Unsupervised Learning of 3D Object Categories from Videos in the Wild

Philipp Henzler[1]* Jeremy Reizenstein[2] Patrick Labatut[2] Roman Shapovalov[2]
Tobias Ritschel[1] Andrea Vedaldi[2] David Novotny[2]

{reizenstein,plabatut,romansh,vedaldi,dnovotny}@fb.com
{p.henzler,t.ritschel}@cs.ucl.ac.uk

[1]University College London [2]Facebook AI Research

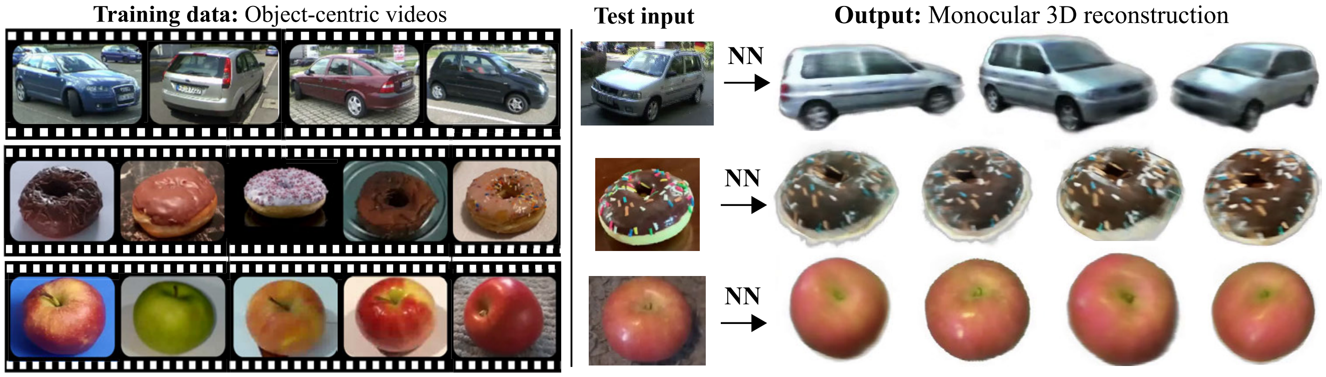
Training data: Object-centric videos
Test input
Output: Monocular 3D reconstruction
NN
NN
NN

Figure 1: We present a novel deep architecture that contributes *Warp-conditioned Ray Embedding (WCE)* to reconstruct and render new views (right) of object categories from one or few input images (middle). Our model is learned automatically from videos of the objects (left) and works on difficult real data where competitor architectures fail to produce good results.

Abstract

Our goal is to learn a deep network that, given a small number of images of an object of a given category, reconstructs it in 3D. While several recent works have obtained analogous results using synthetic data or assuming the availability of 2D primitives such as keypoints, we are interested in working with challenging real data and with no manual annotations. We thus focus on learning a model from multiple views of a large collection of object instances. We contribute with a new large dataset of object centric videos suitable for training and benchmarking this class of models. We show that existing techniques leveraging meshes, voxels, or implicit surfaces, which work well for reconstructing isolated objects, fail on this challenging data. Finally, we propose a new neural network design, called warp-conditioned ray embedding *(WCR), which significantly improves reconstruction while obtaining a detailed implicit representation of the object surface and texture, also compensating for the noise in the initial SfM reconstruction that bootstrapped the learning process. Our evaluation demonstrates performance improvements over several deep monocular reconstruction baselines on existing benchmarks and on our novel dataset. Project webpage:* `https://henzler.github.io/publication/unsupervised_videos/`.

1. Introduction

Understanding and reconstructing categories of 3D objects from 2D images remains an important open challenge in computer vision. Recently, there has been progress in using deep learning methods to do so but, due to the difficulty of the task, these methods still have significant limitations. In particular, early efforts focused on clean synthetic data such as ShapeNet [5], further simplifying the problem by assuming the availability of several images of each object instance,

[1] Work completed during an internship at Facebook AI Research.

knowledge of the object masks, object-centric viewpoints, etc. Methods such as [8, 7, 46, 55, 24] have demonstrated that, under these restrictive assumptions, it is possible to obtain high-quality reconstructions, motivating researchers to look beyond synthetic data.

