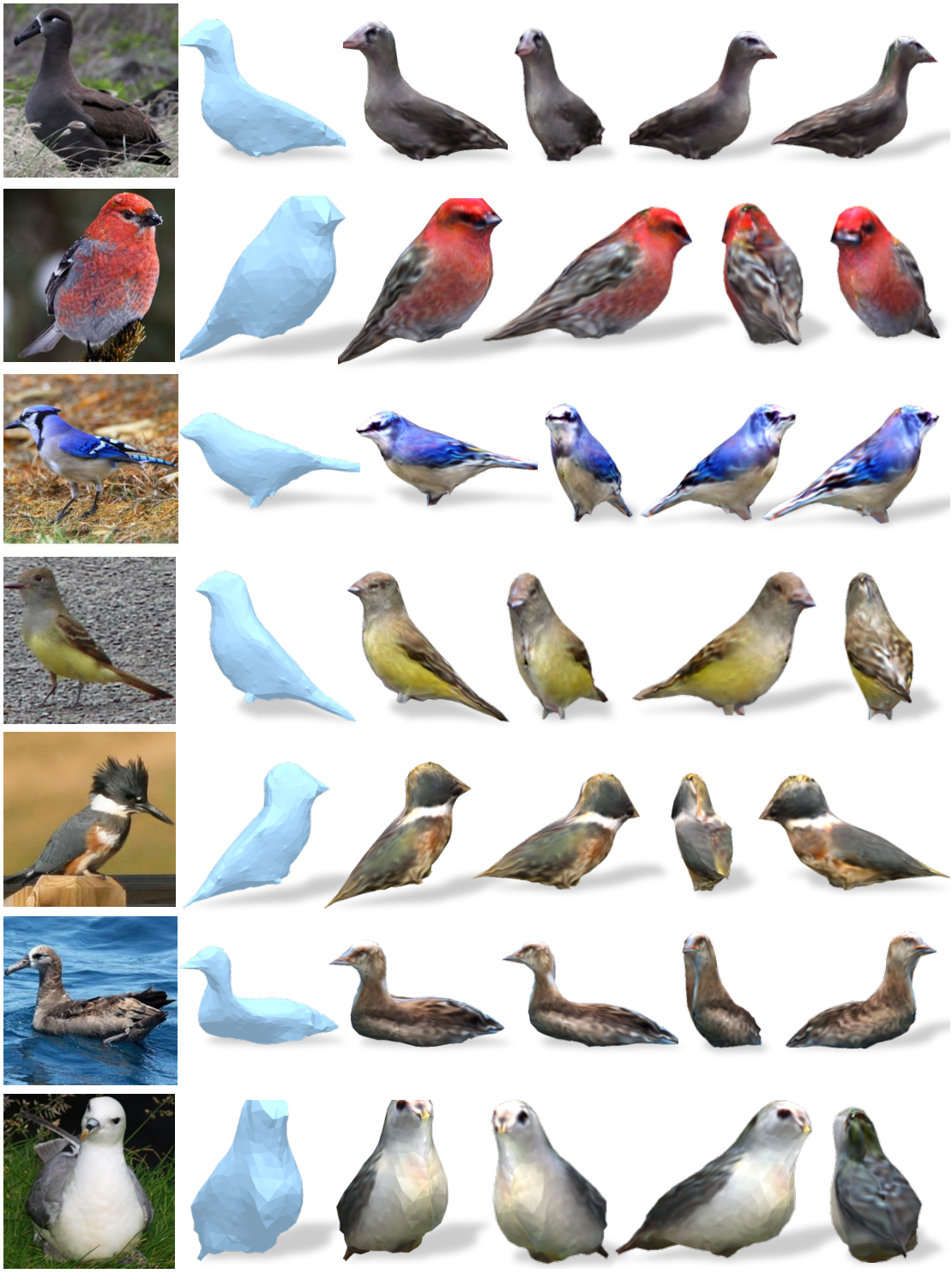

Figure 3.6: **Qualitative results.** For each input test image on the left, we show the predicted mesh, the textured mesh, and the textured mesh from multiple views.

Figure 3.7: **Qualitative results on cars, motorcycles and shoes.** For each image, we show the predicted 3D shape and texture from two viewpoints.

Figure 3.8: **Failure Modes.** The columns, from left to right, show the input image, the predicted shape and texture from the predicted camera, and finally a different view of the textured mesh. See the text for discussion.

Chapter 4

Humans in 4D

4.1 Introduction

In this chapter, we present a fully transformer-based approach for recovering 3D meshes of human bodies from single images, and tracking them over time in video. We obtain unprecedented accuracy in our single-image 3D reconstructions (see Figure 4.1) even for unusual poses where previous approaches struggle. In video, we link these reconstructions over time by 3D tracking, in the process bridging gaps due to occlusion or detection failures. These 4D reconstructions can be seen on the project webpage.

Our problem formulation and approach can be conceived as the "transformerization" of previous work on human mesh recovery, HMR [67] and 3D tracking, PHALP [133]. Since the pioneering ViT paper [31], the process of "transformerization", *i.e.*, converting models from CNNs or LSTMs to transformer backbones, has advanced rapidly across multiple computer vision tasks, *e.g.*, [16, 33, 54, 91, 127, 170]. Specifically for 2D pose (2D body keypoints) this has already been done by ViTPose [178]. We take that as a starting point and through careful design and experimentation, we develop a new version of HMR, which we call HMR 2.0 to acknowledge its antecedent.

We use HMR 2.0 to build a system that can simultaneously reconstruct and track humans from videos. We rely on the recent 3D tracking system, PHALP [133], which we simplify and improve using our pose recovery. This system can reconstruct Humans in 4D, which gives the name to our method, 4DHumans. 4DHumans can be deployed on any video and can jointly track and reconstruct people in video. The functionality of creating a tracking entity for every person is fundamental towards analyzing and understanding humans in video. Besides achieving state-of-the-art results for tracking on the PoseTrack dataset [4], we also apply HMR 2.0 on the downstream application of action recognition. We follow the system design of recent work, [132], and we show that the use of

This chapter is based on joint work with Georgios Pavlakos, Jathushan Rajasegaran, Angjoo Kanazawa, and Jitendra Malik [48], more details for which can be found on the project webpage.

Figure 4.1: **A "transformerized" view of Human Mesh Recovery**. We describe HMR 2.0, a fully transformer-based approach for 3D human pose and shape reconstruction from a single image. Besides impressive performance across a wide variety of poses and viewpoints, HMR 2.0 also acts as the backbone of an improved system for jointly reconstructing and tracking Humans in 4D (4DHumans). Here, we see output reconstructions from HMR 2.0 for each 2D detection in the left image. We encourage the reader to see our 4D reconstructions on the project webpage to appreciate the tracking and temporal stability.

HMR 2.0 can achieve impressive improvements upon the state of the art on action recognition on AVA v2.2 dataset.

This work is unabashedly a systems work. We explored various choices and put together the best combination. Our model will be made publicly available. There is an emerging trend, in computer vision as in natural language processing, of large pretrained models (sometimes also called "foundation models") which find widespread downstream applications and thus justify the scaling effort. HMR 2.0 is such a large pre-trained model which could potentially be useful not just in computer vision, but also in robotics [122, 128, 163], computer graphics [168], bio-mechanics, and other fields where analysis of the human figure and its movement from images or videos is needed.

Our contributions can be summarized as follows:

1. We propose an end-to-end "transformerized" architecture for human mesh recovery, HMR 2.0. Without relying on domain-specific designs, we outperform existing approaches for 3D body pose reconstruction.

2. Building on HMR 2.0, we design 4DHumans that can jointly reconstruct and track humans in video, achieving state-of-the-art results for tracking.

3. We show that better 3D poses from HMR 2.0 result in better performance on the downstream task of action recognition, finally contributing to the state-of-the-art result (42.3 mAP) on the AVA benchmark.

4.2 Related Work

Human Mesh Recovery from a Single Image. Although, there have been many approaches that estimate 3D human pose and shape relying on iterative optimization, *e.g.*, SMPLify [14] and variants [50, 87, 125, 136, 155, 188], for this analysis we will focus on approaches that directly regress the body shape from a single image input. In this case, the canonical example is HMR [67], which uses a CNN to regress SMPL [100] parameters. Since its introduction, many improvements have been proposed for the original method. Notably, many works have proposed alternative methods for pseudo-ground truth generation, including using temporal information [5], multiple views [89], or iterative optimization [83, 65, 124]. SPIN [83] proposed an in-the-loop optimization that incorporated SMPLify [14] in the HMR training. Here, we also rely on pseudo-ground truth fits for training, and we use [84] for the offline fitting.

More recently, there have been works that propose more specialized designs for the HMR architecture. PyMAF [192, 191] incorporates a mesh alignment module for the regression of the SMPL parameters. PARE [81] proposes a body-part-guided attention mechanism for better occlusion handling. HKMR [43] performs a prediction that is informed by the known hierarchical structure of SMPL. HoloPose [51] proposes a pooling strategy that follows the 2D locations of each body joints. Instead, we follow a design without any domain-specific decisions and we show that it outperforms all previous approaches.

Many related approaches are making non-parametric predictions, *i.e.*, instead of estimating the parameters of the SMPL model, they explicitly regress the vertices of the mesh. GraphCMR [82] uses a graph neural network for the prediction, METRO [94] and FastMETRO [20] use a transformer, while Mesh Graphormer [95] adopts a hybrid between the two. Since we regress the SMPL model parameters, instead of the locations of mesh vertices, we are not directly comparable to these. However, we show how we can use a fully "transformerized" design for HMR.

Human Mesh & Motion Recovery from Video. To extend Human Mesh Recovery over time, most methods use the basic backbone of HMR [67] and propose designs for the temporal encoder that fuses the per-frame features. HMMR [69] uses a convolutional encoder on features extracted from HMR [67]. VIBE [80], MEVA [105] and TCMR [21] use a recurrent temporal encoder. DSD [150] combines convolutional and self-attention layers, while MAED [167] and t-HMMR [124] employ a transformer-based temporal encoder. Baradel *et al*. [7, 8] also used a transformer for temporal pose prediction, while operating directly on SMPL poses. One key limitation of these approaches is that they often operate in scenarios where tracking is simple [69, 193], *e.g.*, videos with a single person or minimal occlusions. In contrast to that, our complete 4DHumans approach is also solving the tracking problem.

Tracking People in Video. Recently, there have been approaches that demonstrate state-of-the-art performance for tracking by relying on 3D human reconstruction from HMR models, *i.e.*, T3DP [134] and PHALP [133]. In these methods, every person detection is lifted to 3D using an HMR network [124] and then tracking is performed using the 3D representations from lifting [134] and prediction [133] to track people in video. Empirical results show that PHALP works very well on multiple tracking benchmarks (the main requirement is that the images have enough spatial resolution to permit lifting of the people to 3D). We use these tracking pipelines, and particularly

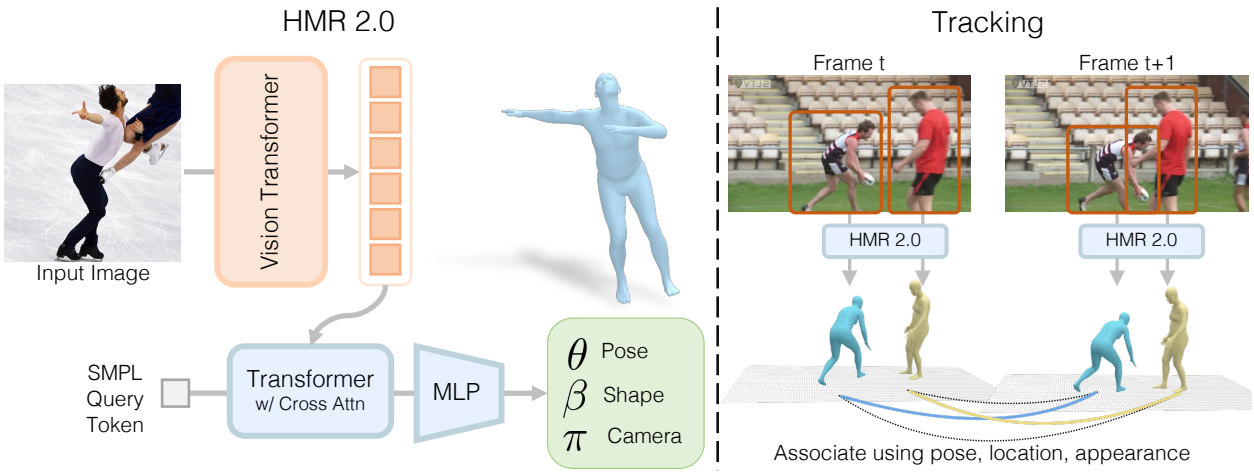

Figure 4.2: **Overview of our approach.** Left: HMR 2.0 is a fully "transformerized" version of a network for Human Mesh Recovery. Right: We use HMR 2.0 as the backbone of our 4DHumans system, that builds on PHALP [133], to jointly reconstruct and track humans in 4D.

PHALP, as a task to evaluate methods for human mesh recovery.

Action Recognition. Action recognition is typically performed using appearance features from raw video input. Canonical examples in this category include SlowFast [37] and MViT [33]. Simultaneously, there are approaches that use features extracted from body pose information, *e.g.*, PoTion [22] and JMRN [144]. A recent approach [132] demonstrates state-of-the-art performance for action recognition by fusing video-based features with features from 3D human pose estimates. We use the pipeline of this approach and employ action recognition as a downstream task to evaluate human mesh recovery methods.

4.3 Reconstructing People

Preliminaries

Body Model. The SMPL model [100] is a low-dimensional parametric model of the human body. Given input parameters for pose ($\theta \in \mathbb{R}^{24\times3\times3}$) and shape ($\beta \in \mathbb{R}^{10}$), it outputs a mesh $M \in \mathbb{R}^{3\times N}$ with $N = 6890$ vertices. The body joints $X \in \mathbb{R}^{3\times k}$ are defined as a linear combination of the vertices and can be computed as $X = MW$ with fixed weights $W \in \mathbb{R}^{N\times k}$. Note that pose parameters θ include the body pose parameters $\theta_b \in \mathbb{R}^{23\times3\times3}$ and the global orientation $\theta_g \in \mathbb{R}^{3\times3}$.

Camera. We use a perspective camera model with fixed focal length and intrinsics K. Each camera $\pi = (R, t)$ consists of a global orientation $R \in \mathbb{R}^{3\times3}$ and translation $t \in \mathbb{R}^3$. Given these parameters, points in the SMPL space (*e.g.*, joints X) can be projected to the image as $x = \pi(X) = \Pi(K(RX + t))$, where Π is a perspective projection with camera intrinsics K. Since θ already includes a global orientation, in practice we assume R as identity and only predict camera translation t.

HMR. The goal of the human mesh reconstruction (HMR) task is to learn a predictor $f(I)$ that given a single image I, reconstructs the person in the image by predicting their 3D pose and shape parameters. Following the typical parametric approaches [67, 83], we model f to predict $\Theta = [\theta, \beta, \pi] = f(I)$ where θ and β are the SMPL pose and shape parameters and π is the camera translation.

Architecture

We re-imagine HMR [67] as an end-to-end transformer architecture that uses no domain specific design choices. Yet, it outperforms all existing approaches that have heavily customized architectures and elaborate design decisions. As shown in Figure 4.2, we use (i) a ViT [31] to extract image tokens, and (ii) a standard transformer decoder that cross-attends to image tokens to output Θ.
ViT. The Vision Transformer, or ViT [31] is a transformer [164] that has been modified to operate on an image. The input image is first patchified into input tokens and passed through the transformer to get output tokens. The output tokens are then passed to the transformer decoder. We use a ViT-H/16, the "Huge" variant with 16×16 input patch size. Please see SupMat for more detail.
Transformer decoder. We use a standard transformer decoder [164] with multi-head self-attention. It processes a single (zero) input token by cross-attending to the output image tokens and ends with a linear readout of Θ. We follow [83] and regress 3D rotations using the representation of [199].

Losses

Following best practices in the HMR literature [67, 83], we train our predictor f with a combination of 2D losses, 3D losses, and a discriminator. Since we train with a mixture of datasets, each having different kinds of annotations, we employ a subset of these losses for each image in a mini-batch. We use the same losses even with pseudo-ground truth annotations. Given an input image I, the model predicts $\Theta = [\theta, \beta, \pi] = f(I)$. Whenever we have access to the ground-truth SMPL pose parameters θ^* and shape parameters β^*, we bootstrap the model predictions using an MSE loss:

$$\mathcal{L}_{\text{smpl}} = ||\theta - \theta^*||_2^2 + ||\beta - \beta^*||_2^2.$$

When the image has accurate ground-truth 3D keypoint annotations X^*, we additionally supervise the predicted 3D keypoints X with an L1 loss:

$$\mathcal{L}_{\text{kp3D}} = ||X - X^*||_1.$$

When the image has 2D keypoints annotations x^*, we supervise projections of predicted 3D keypoints $\pi(X)$ using an L1 loss:

$$\mathcal{L}_{\text{kp2D}} = ||\pi(X) - x^*||_1.$$

Furthermore, we want to ensure that our model predicts valid 3D poses and use the adversarial prior in HMR [67]. It factorizes the model parameters into: (i) body pose parameters θ_b, (ii) shape

parameters β, and (iii) per-part relative rotations θ_i, which is one 3D rotation for each of the 23 joints of the SMPL model. We train a discriminator D_k for each factor of the body model, and the generator loss can be expressed as:

$$\mathcal{L}_{\text{adv}} = \sum_k (D_k(\theta_b, \beta) - 1)^2.$$

Pseudo-Ground Truth fitting

We scale to unlabelled datasets (*i.e.*, InstaVariety [69], AVA [49], AI Challenger [171]) by computing pseudo-ground truth annotations. Given any image, we first use an off-the-shelf detector [91] and a body keypoints estimator [178] to get bounding boxes and corresponding 2D keypoints. We then fit a SMPL mesh to these 2D keypoints using ProHMR [84] to get pseudo-ground truth SMPL parameters θ^* and β^* with camera π^*.

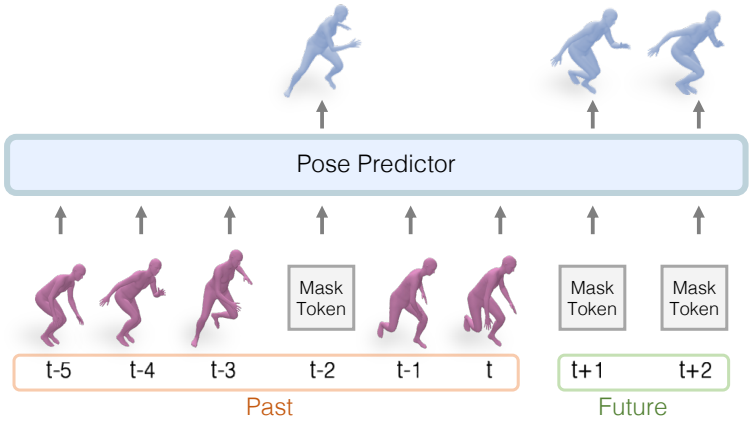

Figure 4.3: **Pose prediction:** We train a BERT-style [29] transformer model on over 1 million tracks obtained from [132]. This allow us to make future predictions and amodal completion of missing detections using the same model. To predict future poses ($t+1$, $t+2$, ...), we query the model with a mask-token using corresponding positional embeddings. Similarly for amodal completion, we replace missing detections with a masked token.

4.4 Tracking People

In videos with multiple people, we need the ability to associate people across time, *i.e.*, perform tracking. For this we build upon PHALP [133], a state-of-the-art tracker based on features derived from HMR-style 3D reconstructions. The basic idea is to detect people in individual frames, and "lift" them to 3D, extracting their 3D pose, location in 3D space (derived from the estimated camera), and 3D appearance (derived from the texture map). A tracklet representation is incrementally built up for each individual person over time. The recursion step is to predict for each tracklet,

the pose, location and appearance of the person in the next frame, all in 3D, and then find best matches between these top-down predictions and the bottom-up detections of people in that frame after lifting them to 3D. The state represented by each tracklet is then updated by the incoming observation, and the process is iterated. It is possible to track through occlusions because the 3D representation of a tracklet continues to be updated based on past history.

We believe that a robust pose predictor should also perform well, when evaluated on this downstream task of tracking, so we use the tracking metrics as a proxy to evaluate the quality of 3D reconstructions. But first we needed to modify the PHALP framework to allow for fair comparison of different pose prediction models. Originally, PHALP used pose features based on the last layer of the HMR network, *i.e.*, a 2048-dimensional embedding space. This limits the ability of PHALP to be used with different pose models (*e.g.*, HMR 2.0, PARE, PyMAF etc.). To create a more generic version of PHALP, we perform the modification of representing pose in terms of SMPL pose parameters, and we accordingly optimize the PHALP cost function to utilize the new pose distance. Similarly, we adapt the pose predictor to operate on the space of SMPL parameters. More specifically, we train a vanilla transformer model [164] by masking random pose tokens as shown in the Fig 4.3. This allows us to predict future poses in time, as well as amodal completion of missing detections. With these modifications, we can plug in any mesh recovery methods and run them on any videos. We call this modified version PHALP′.

4DHumans. To track people in videos, previous approaches relied on off-the-shelf tracking approaches and used their output to reconstruct humans in videos (*e.g.*, take the bounding boxes from tracking output and reconstruct people). For example, PHD [193], HMMR [69] can run on videos with only single person in the scene. In this work, we combine reconstruction and tracking into a single system and show that better pose reconstructions result in better tracking and this combined system can now run on any videos in the wild.

4.5 Experiments

In this section, we evaluate our reconstruction and tracking system qualitatively and quantitatively. First, we show that HMR 2.0 outperforms previous methods on standard 2D and 3D pose accuracy metrics (Section 4.5). Second, we show 4DHumans is a versatile tracker, achieving state-of-the-art performance (Section 4.5). Finally, we further demonstrate the robustness and accuracy of our recovered poses via superior performance on the downstream application of action recognition (Section 4.5).

Setup

Datasets. Following previous work, we use the typical datasets for training, *i.e.*, Human3.6M [59], MPI-INF-3DHP [108], COCO [96] and MPII [3]. Additionally, we use InstaVariety [69], AVA [49] and AI Challenger [171] as extra data where we generate pseudo-ground truth fits.

	Method	3DPW		Human3.6M	
		MPJPE ↓	PA-MPJPE ↓	MPJPE ↓	PA-MPJPE ↓
Temporal	Kanazawa *et al.* [69]	116.5	72.6	-	56.9
	Doersch *et al.* [30]	-	74.7	-	-
	Arnab *et al.* [5]	-	72.2	77.8	54.3
	DSD [150]	-	69.5	59.1	42.4
	VIBE [80]	93.5	56.5	65.9	41.5
Frame-based	Pavlakos *et al.* [126]	-	-	-	75.9
	HMR [67]	130.0	76.7	88.0	56.8
	NBF [118]	-		-	59.9
	GraphCMR [82]	-	70.2	-	50.1
	HoloPose [51]	-	-	60.3	46.5
	DenseRaC [177]	-	-	76.8	48.0
	SPIN [83]	96.9	59.2	62.5	41.1
	DecoMR [189]	-	61.7†	-	39.3†
	DaNet [190]	-	56.9	61.5	48.6
	Song *et al.* [147]	-	55.9	-	56.4
	I2L-MeshNet [112]	100.0	60.0	55.7†	41.1†
	HKMR [43]	-	-	59.6	43.2
	PyMAF [192]	92.8	58.9	57.7	40.5
	PARE [81]	82.0	50.9	76.8	50.6
	PyMAF-X [191]	78.0	47.1	54.2	37.2
	HMR 2.0a	69.8	44.4	45.3	33.8
	HMR 2.0b	81.4	54.5	52.6	33.4

Table 4.1: **Reconstructions evaluated in 3D:** Reconstruction errors (in mm) on the 3DPW and Human3.6M datasets. † denotes the numbers evaluated on non-parametric results. Lower ↓ is better. Please see the text for details.

Baselines. We report performance on benchmarks that we can compare with many previous works (Section 4.5), but we also perform a more detailed comparison with recent state-of-the-art methods, *i.e.*, PyMAF [192], CLIFF [93], HMAR [133] PARE [81], and PyMAF-X [191]. For fairness, we only evaluate the body-only performance of PyMAF-X.

Pose Accuracy

3D Metrics. For 3D pose accuracy, we follow the typical protocols of prior work, *e.g.*, [83], and we present results on the 3DPW test split and on the Human3.6M val split, reporting MPJPE, and PA-MPJPE in Table 4.1. Please notice that we only compare with methods that do not use the training set of 3DPW for training, similar to us. We observe that with our HMR 2.0a model, which trains only on the typical datasets, we can outperform all previous baselines across all metrics. However, we believe that these benchmarks are very saturated and these smaller differences in pose metrics tend to not be very significant. In fact, we observe that by a small compromise of the performance on 3DPW, our HMR 2.0b model, which trains for longer on more data (AVA [49], AI Challenger [171],

Method	LSP-Extended		COCO		PoseTrack	
	@0.05	@0.1	@0.05	@0.1	@0.05	@0.1
PyMAF [192]	-	-	0.68	0.86	0.77	0.92
CLIFF [93]	0.30	0.64	0.63	0.88	0.75	0.93
PARE [81]	0.27	0.60	0.72	0.91	0.79	0.93
PyMAF-X [191]	-	-	0.79	0.93	0.85	0.95
HMR 2.0a	0.38	0.72	0.79	0.95	0.86	0.97
HMR 2.0b	0.54	0.84	0.85	0.96	0.90	0.98

Table 4.2: **Reconstructions evaluated in 2D.** PCK scores of projected keypoints at different thresholds on the LSP-Extended, COCO, and PoseTrack datasets. Higher ↑ is better.

and InstaVariety [69]), achieves results that perform better on more unusual poses than what can be found in Human3.6M and 3DPW. We observe this qualitatively and from performance evaluated on 2D pose reprojection (Table 4.2). Furthermore, we observe that HMR 2.0b is a more robust model and use it for evaluation in the rest of the chapter.

2D Metrics. We evaluate 2D image alignment of the generated poses by reporting PCK of reprojected keypoints at different thresholds on LSP-Extended [64], COCO validation set [96], and Posetrack validation set [4]. Since PyMAF(-X) [192, 191] were trained using LSP-Extended, we do not report numbers for that part of the table. Notice in Table 4.2, that HMR 2.0-b consistently outperforms all previous approaches. On LSP-Extended, which contains unusual poses, HMR 2.0-b achieves PCK@0.05 of 0.54, which is $2\times$ better than the second best (PARE) with 0.27. For PCK@0.05 on easier datasets like COCO and PoseTrack with less extreme poses, HMR 2.0b still outperforms the second-best approaches but by narrower margins of 8% and 6% respectively. HMR 2.0a also outperforms all baselines, but is worse than HMR 2.0b, especially on harder poses in LSP-Extended.

Qualitative Results. We show qualitative results of HMR 2.0 in Figure 4.4. We are robust to extreme poses and partial occlusions. Our reconstructions are well-aligned with the image and are valid when seen from a novel view. Moreover, we compare with our closest competitors in Figure 4.5. We observe that PyMAF-X and particularly PARE often struggle with more unusual poses, while HMR 2.0 returns more faithful reconstructions.

Tracking

For tracking, we first demonstrate the versatility of the modifications introduced by PHALP′, which allow us to evaluate 3D pose estimators on the downstream task of tracking. Then, we evaluate our complete system, 4DHumans, with respect to the state of the art.

Evaluation Setting. Following previous work [134, 133], we report results based on IDs (ID switches), MOTA [74], IDF1 [138], and HOTA [104] on the Posetrack validation set using the protocol of [133], with detections from Mask R-CNN [53].

Tracker	Pose Engine	Posetrack			
		IDs↓	MOTA↑	IDF1↑	HOTA↑
PHALP′	PyMAF [192]	598	58.7	76.2	53.0
	CLIFF [93]	550	58.9	76.6	53.5
	HMAR [133]	523	59.2	76.8	53.5
	PARE [81]	477	59.1	77.1	53.8
	PyMAF-X [191]	472	59.2	76.9	53.7
	HMR 2.0	471	59.1	76.7	53.8

Table 4.3: **Tracking with different 3D pose estimators.** With the modifications of PHALP′, we have a versatile tracker that allows different 3D pose estimators to be plugged into it. HMR 2.0, PARE, and PyMAF-X perform the best in this setting.

Versatility of PHALP′. With the modifications of PHALP′, we abandon the model-specific latent space of [133] and instead, we operate in the SMPL space, which is shared across most mesh recovery systems. This makes PHALP′ more versatile and allows us to plug in different 3D pose estimators and compare them based on their performance on the downstream task of tracking. We perform this comparison in Table 4.3 where we use pose and location cues from state of the art 3D pose estimators (while still using appearance from HMAR [133]). We observe that HMR 2.0, PARE [81] and PyMAF-X [191] perform the best on the Posetrack dataset, with minor differences between them. Note that tracking is often most susceptible to errors in predicted 3D locations with body pose having a smaller effect in performance [133]. This means that good tracking performance can indicate robustness to occlusions, so it is helpful to consider this metric, but it is less helpful to distinguish fine-grained differences in pose. As a result, the competitive results of PARE [81] and PyMAF-X [191] indicate that they handle occlusions gracefully, but their pose estimation might still be less accurate (as observed from Table 4.2). See also Figure 4.5 and SupMat for more qualitative comparisons.

4DHumans. Table 4.4 evaluates tracking performance of our complete system, 4DHumans, on the PoseTrack dataset. Using the same bounding box detector as [134, 133], 4DHumans outperforms existing approaches on all metrics, improving ID Switches by 16%. Using the improved ViTDet detector [91] can improve performance further. As a by-product of our temporal prediction model (Figure 4.3), we can perform amodal completion and attribute a pose to missing detections.

Action Recognition

Evaluation setting. The approach of [132] is the state of the art for action recognition in videos. Given a video input, the authors propose using per-frame 3D pose and location estimates (using off-the-shelf HMR models [133]) as an additional feature for predicting action labels. They also show results for a "pose-only" baseline that predicts action labels using only 3D pose and location estimates. We use this setting to compare our model with baselines on the downstream task of action recognition on the AVA dataset [49]. In [132], the authors train a transformer that takes

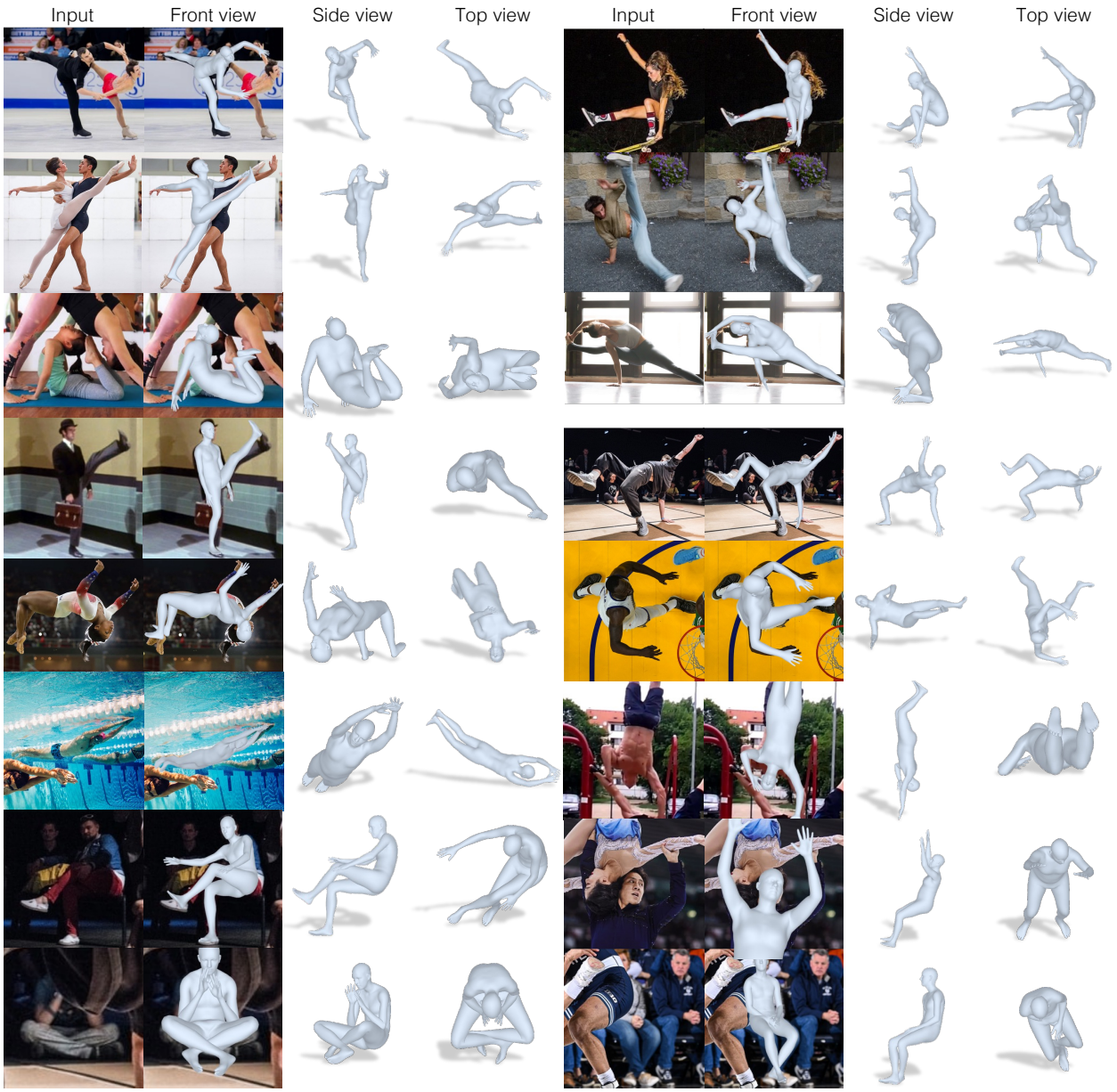

Figure 4.4: **Qualitative evaluation of HMR 2.0.** For each example we show: a) the input image, b) the reconstruction overlay, c) a side view, d) the top view. To demonstrate the robustness of HMR 2.0, we visualize results for a variety of settings - for unusual poses (rows 1-4), for unusual viewpoints (row 5) and for images with poor visibility, extreme truncations and extreme occlusions (rows 6-8).