Other methods have attempted to learn the 3D shape of object categories given a number of independent views of real-world objects, such as a collection of images of different birds. However, in order to simplify the task, most of them use some form of manual or automatic annotations of the 2D images. We seek to relax these assumptions, avoiding the use of manual 2D annotations or *a priori* constraints on the reconstructed shapes.

When it comes to high-quality general-purpose reconstructions, methods such as [36, 31, 33, 39, 65] have demonstrated that these can be obtained by training a deep neural network given only multiple views of a scene or object without manual annotations or particular assumptions on the 3D shape of the scene. Yet, these techniques can only learn a single object or scene at a time, whereas we are interested in modelling entire categories of 3D objects with related but different shapes, textures and reflectances. Nevertheless, the success of these methods motivates the use of multi-view supervision for learning collections of 3D objects.

In this paper, our first goal is thus to learn 3D object categories given as input multiple views of a large collection of different object instances. To the best of our knowledge, this is the first paper to conduct such a large-scale study of reconstruction approaches applied to learning 3D object categories from real-world 2D image data. Unfortunately, existing datasets for 3D category understanding are either small or synthetic. Thus, our first contribution is to introduce a new dataset of videos collected 'in the wild' by Mechanical Turkers (fig. 3). These videos capture a large number of object instances from the viewpoint of a moving camera, with an effect similar to a turntable. Viewpoint changes are estimated with high accuracy using off-the-shelf Structure from Motion (SfM) techniques. We collect hundreds of videos of several different categories.

Our second contribution is to assess current reconstruction technology on our new 'in the wild' data. For example, since each video provides several views of a single object with known camera parameters, it is suitable for an application of recent methods such as NeRF [36], and we find that learning *individual videos* works very well, as expected. However, we show that a direct application of such models to several videos of different but related objects is *much* harder. In fact, we experiment with related representations such as voxels and meshes, and find that they also do not work well if applied naïvely to this task. This is true even though reconstructions are focused on a single object at a time — thus disregarding the background — suggesting that these architectures have a difficult time at handling even relatively mild geometric variability.

Our final contribution is to propose a novel deep neural network architecture to better learn 3D object categories in such difficult conditions. We hypothesize that the main challenge in extending high-quality reconstruction techniques, that work well for single objects, to object categories is the difficulty of absorbing the geometric variability that comes in tackling many different objects together. An obvious but important source of variability is *viewpoint*: given only real images of different objects, it is not obvious how these should align in 3D space, and a lack of alignment adds to the variability that the model must cope with. We address this issue with a novel idea of *Warp-Conditioned Ray Embeddings (WCR)*, a new neural rendering approach that is far less sensitive to inaccurate 3D alignment in the input data. Our method modifies previous differentiable ray marchers to pool information at variable locations in input views, conditioned on the 3D location of reconstructed points.

With this, we are able to train deep neural networks that, given as input a small number of images of new object instances in a given target category, can reconstruct them in 3D, including generating high-quality new views of the objects. Compared to existing state-of-the-art reconstruction techniques, our method achieves better reconstruction quality in challenging datasets of real-world objects.

2. Related Work

Our work is related to many prior papers that leveraged deep learning for 3D reconstruction.

Learning synthetic 3D object categories. Early deep learning methods for 3D reconstruction focused on clean synthetic datasets such as ShapeNet [5]. Fully supervised methods [7, 12] mapped 2D images to 3D voxel grids. Follow-up methods proposed several alternatives: [8, 64] predict a point clouds, Park et al. [44, 1] label each 3D point with its signed distance to the nearest surface point, [35, 6] predict binary per-point occupancies, [11, 10] proposed more structured occupancy functions, and [13, 60] reconstruct meshes from single views. All aforementioned methods require full supervision in form of images and corresponding 3D CAD models. In contrast, our method requires only a set of videos of an object category captured from a moving camera.

Methods that avoid 3D supervision project 3D shapes to 2D images using differentiable rendering, allowing for image space optimization instead of 3D [46, 63, 56, 24, 22, 54].