Figure 4.5: **Qualitative comparison of state-of-the-art mesh recovery methods.** HMR 2.0 returns more faithful reconstructions for unusual poses compared to the closest competitors, PyMAF-X [191] and PARE [81].

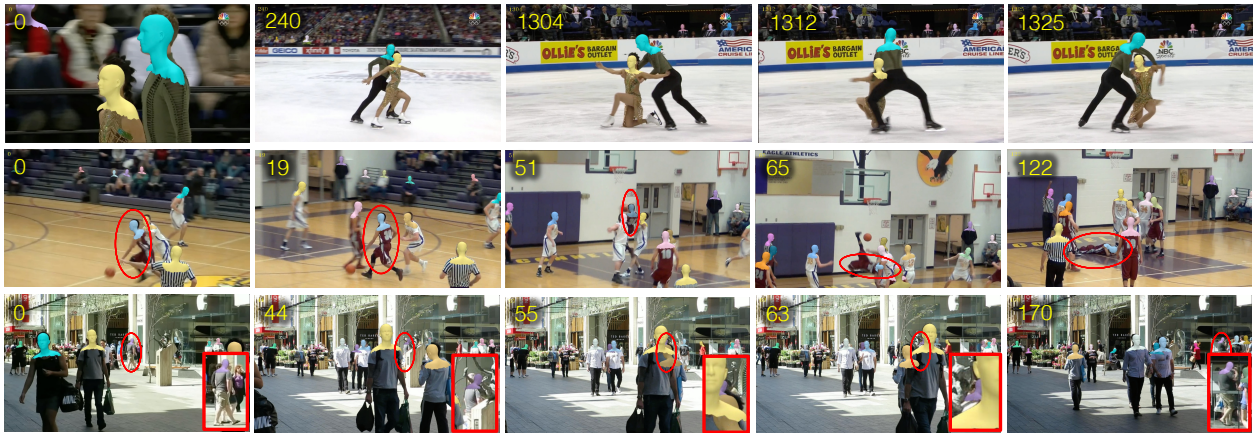

Figure 4.6: **Qualitative tracking results of 4DHumans**. We use head masks (frame number is on the top left). First row: We track people skating on ice with challenging poses and heavy occlusions, in a minute long video without switching identities. Second row: The main person is tracked through multiple interactions with other players. Third row: The person of interest is tracked through long occlusions.

Method	Posetrack			
	IDs↓	MOTA↑	IDF1↑	HOTA↑
Trackformer [109]	1263	33.7	64.0	46.7
Tracktor [9]	702	42.4	65.2	38.5
AlphaPose [35]	2220	36.9	66.9	37.6
Pose Flow [175]	1047	15.4	64.2	38.0
T3DP [134]	655	55.8	73.4	50.6
PHALP [133]	541	58.9	76.4	52.9
4DHumans	455	59.5	77.6	54.1
4DHumans + ViTDet	367	61.9	79.3	57.8

Table 4.4: **Comparison of 4DHumans with the state of the art on the Posetrack dataset.** 4DHumans achieve state-of-the-art tracking performance for all metrics. Incorporating a better detection system [91] leads to further performance improvements.

Action Model	Pose Engine	OM	PI	PM	mAP
[132]	PyMAF [192]	7.3	16.9	34.7	15.4
	CLIFF [93]	9.2	20.0	40.3	18.6
	HMAR [133]	8.7	20.1	40.3	18.3
	PARE [81]	9.2	20.7	41.5	19.1
	PyMAF-X [191]	10.2	21.4	40.8	19.6
	HMR 2.0	11.9	24.6	45.8	22.3

Table 4.5: **Action recognition results on the AVA dataset.** We benchmark different mesh recovery methods on the downstream task of pose-based action recognition. Here, *OM* : Object Manipulation, *PI* : Person Interactions, and *PM* : Person Movement. Higher ↑ is better.

SMPL poses as input and predicts action labels. Following their setup, we train a separate action classification transformer for each baseline.

Comparisons. Comparing results in Table 4.5, we observe that HMR 2.0 outperforms baselines on the different class categories (OM, PI, PM) and overall. It achieves an mAP of 22.3 on the AVA test set, which is 14% better than the second-best baseline. Since accurate action recognition from poses needs fine-grained pose estimation, this is strong evidence that HMR 2.0 predicts more accurate poses than existing approaches. In fact, when combined with appearance features, [132] shows that HMR 2.0 achieves the state of the art of 42.3 mAP on AVA action recognition, which is 7% better than the second-best of 39.5 mAP.

4.6 Conclusion

We study the problem of reconstructing and tracking humans from images and video. First, we propose HMR 2.0, a fully "transformerized" version of a network for the problem of Human Mesh Recovery [67]. HMR 2.0 achieves strong performance on the usual 2D/3D pose metrics, while also acting as the backbone for our improved video tracker. The full system 4DHumans, jointly reconstructs and tracks people in video and achieves state-of-the-art results for tracking. To further illustrate the benefit of our 3D pose estimator, HMR 2.0, we apply it to the task of action recognition, where we demonstrate strong improvements upon previous pose-based baselines.

Our work pushes the boundary of the videos that can be analyzed with techniques for 3D human reconstruction. At the same time, the improved results also demonstrate the type of limitations that need to be addressed in the future. For example, the use of the SMPL model [100] creates certain limitations, and leveraging improved models would allow us to model hand pose and facial expressions [125], or even capture greater age variation, *e.g.*, infants [55] and kids [123, 151]. Moreover, since we consider each person independently, our reconstruction are less successful at capturing the fine-grained nature of people in close proximity, *e.g.*, contact [38]. Besides this, our reconstructions "live" in the camera frame, so for proper understanding of the action in a video, we need to consider everyone in a common world coordinate frame, by reasoning about the camera motion too [184, 187]. Finally, lower input resolution can affect the quality of our reconstructions, which could be addressed by more extreme resolution augmentations [176].

Chapter 5

Conclusion

In conclusion, this thesis takes significant steps towards high-fidelity 3D reconstruction of humans, animals, and objects, from one or many images.

In Chapter 2, we saw how to reconstruct shape when multiple images are available. Our approach, Differentiable Stereopsis (DS), reconstructs objects and scenes from very few images, with noisy camera poses as input. We saw that compared to its volumetric counterparts, DS especially shines in the regime with few input images and high camera pose noise, where the input data is limited and noisy. Despite the shape and camera optimization landscape being highly non-convex in such a low-data setting, our SGD-based optimization avoids local minima by only optimizing for shape and camera parameters, but to minimize a texture reprojection loss.

In Chapter 3, we presented UCMR (Unsupervised-CMR), a learning framework that learns to recover the 3D shape, pose and texture from a single image, by leveraging an image collection of a particular category, without any ground truth 3D shape, multi-view, camera viewpoints or keypoint supervision. As the key contribution, we maintained a camera pose distribution for each image, as a set of camera hypotheses, and optimize them during training to best explain the image given the current shape and texture. We observed that optimizing the cameras in this way avoids a network collapse to degenerate solutions when no ground-truth annotations are available. Eventually, these optimized cameras can be distilled into a separate network to predict cameras from images. We showed that UCMR faithfully predicts shape, camera and texture, while scaling to multiple categories without ground-truth 3D or keypoint annotations.

In Chapter 4, we presented an approach to reconstruct humans and track them over time. At the core of our approach, HMR 2.0, is a fully “transformerized” single-view 3D human pose and shape reconstruction model. We also use these 3D reconstructions as input to build a 3D tracking system, 4DHumans, that enables us to analyze humans in video – dealing with multiple people and maintaining identities through occlusion events. Powered by 4DHumans, we enable using 3D humans pose estimation for downstream tasks, achieving state-of-the-art results for tracking people from monocular video, and contributing to the best performing pose-based human action recognition system.

A common theme across Chapters 3 and 4, where we learn priors for single-view reconstruction, is that of "optimize and distill", *i.e.* first optimize to extract some 3D information for every instance,

and then distill it down into a network. For U-CMR, we optimized cameras while training shape, and distilled the cameras later into a feed-forward network. With HMR 2.0, we first pre-processed an unlabelled dataset, optimizing the SMPL mesh to fit 2D keypoints. We then distill these optimized SMPL parameters into a neural network, *i.e*. train it using the optimized parameters as pseudo-GT.

Something we have not discussed until now, is that the world around us can be modeled using a variety of 3D representations. Classical approaches such as point clouds, voxels, and meshes have been well-studied and are supported by numerous communities outside computer vision. More recently, implicit neural representations, like NeRF [110], have gained popularity for novel-view synthesis. Each of these representations comes with its own set of advantages and disadvantages. Volumetric representations like NeRFs demonstrate promise for 3D reconstruction, particularly in multi-view settings when numerous images are available. However, in sparse view settings with noisy data, a mesh-based representation outperforms volumetric approaches, as illustrated in Chapter 2, due to the inherent surface regularization provided by meshes. In Chapter 3, learning morphable models for shape was most straightforward with meshes, although this constrained us to a fixed topology. However, subsequent work in the community [32] has addressed this limitation. For human representations, in Chapter 4, we used the most popular parametric human model [100], which happens to be mesh-based. While there has been progress in developing more expressive human models like [119], they are yet to be widely adopted within the research community. Regardless, the ideas presented in this thesis are general and can be extended to various types of shape representations.

Even though this thesis takes major steps towards solving 3D reconstruction of objects and scenes in the wild, the problem remains unsolved and is an active area of research. Chapters 3 and 4 show how to learn priors for single-view 3D reconstruction for many categories. However, these learnt priors are category-specific. Even though this can be scaled to a large number of categories, "categories" are an unnatural man-made discretization of an otherwise continuous world. What we truly need, and already have as humans, are 3D priors that work across all objects and scenes. When built, these priors will definitely use semantics, *i.e*. some implicit knowledge of similar instances having similar shapes, but will still work for all data in the wild. For example, the shift we've witnessed in generative image modelling, from class-conditioning [15] to text-conditioning [140], is of a similar kind.

The 3D community has taken a stab at learning category-agnostic priors multiple times [195, 170, 114] but much progress remains to be made. Two things make this problem much harder than its 2D counterparts: the lack of large-scale annotated 3D data, and the high-dimensionality of the 3D data over which these priors need to be learnt. The latter could be resolved as we build more efficient models, *e.g*. RIN [61] decouples computation on the data from the underlying spatial topology in which the data resides. The lack of data, however, is a more fundamental issue. Collecting 3D annotoations of all the complex objects in the real world, with a laser scanner for example, is time-consuming, expensive, and unscalable. Even though size of 3D datasets has increased by orders of magnitude, the biggest 3D datasets today, like Objaverse [27] at 800 thousand models, are still orders of magnitudes smaller than large-scale image datasets like LAION-5B [143], which have 5 billion image-text pairs. Furthermore, these 3D datasets, though helpful, are very different from the actual 3D world, as being CAD models that are synthetically created by artists.

The most natural scalable source of data, for learning 3D priors, is video, of which we have plenty. For example, YouTube-8M [1] has 350 thousand hours of video and the total amount of data on YouTube is closer to 150 million hours. However, it remains unclear how to use this data to build priors on the 3D world. With NeRFs and their dynamic variants, we are making steady progress on 3D-ifying each video individually. Similar to the "optimize and distill" ideas used in Chapters 3 and 4, one solution might be to first reconstruct a large dataset of scenes to 3D via per-video optimization, and then distill the 3D dataset down to a prior. Another exciting direction is to leverage 2D image priors, as encoded in pretrained 2D generative models, to extract a 3D prior. Some recent approaches explore this direction [97], sometimes totally bypassing the need for any 3D data [131]. However, these approaches are far from a general prior on the 3D world.

In this thesis, we have made major steps at learning to reconstruct 3D shapes of humans and objects, from single and multiple views. However, 3D computer vision remains an important and open area of research, where much progress still remains to be made, and given the pace of research in the community, we can anticipate the research community making exciting new progress in this direction over the years to come.

Bibliography

[1] Sami Abu-El-Haija et al. “Youtube-8m: A large-scale video classification benchmark”. In: *arXiv preprint arXiv:1609.08675* (2016).

[2] Kalyan Vasudev Alwala, Abhinav Gupta, and Shubham Tulsiani. “Pre-train, self-train, distill: A simple recipe for supersizing 3d reconstruction”. In: *CVPR*. 2022, pp. 3773–3782.

[3] Mykhaylo Andriluka et al. “2D human pose estimation: New benchmark and state of the art analysis”. In: *CVPR*. 2014.

[4] Mykhaylo Andriluka et al. “PoseTrack: A benchmark for human pose estimation and tracking”. In: *CVPR*. 2018.

[5] Anurag Arnab, Carl Doersch, and Andrew Zisserman. “Exploiting temporal context for 3D human pose estimation in the wild”. In: *CVPR*. 2019.

[6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: *arXiv preprint arXiv:1607.06450* (2016).

[7] Fabien Baradel et al. “Leveraging MoCap data for human mesh recovery”. In: *3DV*. 2021.

[8] Fabien Baradel et al. “PoseBERT: A Generic Transformer Module for Temporal 3D Human Modeling”. In: *PAMI* (2022).

[9] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. “Tracking without bells and whistles”. In: *ICCV*. 2019.

[10] Fausto Bernardini et al. “The ball-pivoting algorithm for surface reconstruction”. In: *IEEE transactions on visualization and computer graphics* 5.4 (1999), pp. 349–359.

[11] Paul J Besl and Neil D McKay. “Method for registration of 3-D shapes”. In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. International Society for Optics and Photonics. 1992, pp. 586–606.

[12] Volker Blanz and Thomas Vetter. “A morphable model for the synthesis of 3D faces”. In: *SIGGRAPH*. 1999.

[13] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Institute, Amsterdam, 2019. URL: http://www.blender.org.

[14] Federica Bogo et al. “Keep it SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image”. In: *ECCV*. 2016.

[15] Andrew Brock, Jeff Donahue, and Karen Simonyan. “Large scale GAN training for high fidelity natural image synthesis”. In: *ICLR* (2019).

[16] Nicolas Carion et al. “End-to-end object detection with transformers”. In: *ECCV*. 2020.

[17] Thomas J. Cashman and Andrew W. Fitzgibbon. “What Shape Are Dolphins? Building 3D Morphable Models from 2D Images”. In: *TPAMI* (2013).

[18] Wenzheng Chen et al. “Learning to predict 3d objects with an interpolation-based differentiable renderer”. In: *NeurIPS*. 2019.

[19] Wenzheng Chen et al. “Synthesizing training images for boosting human 3d pose estimation”. In: *3DV*. 2016.

[20] Junhyeong Cho, Kim Youwang, and Tae-Hyun Oh. “Cross-Attention of Disentangled Modalities for 3D Human Mesh Recovery with Transformers”. In: *ECCV*. 2022.

[21] Hongsuk Choi et al. “Beyond static features for temporally consistent 3D human pose and shape from a video”. In: *CVPR*. 2021.

[22] Vasileios Choutas et al. “PoTion: Pose motion representation for action recognition”. In: *CVPR*. 2018.

[23] Christopher B Choy et al. “3d-r2n2: A unified approach for single and multi-view 3d object reconstruction”. In: *ECCV*. 2016.

[24] Jasmine Collins et al. “ABO: Dataset and Benchmarks for Real-World 3D Object Understanding”. In: *CVPR* (2022).

[25] Yuchao Dai, Hongdong Li, and Mingyi He. “A simple prior-free method for non-rigid structure-from-motion factorization”. In: *IJCV* (2014).

[26] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. “Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach”. In: *CGIT*. 1996.

[27] Matt Deitke et al. “Objaverse: A Universe of Annotated 3D Objects”. In: *arXiv preprint arXiv:2212.08051* (2022).

[28] Amaël Delaunoy and Marc Pollefeys. “Photometric bundle adjustment for dense multi-view 3d modeling”. In: *CVPR*. 2014, pp. 1486–1493.

[29] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[30] Carl Doersch and Andrew Zisserman. “Sim2real transfer learning for 3D human pose estimation: Motion to the rescue”. In: *NeurIPS* (2019).

[31] Alexey Dosovitskiy et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *ICLR* (2021).

[32] Shivam Duggal and Deepak Pathak. “Topologically-Aware Deformation Fields for Single-View 3D Reconstruction”. In: *CVPR* (2022).

[33] Haoqi Fan et al. “Multiscale vision transformers”. In: *ICCV*. 2021.

[34] Haoqiang Fan, Hao Su, and Leonidas J Guibas. "A Point Set Generation Network for 3D Object Reconstruction From a Single Image". In: *CVPR*. 2017.

[35] Hao-Shu Fang et al. "RMPE: Regional multi-person pose estimation". In: *ICCV*. 2017.

[36] Olivier Faugeras and Renaud Keriven. "Complete dense stereovision using level set methods". In: *ECCV*. Springer. 1998, pp. 379–393.

[37] Christoph Feichtenhofer et al. "Slowfast networks for video recognition". In: *ICCV*. 2019.

[38] Mihai Fieraru et al. "Three-dimensional reconstruction of human interactions". In: *CVPR*. 2020.

[39] William T Freeman. "The generic viewpoint assumption in a framework for visual perception". In: *Nature* 368.6471 (1994), pp. 542–545.

[40] Yasutaka Furukawa and Jean Ponce. "Accurate, dense, and robust multiview stereopsis". In: *TPAMI* (2009).

[41] Matheus Gadelha, Subhransu Maji, and Rui Wang. "3d shape induction from 2d views of multiple objects". In: *3DV*. 2017.

[42] Andreas Geiger et al. "Vision meets robotics: The kitti dataset". In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237.

[43] Georgios Georgakis et al. "Hierarchical kinematic human mesh recovery". In: *ECCV*. 2020.

[44] R. Girdhar et al. "Learning a Predictable and Generative Vector Representation for Objects". In: *ECCV*. 2016.

[45] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. "Mesh R-CNN". In: *ICCV*. 2019.

[46] Shubham Goel, Georgia Gkioxari, and Jitendra Malik. "Differentiable Stereopsis: Meshes From Multiple Views Using Differentiable Rendering". In: *CVPR*. June 2022, pp. 8635–8644.

[47] Shubham Goel, Angjoo Kanazawa, and Jitendra Malik. "Shape and viewpoint without keypoints". In: *ECCV*. Springer. 2020, pp. 88–104.

[48] Shubham Goel et al. "Humans in 4D: Reconstructing and Tracking Humans with Transformers". In: *arXiv preprint* (2023).

[49] Chunhui Gu et al. "AVA: A video dataset of spatio-temporally localized atomic visual actions". In: *CVPR*. 2018.

[50] Peng Guan et al. "Estimating human shape and pose from a single image". In: *ICCV*. 2009.

[51] Riza Alp Guler and Iasonas Kokkinos. "HoloPose: Holistic 3D human reconstruction in-the-wild". In: *CVPR*. 2019.

[52] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[53] Kaiming He et al. "Mask R-CNN". In: *ICCV*. 2017.

[54] Kaiming He et al. “Masked autoencoders are scalable vision learners”. In: *CVPR*. 2022.

[55] Nikolas Hesse et al. “Learning and tracking the 3D body shape of freely moving infants from RGB-D sequences”. In: *PAMI* (2019).

[56] Vu Hoang Hiep et al. “Towards high-resolution large-scale multi-view stereo”. In: *CVPR*. IEEE. 2009, pp. 1430–1437.

[57] John F Hughes and James D Foley. *Computer graphics: principles and practice*. Pearson Education, 2014.

[58] Eldar Insafutdinov and Alexey Dosovitskiy. “Unsupervised learning of shape and pose with differentiable point clouds”. In: *NeurIPS*. 2018.

[59] Catalin Ionescu et al. “Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments”. In: *PAMI* (2013).

[60] Michal Irani, P Anandan, and Meir Cohen. “Direct Recovery of Planar-Parallax from Multiple Frames”. In: *TPAMI* (2002).

[61] Allan Jabri, David Fleet, and Ting Chen. “Scalable Adaptive Computation for Iterative Generation”. In: *arXiv preprint arXiv:2212.11972* (2022).

[62] Tomas Jakab et al. “Unsupervised learning of object landmarks through conditional image generation”. In: *NeurIPS*. 2018.

[63] Rasmus Jensen et al. “Large scale multi-view stereopsis evaluation”. In: *CVPR*. IEEE. 2014, pp. 406–413.

[64] Sam Johnson and Mark Everingham. “Learning effective human pose estimation from inaccurate annotation”. In: *CVPR*. 2011.

[65] Hanbyul Joo, Natalia Neverova, and Andrea Vedaldi. “Exemplar fine-tuning for 3D human model fitting towards in-the-wild 3D human pose estimation”. In: *3DV*. 2021.

[66] Hanbyul Joo et al. “Panoptic Studio: A Massively Multiview System for Social Motion Capture”. In: (2015).

[67] Angjoo Kanazawa et al. “End-to-end recovery of human shape and pose”. In: *CVPR*. 2018.

[68] Angjoo Kanazawa et al. “Learning 3D deformation of animals from 2D images”. In: *Computer Graphics Forum*. Wiley Online Library. 2016.

[69] Angjoo Kanazawa et al. “Learning 3D human dynamics from video”. In: *CVPR*. 2019.

[70] Angjoo Kanazawa et al. “Learning Category-Specific Mesh Reconstruction from Image Collections”. In: *ECCV*. 2018.

[71] Angjoo Kanazawa et al. “Learning category-specific mesh reconstruction from image collections”. In: *ECCV*. 2018.

[72] Abhishek Kar, Christian Häne, and Jitendra Malik. “Learning a Multi-View Stereo Machine”. In: *NeurIPS*. 2017.

[73] Abhishek Kar et al. “Category-Specific Object Reconstruction from a Single Image”. In: *CVPR*. 2015.

[74] Rangachar Kasturi et al. “Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol”. In: *PAMI* (2008).

[75] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. “Neural 3D Mesh Renderer”. In: *CVPR*. 2018.

[76] Will Kay et al. “The kinetics human action video dataset”. In: *arXiv preprint arXiv:1705.06950* (2017).

[77] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. “Poisson surface reconstruction”. In: *Proceedings of the fourth Eurographics symposium on Geometry processing*. Vol. 7. 2006.

[78] Alexander Kirillov et al. “PointRend: Image Segmentation as Rendering”. In: *CVPR*. 2020.

[79] Arno Knapitsch et al. “Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction”. In: *ACM Transactions on Graphics* (2017).

[80] Muhammed Kocabas, Nikos Athanasiou, and Michael J Black. “VIBE: Video inference for human body pose and shape estimation”. In: *CVPR*. 2020.

[81] Muhammed Kocabas et al. “PARE: Part attention regressor for 3D human body estimation”. In: *ICCV*. 2021.

[82] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. “Convolutional mesh regression for single-image human shape reconstruction”. In: *CVPR*. 2019.

[83] Nikos Kolotouros et al. “Learning to reconstruct 3D human pose and shape via model-fitting in the loop”. In: *ICCV*. 2019.

[84] Nikos Kolotouros et al. “Probabilistic modeling for human mesh recovery”. In: *ICCV*. 2021.

[85] Nilesh Kulkarni, Abhinav Gupta, and Shubham Tulsiani. “Canonical Surface Mapping via Geometric Cycle Consistency”. In: *ICCV*. 2019.

[86] Nilesh Kulkarni et al. “Articulation-aware canonical surface mapping”. In: *CVPR*. 2020, pp. 452–461.

[87] Christoph Lassner et al. “Unite the people: Closing the loop between 3D and 2D human representations”. In: *CVPR*. 2017.

[88] Aldo Laurentini. “The visual hull concept for silhouette-based image understanding”. In: *TPAMI* 16.2 (1994), pp. 150–162.

[89] Vincent Leroy et al. “SMPLy benchmarking 3D human pose estimation in the wild”. In: *3DV*. 2020.

[90] Xueting Li et al. “Self-supervised single-view 3D reconstruction via semantic consistency”. In: *ECCV*. Springer. 2020, pp. 677–693.

[91] Yanghao Li et al. “Exploring plain vision transformer backbones for object detection”. In: *ECCV*. 2022.

[92] Zhengqi Li and Noah Snavely. “Megadepth: Learning single-view depth prediction from internet photos”. In: *CVPR*. 2018, pp. 2041–2050.

[93] Zhihao Li et al. “CLIFF: Carrying location information in full frames into human pose and shape estimation”. In: *ECCV*. 2022.

[94] Kevin Lin, Lijuan Wang, and Zicheng Liu. “End-to-end human pose and mesh reconstruction with transformers”. In: *CVPR*. 2021.

[95] Kevin Lin, Lijuan Wang, and Zicheng Liu. “Mesh graphormer”. In: *ICCV*. 2021.

[96] Tsung-Yi Lin et al. “Microsoft COCO: Common objects in context”. In: *ECCV*. 2014.

[97] Ruoshi Liu et al. *Zero-1-to-3: Zero-shot One Image to 3D Object*. 2023. arXiv: 2303.11328 [cs.CV].

[98] Shichen Liu et al. “Soft rasterizer: A differentiable renderer for image-based 3d reasoning”. In: *ICCV*. 2019.

[99] Stephen Lombardi et al. “Neural volumes: Learning dynamic renderable volumes from images”. In: *CVPR* (2019).

[100] Matthew Loper et al. “SMPL: A Skinned Multi-Person Linear Model”. In: *ACM Trans. Graphics (Proc. SIGGRAPH Asia)* 34.6 (Oct. 2015), 248:1–248:16.

[101] Matthew M Loper and Michael J Black. “OpenDR: An approximate differentiable renderer”. In: *ECCV*. 2014.

[102] Ilya Loshchilov and Frank Hutter. “Decoupled weight decay regularization”. In: *arXiv preprint arXiv:1711.05101* (2017).

[103] Ilya Loshchilov and Frank Hutter. “SGDR: Stochastic gradient descent with warm restarts”. In: *arXiv preprint arXiv:1608.03983* (2016).

[104] Jonathon Luiten et al. “HOTA: A higher order metric for evaluating multi-object tracking”. In: *IJCV* (2021).

[105] Zhengyi Luo, S Alireza Golestaneh, and Kris M Kitani. “3D human motion estimation via motion compression and refinement”. In: *ACCV*. 2020.

[106] Naureen Mahmood et al. “AMASS: Archive of motion capture as surface shapes”. In: *ICCV*. 2019.

[107] Manuel Marques and João Costeira. “Estimating 3D shape from degenerate sequences with missing data”. In: *CVIU* (2009).

[108] Dushyant Mehta et al. “Monocular 3D human pose estimation in the wild using improved CNN supervision”. In: *3DV*. 2017.

[109] Tim Meinhardt et al. “TrackFormer: Multi-object tracking with transformers”. In: *CVPR*. 2022.

[110] Ben Mildenhall et al. “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis”. In: *ECCV*. 2020.

[111] Patrick Min. *Binvox*. http://www.patrickmin.com/binvox. 2004 - 2019.

[112] Gyeongsik Moon and Kyoung Mu Lee. “I2L-MeshNet: Image-to-lixel prediction network for accurate 3D human pose and mesh estimation from a single RGB image”. In: *ECCV*. 2020.

[113] Andrew Nealen et al. “Laplacian mesh optimization”. In: *CGIT*. 2006.

[114] Alex Nichol et al. “Point-E: A System for Generating 3D Point Clouds from Complex Prompts”. In: *arXiv preprint arXiv:2212.08751* (2022).

[115] Michael Niemeyer et al. “Differentiable Volumetric Rendering: Learning Implicit 3D Representations Without 3D Supervision”. In: *CVPR*. June 2020.

[116] Fakir S. Nooruddin and Greg Turk. “Simplification and Repair of Polygonal Models Using Volumetric Techniques”. In: *IEEE Transactions on Visualization and Computer Graphics* (2003).

[117] David Novotny et al. “C3DPO: Canonical 3D Pose Networks for Non-Rigid Structure From Motion”. In: *ICCV*. 2019.

[118] Mohamed Omran et al. “Neural body fitting: Unifying deep learning and model based human pose and shape estimation”. In: *3DV*. 2018.

[119] Ahmed A A Osman, Timo Bolkart, and Michael J. Black. “STAR: A Sparse Trained Articulated Human Body Regressor”. In: *European Conference on Computer Vision (ECCV)*. 2020, pp. 598–613. URL: https://star.is.tue.mpg.de.

[120] Jeong Joon Park et al. “Deepsdf: Learning continuous signed distance functions for shape representation”. In: *CVPR*. 2019.

[121] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *NeurIPS*. 2019.

[122] Austin Patel et al. “Learning to Imitate Object Interactions from Internet Videos”. In: *arXiv preprint arXiv:2211.13225* (2022).

[123] Priyanka Patel et al. “AGORA: Avatars in geography optimized for regression analysis”. In: *CVPR*. 2021.

[124] Georgios Pavlakos, Jitendra Malik, and Angjoo Kanazawa. “Human mesh recovery from multiple shots”. In: *CVPR*. 2022.

[125] Georgios Pavlakos et al. “Expressive body capture: 3D hands, face, and body from a single image”. In: *CVPR*. 2019.

[126] Georgios Pavlakos et al. “Learning to estimate 3D human pose and shape from a single color image”. In: *CVPR*. 2018.

[127] William Peebles and Saining Xie. “Scalable Diffusion Models with Transformers”. In: *arXiv preprint arXiv:2212.09748* (2022).

[128] Xue Bin Peng et al. “SFV: Reinforcement learning of physical skills from videos”. In: *ACM Transactions On Graphics (TOG)* 37.6 (2018), pp. 1–14.

[129] Ulrich Pinkall and Konrad Polthier. “Computing discrete minimal surfaces and their conjugates”. In: *Experimental mathematics* (1993).

[130] Jean-Philippe Pons, Renaud Keriven, and Olivier Faugeras. “Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score”. In: *IJCV* 72.2 (2007), pp. 179–193.

[131] Ben Poole et al. “DreamFusion: Text-to-3D using 2D Diffusion”. In: *arXiv* (2022).

[132] Jathushan Rajasegaran et al. “On the Benefits of 3D tracking and Pose for Human Action Recognition”. In: *CVPR*. 2023.

[133] Jathushan Rajasegaran et al. “Tracking people by predicting 3D appearance, location and pose”. In: *CVPR*. 2022.

[134] Jathushan Rajasegaran et al. “Tracking people with 3D representations”. In: *NeurIPS*. 2021.

[135] Nikhila Ravi et al. “Accelerating 3D Deep Learning with PyTorch3D”. In: *arXiv:2007.08501* (2020).

[136] Davis Rempe et al. “HuMoR: 3D human motion model for robust pose estimation”. In: *ICCV*. 2021.

[137] Google Research. *Scanned Objects Dataset*. http://goo.gle/scanned-objects. 2020.