Learning 3D object categories in the wild. Early reconstruction methods for 3D object categories used Non-Rigid SfM (NR-SfM) applied to 2D keypoint annotations [4, 58, 3]. CMR [23] used NR-SfM and 2D keypoints to initialize the camera poses on the CUB [59] dataset based on differentiable mesh rendering [26, 32, 6]. The texturing model of CMR was improved in DIB-R [6].

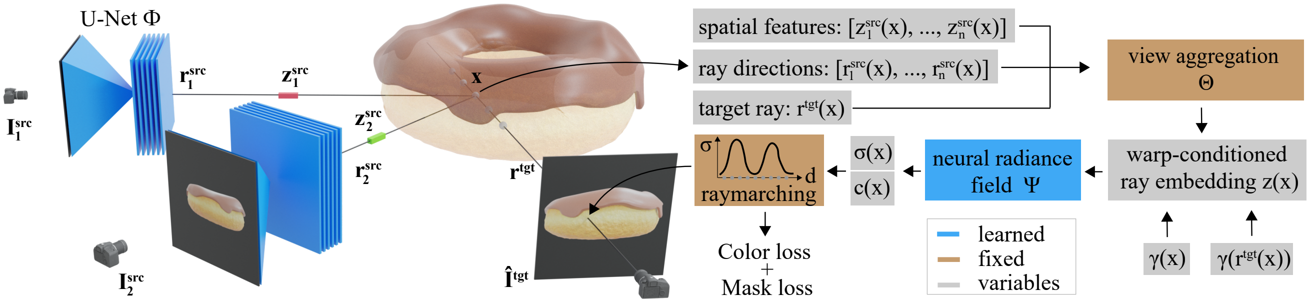

Figure 2: Our method takes as input an image and produces per pixel features using a U-Net Φ. We then shoot rays from a target view and retrieve per-pixel features from one or multiple source images. Once all spatial feature vectors are aggregated into a single feature vector (see Section 3.3 for more details), we combine them with their harmonic embeddings and pass them to an MLP yielding per location colors and opacities. Finally, we use differentiable raymarching to produce a rendered image.

Instead of assuming knowledge of pose, [28, 27, 14] assume a deformable 3D template. PlatonicGAN [19] enables template-free 3D reconstruction via differentiable emission-absorption raymarching, but requires knowledge of the camera-pose distribution.

Similarly, [62] does not require pose supervision, but it has been demonstrated only for limited viewpoint variations. Li et al. [29] do not assume camera poses as input, but use the self-supervised semantic features of [21] as a proxy for 2D keypoints as well as further constraints such as symmetry to help the reconstruction. We avoid such constraints for the sake of generality. Exploiting the StyleGAN [25] latent space, Zhang et al. [67] only require very few manual pose annotations. Our method, furthermore, does not require keypoint or pose supervision; instead, it recovers scene-specific camera poses automatically by analyzing camera motion. [42, 43] canonically align point clouds by only supervising with relative pose, but only learn a shape model.

Generative models trained in the wild were proposed in [9, 38]. While these methods can 'hallucinate' high-quality images, they, unlike us, are unable to also perform reconstruction of the objects given an image as input.

Implicit representation of 3D scenes. NeRF [36] has raised the interest in neural scene representation due to its high-quality output, inspired by positional encoding proposed in [57] and differentiable volume rendering from [19, 55]. NSVF [31] combined NeRF and voxel grids to improve the scalability and expressivity of the model whereas Yariv et al [65] uses sphere tracing to render signed distance fields. GRAF [51] extended NeRF to allow learning category-specific image generators, but do not perform reconstruction, which is our goal. Our method is inspired by NeRF, however, we learn a model of a whole object category, rather than a single scene or object.

Recent works, [20, 47, 48, 66, 61] utilize sampled per-pixel encodings similar to us. [66] averages features over multiple views and [61] learns to interpolate between views in an IBR fashion [18, 2] which prevents inpainting unseen areas. Our method aggregates latent encodings, which allows for representing unseen areas. Furthermore, we observed that simply averaging features from significantly different viewpoints, as done in [66], hurts performance. We thus propose to aggregate depending on view angles.