[138] Ergys Ristani et al. “Performance measures and a data set for multi-target, multi-camera tracking”. In: *ECCV*. 2016.

[139] Lawrence G Roberts. “Machine perception of three-dimensional solids”. PhD thesis. Massachusetts Institute of Technology, 1963.

[140] Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2021. arXiv: 2112.10752 [cs.CV].

[141] Johannes Lutz Schönberger and Jan-Michael Frahm. “Structure-from-Motion Revisited”. In: *CVPR*. 2016.

[142] Johannes Lutz Schönberger et al. “Pixelwise View Selection for Unstructured Multi-View Stereo”. In: *ECCV*. 2016.

[143] Christoph Schuhmann et al. “Laion-5b: An open large-scale dataset for training next generation image-text models”. In: *NeurIPS* (2022).

[144] Anshul Shah et al. “Pose and joint-aware action recognition”. In: *WACV*. 2022.

[145] Zhixin Shu et al. “Deforming autoencoders: Unsupervised disentangling of shape and appearance”. In: *ECCV*. 2018.

[146] Pawan Sinha and Edward Adelson. “Recovering reflectance and illumination in a world of painted polyhedra”. In: *ICCV*. IEEE. 1993, pp. 156–163.

[147] Jie Song, Xu Chen, and Otmar Hilliges. "Human body model fitting by learned gradient descent". In: *ECCV*. 2020.

[148] Shuran Song et al. "Semantic scene completion from a single depth image". In: *CVPR*. 2017.

[149] Kasey C Soska, Karen E Adolph, and Scott P Johnson. "Systems in development: motor skill acquisition facilitates three-dimensional object completion." In: *Developmental psychology* 46.1 (2010), p. 129.

[150] Yu Sun et al. "Human mesh recovery from monocular images via a skeleton-disentangled representation". In: *ICCV*. 2019.

[151] Yu Sun et al. "Putting people in their place: Monocular regression of 3D people in depth". In: *CVPR*. 2022.

[152] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science and Business Media, 2010.

[153] James Thewlis, Hakan Bilen, and Andrea Vedaldi. "Unsupervised learning of object frames by dense equivariant image labelling". In: *NeurIPS*. 2017.

[154] D'Arcy Thompson. *On Growth and Form*. Cambridge Univ. Press, 1917.

[155] Garvita Tiwari et al. "Pose-NDF: Modeling human pose manifolds with neural distance fields". In: *ECCV*. 2022.

[156] Lorenzo Torresani, Aaron Hertzmann, and Chris Bregler. "Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors". In: *TPAMI* (2008).

[157] Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. "Multi-view Consistency as Supervisory Signal for Learning Shape and Pose Prediction". In: *CVPR*. 2018.

[158] Shubham Tulsiani, Nilesh Kulkarni, and Abhinav Gupta. "Implicit mesh reconstruction from unannotated image collections". In: *arXiv preprint arXiv:2007.08504* (2020).

[159] Shubham Tulsiani et al. "Factoring Shape, Pose, and Layout from the 2D Image of a 3D Scene". In: *CVPR*. 2018.

[160] Shubham Tulsiani et al. "Multi-view Supervision for Single-view Reconstruction via Differentiable Ray Consistency". In: *CVPR*. 2017.

[161] Shimon Ullman. "The interpretation of structure from motion". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 203.1153 (1979), pp. 405–426.

[162] Gül Varol et al. "Learning from Synthetic Humans". In: *CVPR*. 2017.

[163] Vasileios Vasilopoulos et al. "Reactive semantic planning in unexplored semantic environments using deep perceptual feedback". In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4455–4462.

[164] Ashish Vaswani et al. "Attention is all you need". In: *NIPS*. 2017.

[165] Sara Vicente et al. "Reconstructing PASCAL VOC". In: *CVPR*. 2014.

[166] C. Wah et al. *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. rep. CNS-TR-2011-001. California Institute of Technology, 2011.

[167] Ziniu Wan et al. "Encoder-decoder with multi-level attention for 3D human shape and pose estimation". In: *ICCV*. 2021.

[168] Chung-Yi Weng et al. "HumanNeRF: Free-viewpoint rendering of moving people from monocular video". In: *CVPR*. 2022.

[169] Charles Wheatstone. "Contributions to the physiology of vision. —Part the first. On some remarkable, and hitherto unobserved, phenomena of binocular vision". In: (1838).

[170] Chao-Yuan Wu et al. "Multiview Compressive Coding for 3D Reconstruction". In: *CVPR*. 2023.

[171] Jiahong Wu et al. "AI Challenger: A large-scale dataset for going deeper in image understanding". In: *arXiv preprint arXiv:1711.06475* (2017).

[172] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. "Unsupervised Learning of Probably Symmetric Deformable 3D Objects from Images in the Wild". In: *arXiv preprint arXiv:1911.11130* (2019).

[173] Zhirong Wu et al. "3D ShapeNets: A deep representation for volumetric shapes". In: *CVPR*. 2015.

[174] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. "Beyond pascal: A benchmark for 3d object detection in the wild". In: *WACV*. 2014.

[175] Yuliang Xiu et al. "Pose Flow: Efficient online pose tracking". In: *BMVC*. 2018.

[176] Xiangyu Xu et al. "3D human pose, shape and texture from low-resolution images and videos". In: *PAMI* (2021).

[177] Yuanlu Xu, Song-Chun Zhu, and Tony Tung. "DenseRaC: Joint 3D pose and shape estimation by dense render-and-compare". In: *ICCV*. 2019.

[178] Yufei Xu et al. "ViTPose: Simple Vision Transformer Baselines for Human Pose Estimation". In: *NeurIPS*. 2022.

[179] Jianfeng Yan et al. "Dense Hybrid Recurrent Multi-view Stereo Net with Dynamic Consistency Checking". In: *ECCV*. 2020.

[180] Xinchen Yan et al. "Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision". In: *NeurIPS*. 2016.

[181] Yi Yang and Deva Ramanan. "Articulated human detection with flexible mixtures of parts". In: *PAMI* (2012).

[182] Yao Yao et al. "MVSNet: Depth inference for unstructured multi-view stereo". In: *ECCV*. 2018, pp. 767–783.

[183] Lior Yariv et al. "Multiview Neural Surface Reconstruction by Disentangling Geometry and Appearance". In: *NeurIPS* 33 (2020).

[184] Vickie Ye et al. “Decoupling Human and Camera Motion from Videos in the Wild”. In: *CVPR*. 2023.

[185] Yufei Ye, Shubham Tulsiani, and Abhinav Gupta. “Shelf-Supervised Mesh Prediction in the Wild”. In: *arXiv preprint arXiv:2102.06195* (2021).

[186] Zhichao Yin and Jianping Shi. “Geonet: Unsupervised learning of dense depth, optical flow and camera pose”. In: *CVPR*. 2018, pp. 1983–1992.

[187] Ye Yuan et al. “GLAMR: Global occlusion-aware human mesh recovery with dynamic cameras”. In: *CVPR*. 2022.

[188] 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: *CVPR*. 2018.

[189] Wang Zeng et al. “3D human mesh regression with dense correspondence”. In: *CVPR*. 2020.

[190] Hongwen Zhang et al. “DaNnet: Decompose-and-aggregate network for 3D human shape and pose estimation”. In: *Proceedings of the 27th ACM International Conference on Multimedia*. 2019.

[191] Hongwen Zhang et al. “PyMAF-X: Towards well-aligned full-body model regression from monocular images”. In: *arXiv preprint arXiv:2207.06400* (2022).

[192] Hongwen Zhang et al. “PyMAF: 3D human pose and shape regression with pyramidal mesh alignment feedback loop”. In: *ICCV*. 2021.

[193] Jason Y Zhang et al. “Predicting 3D human dynamics from video”. In: *ICCV*. 2019.

[194] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Networks as a Perceptual Metric”. In: *CVPR*. 2018.

[195] Xiuming Zhang et al. “Learning to reconstruct shapes from unseen classes”. In: *NeurIPS*. 2018.

[196] Yuyang Zhang et al. “Unsupervised Multi-View Constrained Convolutional Network for Accurate Depth Estimation”. In: *IEEE Transactions on Image Processing* 29 (2020), pp. 7019–7031.

[197] Tinghui Zhou et al. “Stereo magnification: Learning view synthesis using multiplane images”. In: *SIGGRAPH* (2018).

[198] Tinghui Zhou et al. “Unsupervised learning of depth and ego-motion from video”. In: *CVPR*. 2017, pp. 1851–1858.

[199] Yi Zhou et al. “On the continuity of rotation representations in neural networks”. In: *CVPR*. 2019.

[200] Silvia Zuffi et al. “Three-D Safari: Learning to Estimate Zebra Pose, Shape, and Texture from Images "In the Wild"”. In: *ICCV*. 2019.

Appendix A

Chapter 2 Supplementary Material

A.1 Additional Training Details

Misc

We downsample the original 2048×2048 images to 400×400 for all methods but COLMAP. Both NeRF baselines are trained for 100k iterations and take $\sim$ 10 hours to run on a 32GB NVIDIA V100. DS is trained for 50k iterations and takes $\sim$ 9 hours to run on the same GPU.

Bi-directional distance transform loss

This section describes the bi-directional distance transform loss $\mathcal{L}_{\text{bi-dt}}$ used in the mask reconstruction loss in Equation 6 of the main text. Our novel loss is a differentiable version of a non-differentiable naive bi-directional distance transform loss. We first define the naive non-differentiable loss and then describe our differentiable adaptation to it. For convenience, we misuse A (or A^r) to refer to the set of pixels where GT mask A (or the rendered mask A^r) is 1.

A naive non-differentiable bi-directional distance transform loss $\mathcal{L}_{\text{bi-dt-naive}}(A^r, A)$ between the rendered mask A^r and GT mask A consists of 2 components. The first penalizes pixels $p \in A^r \setminus A$, where the rendered mask is 1 but GT mask is 0, by the distance of p from $\texttt{NN}(p, A)$ - the closest occupied pixel in the GT mask. $\texttt{NN}$ is the nearest neighbor operation in euclidean space. The second component penalizes pixels $q \in A \setminus A^r$, where the GT mask is 1 but rendered mask is 0, by the distance of q from $\texttt{NN}(q, A^r)$ - the closest occupied pixel in the rendered mask.

$$\mathcal{L}_{\text{bi-dt-naive}}(A^r, A) = \sum_{p \in A^r \setminus A} \texttt{NN}_d(p, A) + \sum_{q \in A \setminus A^r} \texttt{NN}_d(q, A^r)) \tag{A.1}$$

where $\texttt{NN}_d(p, A) = d(p, \texttt{NN}(p, A))$ is the distance between p and its nearest neighbor in A as measured by a metric d.

The operation of finding the set of pixel locations where the rendered mask A^r is occupied is the only non-differentiable one in the loss above. Our novel adaptation provides a differentiable approximation for the same. Given a pixel p in i-th image I_i, recall (from Section 3.1) that the

rasterizer finds K points $x_{1..K}$ on the surface of the mesh that project to p under camera π_i. Therefore, $\pi_i(x_k)$ is a differentiable approximation to p for each x_k. We approximate $p \sim \hat{p} = \sum_k w_k \pi_i(x_k)$ as a normalized weighted sum of projections $\pi_i(x_k)$ with weights from the softmax blending that composites texels $c_{1..K}$ into a final color at p. Let $\hat{A^r} = \{\hat{p} | p \in A^r; \hat{p} = \sum_k w_k \pi_i(x_k)\}$. The full differentiable bi-directional distance transform loss is

$$\mathcal{L}_{\text{bi-dt}}(A^r, A) = \sum_{p \in A^r \setminus A} \texttt{NN}_d(\hat{p}, A) + \sum_{q \in A \setminus A^r} \texttt{NN}_d(q, \hat{A^r})) \tag{A.2}$$

where $d(x, y) = \texttt{clamp}(||x - y||_2, \tau_{\text{min}}, \tau_{\text{max}})$ is the clamped L-2 euclidean distance. We set τ_{min} to 2 pixels and τ_{max} to 0.1 of the shorter image dimension.

Learning Schedule

We run our optimization for 50k iterations using SGD with a momentum of 0.9 and an initial learning rate of 0.01. We use cosine annealing for our learning rate with warm restarts [103] which we find helps avoid local minima for both shape and cameras. We clip gradient norms for stability. For mesh rasterization, we use PyTorch3D [135] with $K = 6$ and a blur radius that decays exponentially from 5×10^{-5} to 10^{-6} over the course of optimization.

Aligning meshes for benchmarking

As cameras are optimized, ground-truth and optimized shapes are not in a common coordinate space and must be aligned before benchmarking. However, different benchmarking metrics are minimized by different alignments. Therefore, for every instance, for each metric, we find 3 possible alignments and pick the best. The first is a brute force minimization of Chamfer-L2 over scale/depth in camera 0's view coordinate space. The second and third additionally optimize rotation and translation via ICP from ground-truth to predicted mesh and vice-versa.

A.2 Additional Results

COLMAP

In Fig. A.5, we show dense pointcloud reconstructions by COLMAP on scenes from the Tanks and Temples dataset as the number of views reduces from 100 to 50 to 25 to 15. The pointcloud density drops drastically as we reduce the number of views below 50. With 15 views, the reconstruction is practically empty and the scene is unrecognizable. Our attempts to mesh these points using COLMAP's Poisson and Delaunay meshers failed. In contrast, as seen in Fig. 8 of the main text, our approach is far more robust with the same views.

In Fig. A.6, we show dense pointcloud reconstructions by COLMAP on instances from the Google dataset with 8 input views and ground-truth cameras. COLMAP fails on 2/50 instances and of the 48 instances it works on, it only reconstructs parts of the scene that are textured and visible

in close (narrow-baseline) views. It still misses surfaces that are visible in only 2-3 wide baseline views. For example in Fig. A.6, COLMAP fails to reconstruct the back of the green backpack which is visible only in 2/8 input views.

Cameras from SfM We tried using COLMAP's Structure-from-Motion (SfM) pipeline to ameliorate the need for noisy camera inputs. It often, if not always, fails to register all cameras into a single coordinate space correctly and usually ends up with multiple groups – each with 2-3 cameras with reasonably accurate relative orientation. The inability to merge the multiple groups into a single coordinate frame made it infeasible to use the COLMAP cameras as initializations. Note that SfM's failure is not surprising since we are working with 8 white-background views covering 360-degrees of the object. On average, any surface is visible from ~ 3 views only. Furthermore, white background images are harder to calibrate than in-context images with background because of the lack of visual overlap in the background.

Further comparisons

In addition to comparisons with NeRF [110] and NeRF-opt in the main paper, we compare to COLMAP [141], IDR [183], and DS-naive. Table A.2 shows results on GSO following the evaluation protocol of the main paper. All methods use 8 views and $\sigma = 30^\circ$ camera noise, except COLMAP, which uses ground-truth cameras. For COLMAP, we compute metrics on the reconstructed dense point cloud via uniform random sampling.

DS outperforms all baselines across all metrics. IDR gets slightly better shapes but worse cameras than NeRF-opt. However, neither can recover accurate geometry under camera noise. Surprisingly, DS-naive performs similar to NeRF-opt but with better Chamfer and Normal Consistency, suggesting the mesh representation is robust in noisy settings. We note that COLMAP has extremely high precision (99.8%, compared to DS's 88.4%) but very poor recall (35.5%, compared to DS's 78.2%) at a threshold of 0.2. This suggests that COLMAP reconstructs accurate point clouds but misses large parts of the shape, supporting the qualitative claim in Appendix A.2. The comparison with COLMAP, a method that finds view correspondences followed by triangulation, suggests that such methods have a hard time reconstructing shapes from limited views.

Tanks and Temples Fig. A.1 compares on Tanks and Temples with 15 views and no camera noise (to be compared to Fig. 2.8 in the main paper). Table A.1 shows the corresponding quantitative comparison following the evaluation protocol of the main paper. However, since the ground-truth pointclouds are hollow (without the bottom), the reported numbers only approximate the quality of the shape. NeRF-opt produces lower quality shapes than DS. With more views and no camera noise, IDR performs slightly better than DS. This is not a surprise. Our approach has an advantage when few views (≤ 12) are available and cameras are noisy. But our approach, even with more views ($= 15$) and no camera noise, reconstructs shapes well, performing better than NeRF-opt and similar to IDR.

DTU dataset Fig. A.2 compares on scenes from the DTU dataset with 6 views and linear camera noise following IDR [183]. Note that this data is easier to reconstruct because the 6 views span

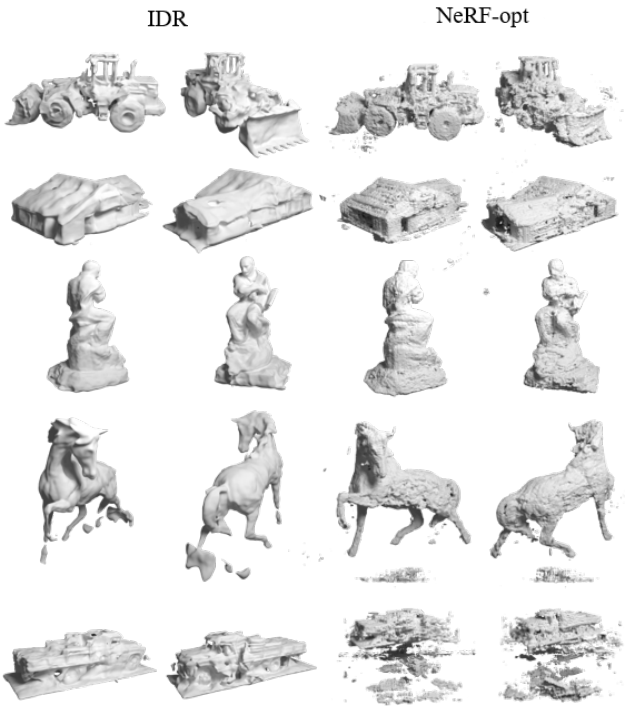

Figure A.1: IDR and NeRF-opt on Tanks and Temples with 15 views.

Scene	Method	Chamfer↓	F1@.1↑	F1@.2↑
Barn	IDR	**0.14**	**47.6**	**70.2**
	NeRF-opt	0.98	33.4	48.2
	DS	0.39	35.1	. 56.0
Caterpillar	IDR	**0.06**	**59.6**	**81.9**
	NeRF-opt	0.15	54.5	77.7
	DS	0.07	55.8	76.4
Ignatius	IDR	**0.18**	**68.2**	**86.9**
	NeRF-opt	0.20	54.6	74.5
	DS	0.30	53.4	75.8
Truck	IDR	**0.06**	**54.8**	**79.7**
	NeRF-opt	1.40	23.9	40.8
	DS	0.14	52.0	70.7

Table A.1: Results on Tanks and Temples scenes with 15 input views.

Method	Chamfer↓	F1@.1↑	F1@.2↑	NC↑	Rot↓
COLMAP	0.38	35.2	52.3	-	-
IDR	0.52	22.0	40.5	0.23	22.8
NeRF-opt	0.31	33.1	58.6	0.28	13.4
DS-naive	0.22	29.5	53.0	0.54	14.7
DS	**0.10**	**63.5**	**80.6**	**0.68**	**0.54**

Table A.2: Results on GSO with 8 views and $\sigma = 30^\circ$ camera noise. NC=Normal Consistency; Rot=Rotation Error (in degrees).

only $1/8^{\text{th}}$ of the entire viewing sphere and the cameras have an average camera noise of only $\sim 1^\circ$, which is much lesser than most of our experiments on GSO. We observe that both DS and IDR outperform NeRF-opt, which has cloudy artifacts. IDR works well in this setting with few views and almost-correct cameras. This is not a surprise as DS has an advantage with few views and noisy cameras. IDR reconstructions are slightly more detailed than DS at some places (*e.g.* at the windows of the house and the feet of the bird). However, IDR shapes are less constrained and contain erroneous blobs (*e.g.* at the belly of the teddy bear, the middle and side of the house, and the head of the boy) because of the lack of a sufficient number of views.

DS

In Fig. A.3 , we show shape and texture reconstruction from DS on more instances in the GSO Dataset.

In Fig. A.4, we show some failure modes of DS. Thin structures and small objects are particularly hard to reconstruct. This is because we use a mesh representation with surface smoothening regularization. DS does not model surface specularities and lighting and subsequently also struggles with specular surfaces.

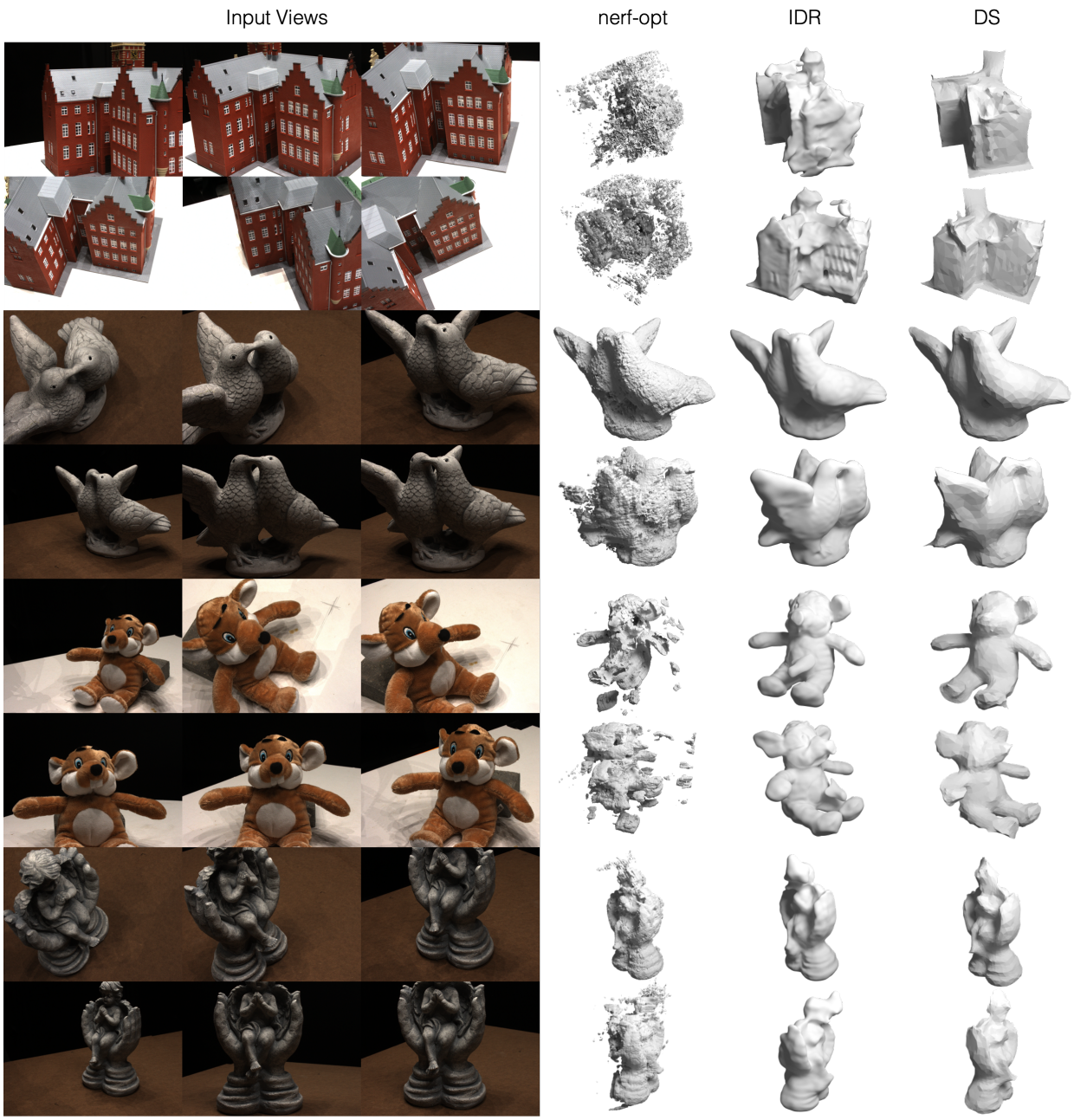

Figure A.2: Results on DTU with 6 views and linear camera noise. Figure compares DS to IDR and NeRF-opt.

Figure A.3: Qualitative shape and texture reconstructions by DS using 8 views and 20° camera noise for more instances from Google's Scanned Objects Dataset

Figure A.4: Failure modes for two instances from Google's Scanned Objects. For the leftmost example, we show the input views (left) and reconstructions (middle) and ground truth shape (right). For the rightmost example, we show input views (left), reconstructions (right top) and ground truth shape (right bottom).

Figure A.5: COLMAP-reconstructed dense pointclouds with varying number of input views from scenes in the Tanks and Temples dataset.

Figure A.6: COLMAP-reconstructed dense pointclouds with 8 input views and ground-truth cameras for objects in the Google Scanned Objects Dataset.

Appendix B

Chapter 3 Supplementary Material

Overview

In this supplementary material, we present additional results including visualization of the shape, texture and camera-multiplex, comparisons to CMR [70], qualitative results on random test samples, an ablation study on texture prediction model, and additional details on the network architecture.

B.1 More Results

PCA in Texture space

In Figure B.1, we visualize the learnt texture space by running PCA on predicted uv-textures across the entire test dataset. On the left, the mean texture is a rather dull gray colour, as expected. On the right, we visualize axes of variation. In the first column, we see low-frequency variations in overall colour, head and belly of the bird. In the second column, we see slightly higher-frequency variations that assign different colors to different parts (head, back, belly and wings) of the bird. We can even recognize eyes and beak in the last two rows on the right.

Learnt shape space for other categories

In Figure B.2, we compare the input template mesh to the learnt shape space for car, motorbike and shoe categories. Observe, on the left, that the input template and learnt mean shape differ substantially for each of these categories - some more than others. For example, the front tire of the motorbike becomes more significantly more prominent, the back of the car becomes more rounded and the shoe becomes slimmer and more elongated. On the right, we visualize the space of learned shapes by running PCA on all shapes obtained on training and test dataset. The three PCA axis visualized show interesting deformations. For example, in the motorcycle, the three visualized axes vary in prominence of front tire, size of fuel-tank and concavity of seat respectively.

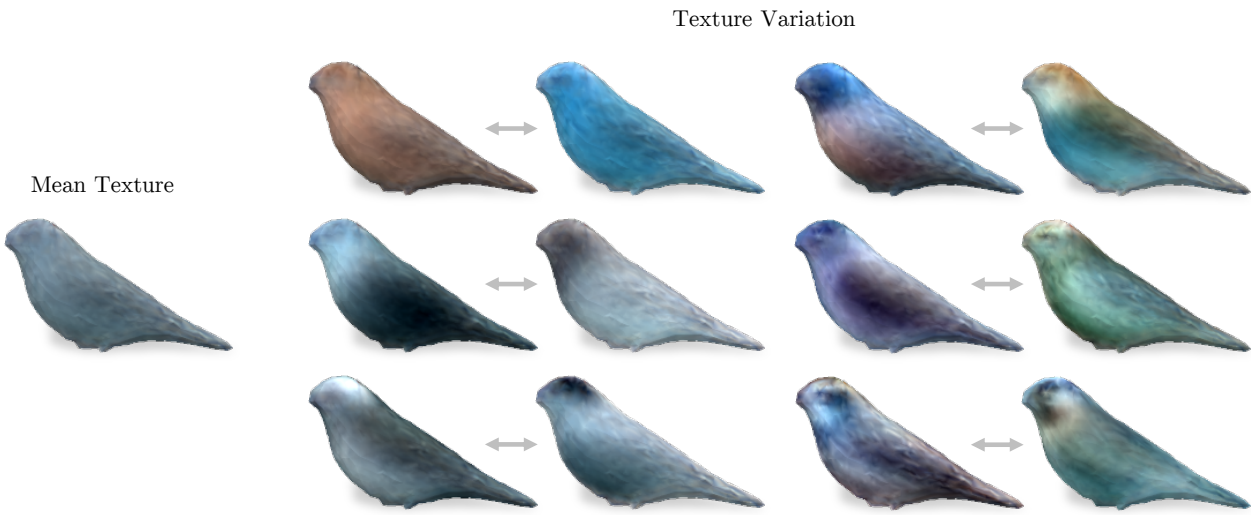

Figure B.1: **PCA in UV-Texture space.** We visualize the learnt texture space by running PCA over all predicted uv-texture maps on the test dataset, and rendering axes of variation on the learnt mean bird shape. See text for discussion.

Camera-multiplex visualization over time.

Figure B.3 shows how the azimuth-elevation distribution of camera poses in the camera-multiplex (over the entire training dataset) changes as training progresses. Observe that the distribution changes rapidly initially and results in a final distribution that is very different from the initial distribution.

Qualitative comparison to CMR.

We qualitatively compare results from U-CMR to 2 variants of CMR [70]. The first is the official CMR model (CMR-official) that was trained using additional keypoint losses and used NMR [75] as it's differentiable renderer. The second is our implementation of CMR (CMR-ours) which is similar to U-CMR in it's architecture for shape/texture, using Softras [98] for rendering, regularizing shape using the graph-laplacian and having the same template mesh as it's initial mean shape, but different from U-CMR in that it uses the ground-truth camera pose from SFM during training. Unlike CMR-official, CMR-ours does not include vertex-keypoint reprojection loss.

In Figures B.5-B.6, we compare CMR-official, CMR-ours and U-CMR. For each input image, the first row is from CMR-official, second is from CMR-ours and the last row is U-CMR. Observe that CMR-official is not as accurate as CMR-ours in capturing the shape and texture of the underlying bird but has pointier beaks and feet because of the keypoint reprojection loss it uses. The figures show that U-CMR shapes are qualitatively very similar to CMR-ours, hence exemplifying our assertion that U-CMR's camera-multiplex optimization alleviates the need for ground-truth camera pose supervision for most cases.

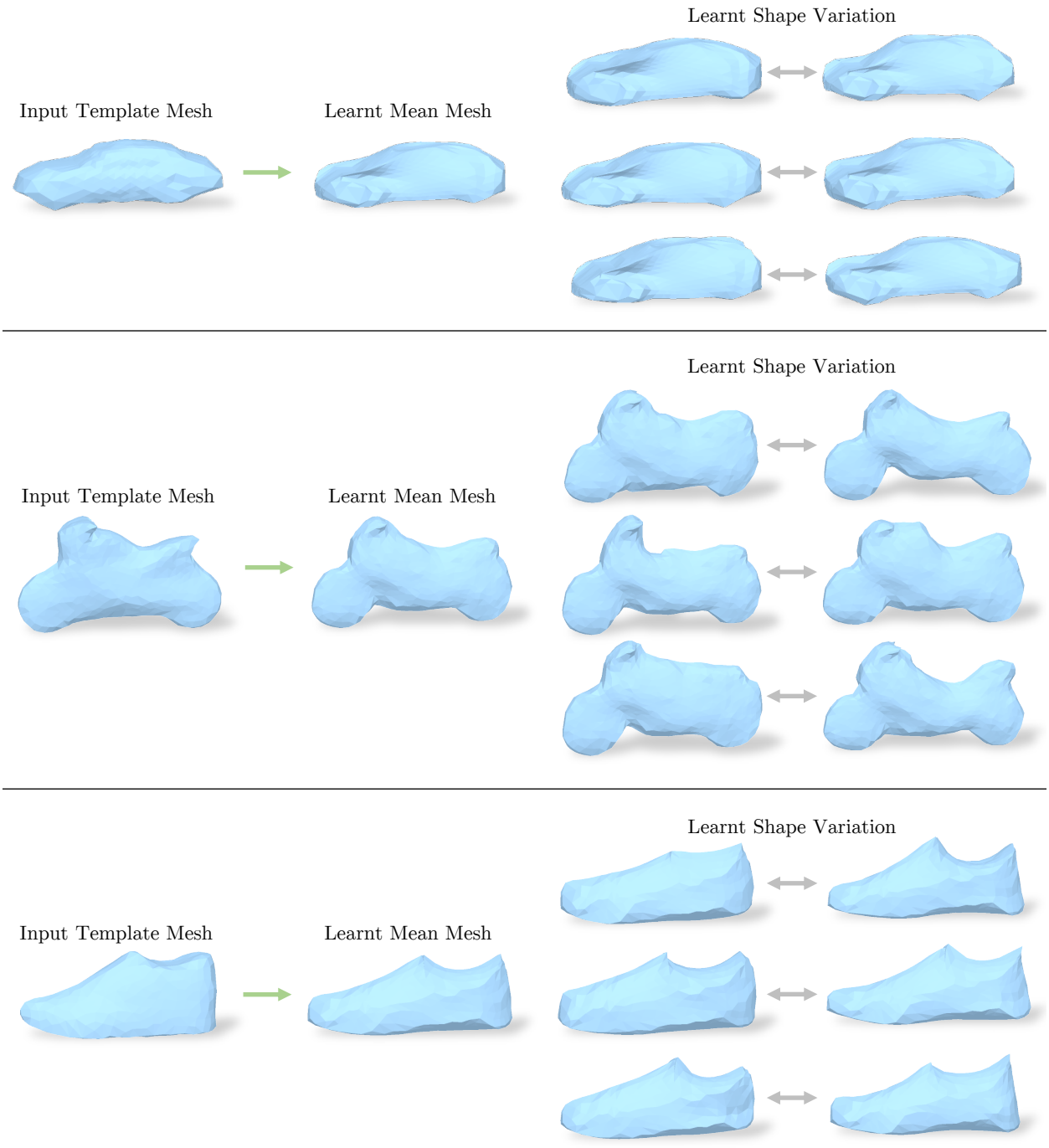

Figure B.2: **Learned Shape on other categories.** On the left, we compare the template shape to the final learnt mean mesh. On the right, we visualize the space of learned shapes by running PCA on all shapes and show three axis of deformations.

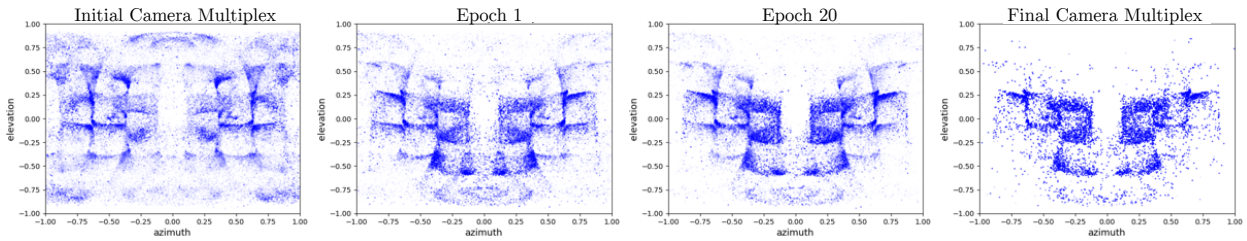

Figure B.3: **Change in Camera Pose Distributions during training.** We show azimuth-elevation scatter plots of $K = 40$ camera poses in all the camera-multiplex of the CUB train set as training progresses. Points corresponding to less probable cameras have a lower alpha value and are more transparent. Starting from top left, we have the camera poses after the camera-multiplex initialization, after 1 training epoch, after 20 training epochs and the final optimized camera poses. The number of camera poses in each multiplex (K) is pruned down from 40 to 4 after epoch 20.

We compare U-CMR and CMR-ours on a random subset of 15 images from the test dataset in Figures B.7-B.9. Observe that U-CMR (second row) accurately predicts shapes that are very similar to those from CMR-ours when the bird is not articulating too much.

Ablation on texture prediction model.

We experiment with two different architectures for predicting the texture. First, we explore predicting texture as texture-flow, which is used in CMR. Texture-flow is a 2D positional offset for every pixel in the UV image that specifies where to sample the RGB values from the input image. Second, we predict the UV image values directly (Texture-gen) using a decoder attached to a bottleneck with spatial dimensions preserved. This is the final approach used in U-CMR. We observed that predicting a flow-field can lead to flat degenerate shapes and a collapse of optimized camera poses. Figure B.4 shows the collapse in the final optimized camera poses when predicting texture as a flow-field. This is because even when the camera pose is wrong, texture-flow is able to learn to adjust to the bad camera, as the output of the texture-flow across different instances is not necessarily correlated. However, when predicting the texture directly through a decoder, the network learns an implicit spatial prior of the texture across the dataset. For example, the network needs to learn to generate the texture of the eye at the same location in every texture map. Similarly for wings, breast, head etc; as the texture map is predicted in a canonical semantic space. This spatial prior that is learned through a spatial decoder allows the texture prediction to disambiguate incorrect and correct poses in the camera-multiplex.

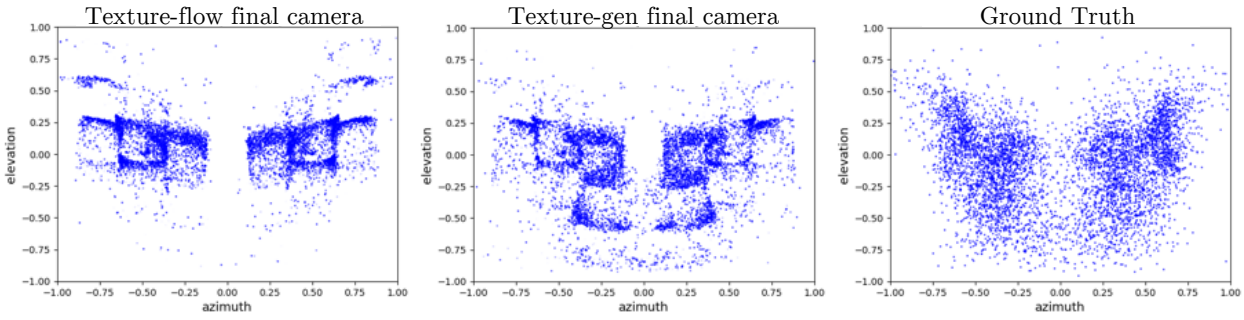

Figure B.4: **Texture-flow camera-multiplex distribution on training set.** On the left, we show the azimuth-elevation distribution of the final camera-multiplex when we predict texture as a flow-field for sampling from the input image. Note how the camera poses have collapsed to 2 broad areas. Center: U-CMR camera-multiplex, Right: GT camera distribution

B.2 Training details

Architecture details.

Our code is available on our project page: https://shubham-goel.github.io/ucmr. The shape and texture predictor f has an encoder-decoder architecture. Image $I \in \mathbb{R}^{256\times256\times3}$ is first encoded to latent feature map $z \in \mathbb{R}^{4\times4\times256}$ using Resnet-18. The shape head takes flattened $z \in \mathbb{R}^{16\times256}$ as input and passes it through 2 fully connected layers, each with 200 output channels and then a final linear layer for predicting $\Delta_V \in \mathbb{R}^{|V|\times3}$. The texture head takes the latent feature map and bilinearly samples it to $z \in \mathbb{R}^{4\times8\times256}$, this is followed by 7 Resnet blocks with 256, 256, 256, 128, 64, 32, 16 output channels respectively, with intermediate bilinear upsampling by a factor of 2 after blocks 1, 3, 4, 5, 6. This is then sent to a final convolution layer that outputs the texture map $I^{uv} \in \mathbb{R}^{128\times256\times3}$. The Resnet encoder uses ReLU activations while the shape and texture heads use Leaky-ReLU activations. All networks use batchnorm for normalization. We will release our code upon publication.

After training the shape and texture prediction with camera-multiplex, we learn the feed-forward camera pose predictor g. We attach this as another head to the latent variable z from the shared Resnet-18 trained for shape and texture prediction with camera-multiplex. We freeze the encoder, and then train a fully connected head for predicting camera scale $s \in \mathbb{R}$, translation $t \in \mathbb{R}^2$ and rotation (as quaternion $q \in \mathbb{R}^4$) through 2 fully connected layer each with 200 channels.

Figure B.5: **CMR-official vs CMR-ours vs U-CMR.** We compare U-CMR (third row) to CMR-ours (second row) and CMR-official (first row) on selected images from the test dataset. The first 2 columns show the predicted shape and texture from the predicted camera viewpoint. The last 2 columns are novel viewpoints of the textured mesh.

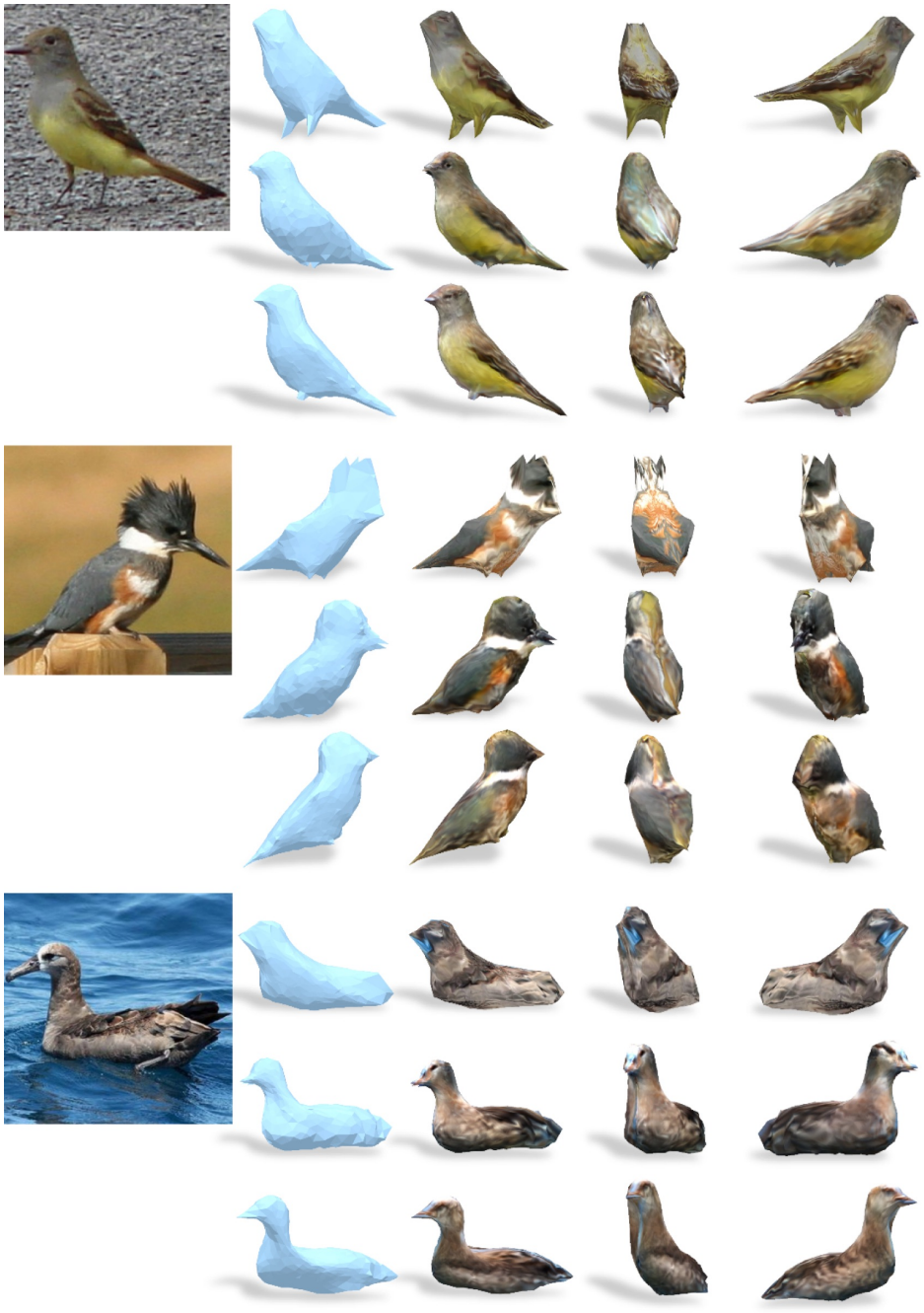

Figure B.6: **CMR-official vs CMR-ours vs U-CMR.** We compare U-CMR (third row) to CMR-ours (second row) and CMR-official (first row) on selected images from the test dataset. The first 2 columns show the predicted shape and texture from the predicted camera viewpoint. The last 2 columns are novel viewpoints of the textured mesh.

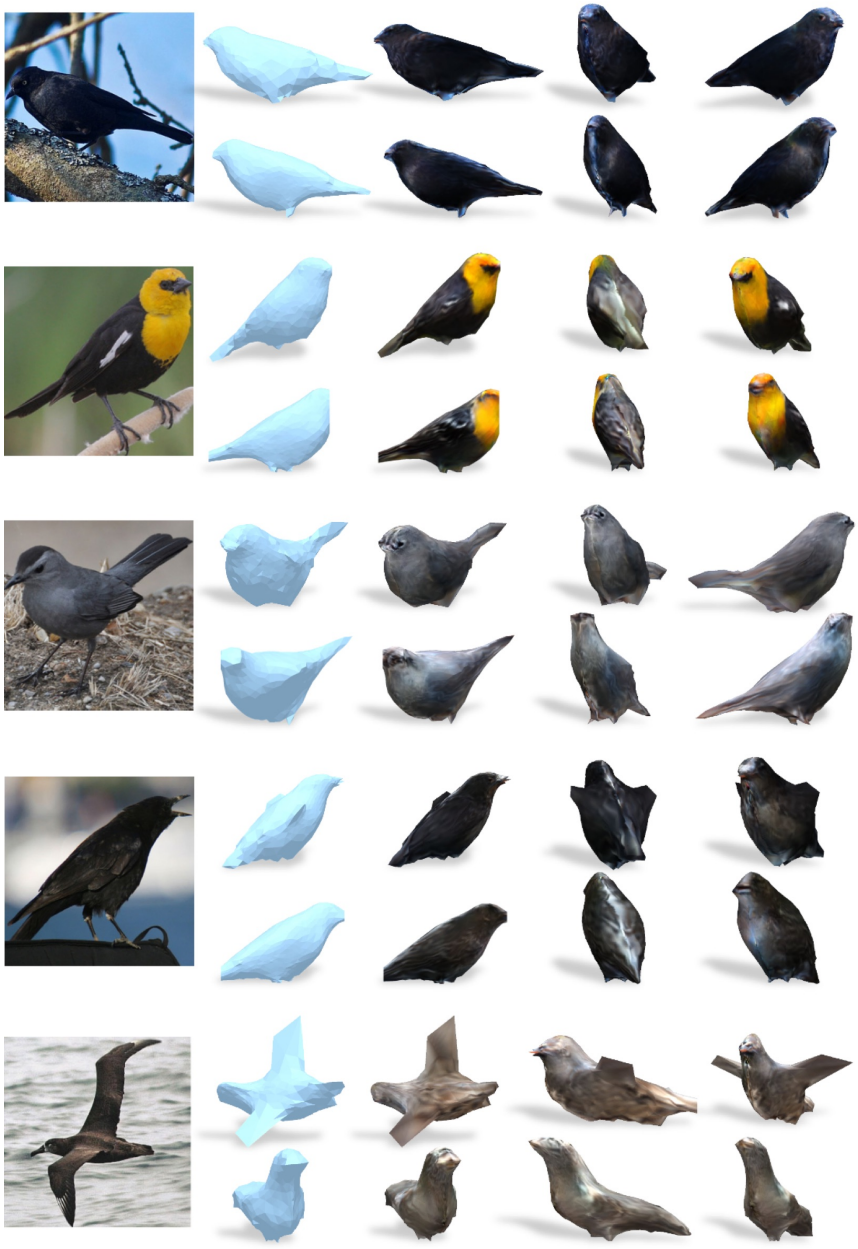

Figure B.7: **CMR-ours vs U-CMR on *random* subset.** We compare U-CMR (second row) to CMR-ours (first row) on a random subset of images from the testset. The first 2 columns show the predicted shape and texture from the predicted camera viewpoint. The last 2 columns are novel viewpoints of the textured mesh.

Figure B.8: **CMR-ours vs U-CMR on *random* subset.** We compare U-CMR (second row) to CMR-ours (first row) on a random subset of images from the test dataset. The first 2 columns show the predicted shape and texture from the predicted camera viewpoint. The last 2 columns are novel viewpoints of the textured mesh.

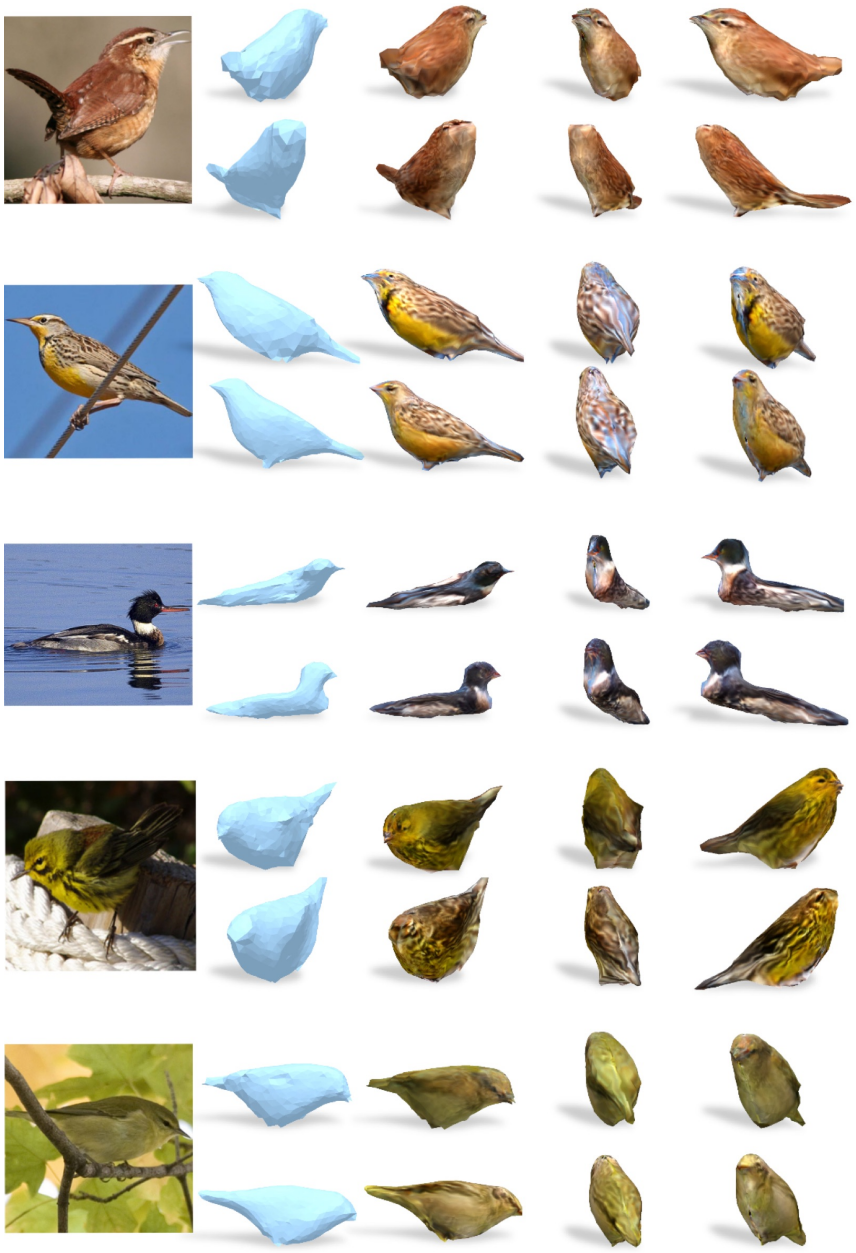

Figure B.9: **CMR-ours vs U-CMR on *random* subset.** We compare U-CMR (second row) to CMR-ours (first row) on a random subset of images from the test dataset. The first 2 columns show the predicted shape and texture from the predicted camera viewpoint. The last 2 columns are novel viewpoints of the textured mesh.

Appendix C

Chapter 4 Supplementary Material

We provide more details about HMR 2.0, *i.e.*, the architecture we use (Section C.1), the data (Section C.2) and the training pipeline (Section C.3). Furthermore, we describe the aspect of pose prediction (Section C.4) and we discuss the metrics we use for evaluation (Section C.5). Then we discuss the experimental settings for tracking (Section C.6), and action recognition (Section C.7). Finally, we provide additional qualitative results (Section C.8).

C.1 HMR 2.0 architecture details

The architecture of our HMR 2.0 model is based on a ViT image encoder and a transformer decoder. We use a ViT-H/16 ("huge") pre-trained on the task of 2D keypoint localization [178]. It has 50 transformer layers, takes a 256×192 sized image as input, and outputs 16×12 image tokens, each of dimension 1280. Our transformer decoder is a standard transformer decoder architecture [164] with 6 layers, each containing multi-head self-attention, multi-head cross-attention, and feed-forward blocks, with layer normalization [6]. It has a 2048 hidden dimension, 8 (64-dim) heads for self- and cross-attention, and a hidden dimension of 1024 in the feed-forward MLP block. It operates on a single learnable 2048-dimensional SMPL query token as input and cross-attends to the 16×12 image tokens. Finally, a linear readout on the output token from the transformer decoder gives pose θ, shape β, and camera π.

C.2 Data details

In our training, we adopt the training data conventions of previous works [84], using images from Human3.6M [59], COCO [96], MPII [3] and MPI-INF-3DHP [108]. This forms the training set for the version we refer to as HMR 2.0a in the main manuscript. For the eventual HMR 2.0b version, we additionally generate pseudo-ground truth SMPL [100] fits for images from AVA [49], InstaVariety [69] and AI Challenger [171]. Since AVA and InstaVariety include videos, we collect frames by sampling at 1fps and 5fps respectively. For pseudo-ground truth generation, we use ViTDet [91] for bounding box detection and ViTPose [178] for keypoint detection, while fitting

happens using ProHMR [84]. We discard detections with very few 2D detected keypoints (less than five) and low detection confidence (threshold 0.5). We also discard fits with unnatural body shapes (*i.e.*, body shape parameters outside $[-3, 3]$), unnatural body poses (computed using a per-joint histogram of poses on AMASS [106]), and large fitting errors (*i.e.*, which indicates that the reconstruction was not successful). For training our HMR 2.0b model, we sample with different probabilities from each dataset, *i.e.*, Human3.6M: 0.1, MPII: 0.1, MPI-INF-3DHP: 0.1, AVA: 0.15, AI Challenger: 0.15, InstaVariety: 0.2, COCO: 0.2.

C.3 Training details

We train our main model using 8 A100 GPUs with an effective batch size of $8 \times 48 = 384$. We use an AdamW optimizer [102] with a learning rate of 1e-5, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a weight decay of 1e-4. Training lasts for 1M iterations, which takes roughly six days. For our main model HMR 2.0b, we train the network end-to-end. However, for the HMR 2.0a variant, the ViT encoder remains frozen, allowing a larger effective batch size of $8 \times 512 = 4096$, learning rate of 1e-4, and fewer training iterations of 100K (*i.e.*, roughly equivalent number of epochs).

C.4 Pose prediction

For the pose prediction model, we train a vanilla transformer model [164] from the tracklets obtained by [132]. Each tracklet at every time instance contains 3D pose and 3D location information, where the pose is parameterized by the SMPL model [100] and the location is represented as the translation in the camera frame. The transformer has 6 layers and 8 self-attention heads with a hidden dimension of 256. Each output token regresses the 3D pose and 3D location of the person at the specified time-step. We train this model by randomly masking input pose tokens and applying the loss on the masked tokens. During inference, to predict a future 3D pose, we query the model by reading out from a future time-step, using a learned mask-token as input to that time-step. Similarly for amodal completion, we replace the missing detections with the learned mask-token and read out from the output at the corresponding time-step. The model is trained with a batch size of 64 sequences and a sequence length of 128 tokens. We use the AdamW optimizer [102] with a learning rate of 0.001 and $\beta_1 = 0.9, \beta_2 = 0.95$.

C.5 Metrics

For our evaluation, we use the metrics that are common in the literature:
3D Pose: We follow [67] and we use MPJPE and PA-MPJPE. MPJPE refers to Mean Per Joint Position Error and it is the average L2 error across all joint, after aligning with the root node. PA-MPJPE is similar but it computed after aligning the predicted pose with the ground-truth pose using Procrustes Alignment.

2D Pose: We use PCK as defined in [181]. This is the Percentage of Correctly localized Keypoints, where a keypoint is considered as correctly localized if its L2 distance from the ground-truth keypoint is less than a threshold t. We report results using different thresholds (@0.05 and @0.1 of image size).
Tracking: Following [134, 133], we use standard tracking metrics. This includes ID switches (IDs), MOTA [74], IDF1 [138], and HOTA [104].
Action Recognition: We report results using mAP metrics as defined in the AVA dataset [49]. We further provided a more fine-grained analysis reporting results on different action categories: actions that involve Object Manipulation (OM), actions that involve Person Interactions (PI), and actions that involve Person Movement (PM). The results in these categories are also reported using mAP.

C.6 Tracking with PHALP$'$

In the main manuscript, we compare different human mesh recovery systems on the downstream problem of tracking (Table 3 of the main manuscript). For this, we modify the PHALP approach [133], so that pose distance is computed on the SMPL space that all the models share. To make this comparison fair, we keep other variables similar to the original PHALP (*e.g.*, same appearance embedding). Note that this comparison is generous to baselines that do not model appearance themselves. Eventually, our final 4DHumans system is using an updated appearance head and our new pose prediction, which lead to the state-of-the-art performance for tracking on PoseTrack (Table 4 of the main manuscript).

C.7 Action recognition

As an alternative way to assess the quality of 3D human reconstruction, we evaluate various human mesh recovery systems on the downstream task of action recognition on AVA (please refer to the attached anonymous manuscript [132] for more details on the task definition). More specifically, we take the tracklets from [132], which were generated by running PHALP [133] on the Kinetics [76] and AVA [49] datasets. Then, we replace the poses from various human mesh recovery models (*i.e.*, PyMAF [192], PyMAF-X [191], PARE [81], CLIFF [93], HMAR [133], HMR 2.0) and evaluate their performance on the action recognition task. In this pose-only setting, the action recognition model has access only to the 3D poses (in the SMPL format) and 3D location and is trained to predict the action of each person. For a fair comparison and to achieve the best performance for each 3D pose regressor, we retrain the action recognition model specifically for each 3D pose method.

C.8 Additional qualitative results

We have already provided a lot of qualitative results of HMR 2.0, both in the main manuscript and in the supplementary video. Here, we provide additional results, including comparisons with our closest competitors (Figure C.1), and a demonstration of our results in a variety of challenging cases, including successes (Figure C.2) and failure cases (Figure C.3).

Figure C.1: **Qualitative comparison of our approach with state-of-the-art methods.** We compare HMR 2.0 with our closest competitors, PyMAF-X [191], PARE [81] and CLIFF [93]. For each example, we show the input image, and results from each method (including the frontal and a side view). HMR 2.0 is significantly more robust in a variety of settings, including images with unusual poses, unusual viewpoints and heavy person-person overlap.

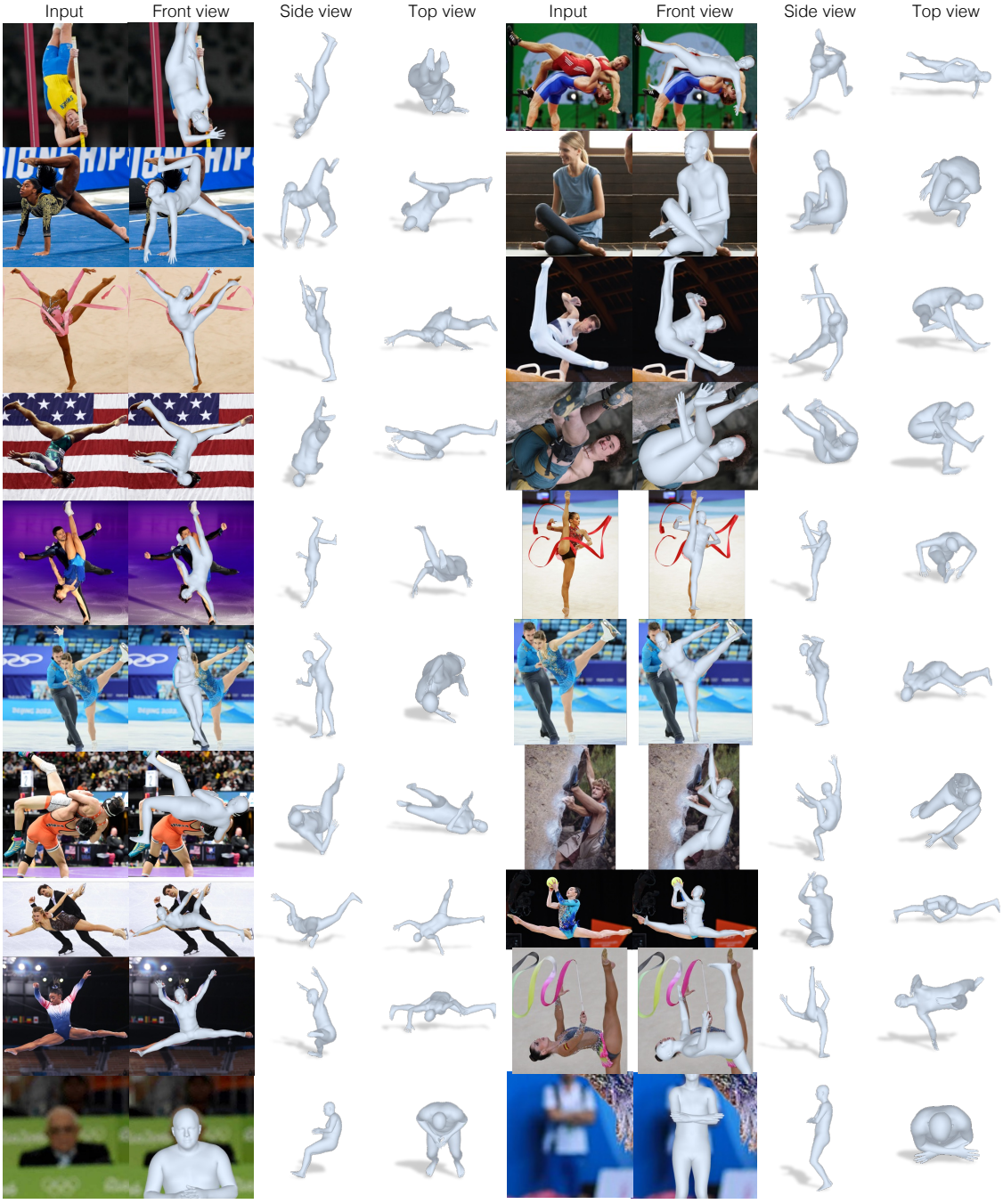

Figure C.2: **Qualitative results of our approach on challenging examples.** For each example we show the input image, the reconstruction overlay, a side view and the top view. The examples include unusual poses, unusual viewpoints, people in close interaction, extreme truncations and occlusions, as well as blurry images.

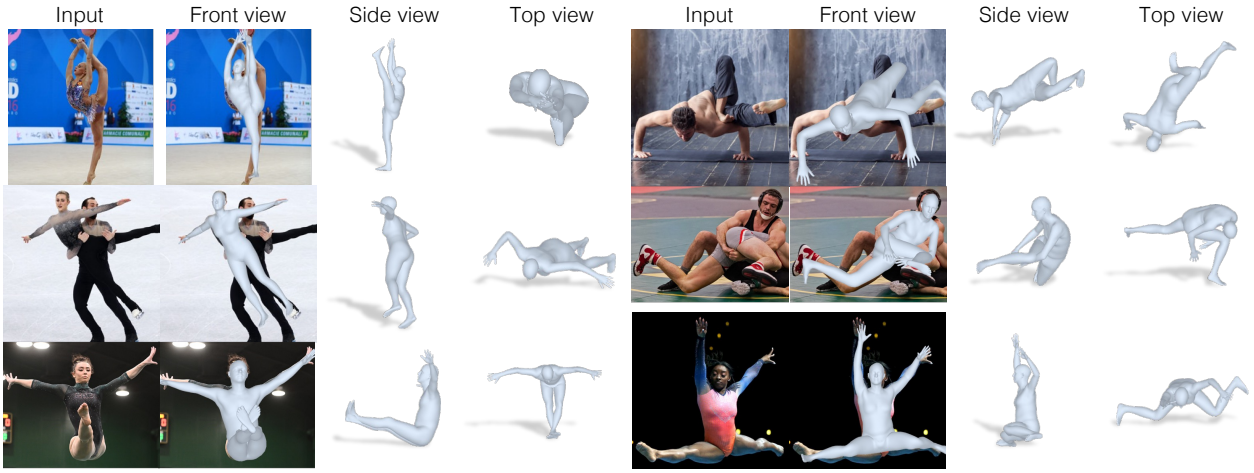

Figure C.3: **Failures of single frame 3D human reconstruction with HMR 2.0.** Despite the increased robustness of our method, we observe that HMR 2.0 occasionally recovers erroneous reconstructions in cases with very unusual articulation (first row), heavy person-person interaction (second row), and very challenging depth ordering for the different body parts (third row).