3. Method

Overview. The goal of our method is to learn a model of a 3D object category from a dataset $\{\mathcal{V}^p\}_{p=1}^{N_{\text{video}}}$ of video sequences. Each video $\mathcal{V}^p = (I_t^p)_{0 \le t < T^p}$ consists of $T^p \in \mathbb{N}$ color frames $I_t^p \in \mathbb{R}^{3 \times H \times W}$. While we do not use any manual annotations for the videos, we do pre-process them using a Structure-from-Motion algorithm (COLMAP [49]). In this manner, for each video frame I_t^p, we obtain sequence-specific camera poses $g_t^p \in SE(3)$ and the camera instrinsics $K_t^p \in \mathbb{R}^{3 \times 3}$. We further obtain a segmentation mask $m_t^p \in R^{1 \times H \times W}$ of the given category using Mask-RCNN [16].

The model parametrizes the appearance and geometry of the object in each video with an implicit surface map Ψ:

$$\Psi : \mathbb{R}^3 \times \mathbb{S}^2 \times \mathcal{Z} \to \mathbb{R}^3 \times \mathbb{R}_+ \quad \Psi(\mathbf{x}, \mathbf{r}, \mathbf{z}) = (\mathbf{c}, \sigma),$$

which labels each 3D scene point $\mathbf{x} \in \mathbb{R}^3$ and viewing direction $\mathbf{r} \in \mathbb{S}^2$ with an RGB triplet $\mathbf{c}(\mathbf{x}, \mathbf{r}, \mathbf{z}) \in \mathbb{R}^3$ and an occupancy value $\sigma(\mathbf{x}, \mathbf{z}) \in (0, 1]$ representing the opaqueness of the 3D space. Furthermore, the implicit function Ψ is conditioned on a latent code $\mathbf{z} \in \mathcal{Z}$ that captures the factors of variation of the object. By changing $\mathbf{z}$ we can adjust the occupancy field to represent shapes of different objects of a visual category. As described in section 3.3, the design of the latent space $\mathcal{Z}$ is crucial for the success of the method.

While we use video sequences to train the model, at test time we would like to reconstruct any new object instance from a small number of images. To this end, we learn an encoder function

$$\Phi : \mathbb{R}^{3 \times H \times W \times N_{\text{src}}} \to \mathcal{Z},$$

that takes a number of input *source* images $\{I_1^{\text{src}}, \ldots, I_{N_{\text{src}}}^{\text{src}}\}$ of the new instance and produces the latent code $\mathbf{z} \in \mathcal{Z}$.

Given a known target view (different view than the source images) we render the implicit surface to form a color image $\hat{I}^{\text{tgt}} \in \mathbb{R}^{3\times H\times W}$ and minimize the discrepancy between the rendered $\hat{I}^{\text{tgt}}$ and the masked ground truth image I^{tgt}.

In the following, we describe the main building blocks of our method. The rendering step follows Emission-Absorption raymarching [34, 19, 37, 54] as detailed in section 3.1. Section 3.2 describes the specifics of the surface function Ψ, and section 3.3 introduces the main technical contribution — a novel Warp-Conditioned Ray Embedding that defines the image encoder Φ.

3.1. Implicit surface rendering

In order to render a target image $\hat{I}^{\text{tgt}}$, we emmit a ray from the camera center through each pixel, assigning the color of ray's first 'intersection' with the surface to the respective pixel. Formally, let $\Omega = \{0,\dots,W-1\}\times\{0,\dots,H-1\}$ be an image grid, $u \in \Omega$ the index of a pixel, and $Z \in \mathbb{R}_+$ a depth value. Following the ray from the camera center through u to depth $Z \geq 0$ results in the 3D point: $\bar{\mathbf{x}}(u, Z) = Z \cdot K^{-1}[u^\top\ 1]^\top$, where $K \in \mathbb{R}^{3\times 3}$ are the camera intrinsics. The camera's pose is given by an Euclidean transformation $g^{\text{tgt}} \in SE(3)$, where we use the convention that $\bar{\mathbf{x}} = g^{\text{tgt}}(\mathbf{x})$ maps points $\mathbf{x}$ expressed in the world reference frame to points $\bar{\mathbf{x}}$ in camera coordinates.

In order to determine the color of a pixel $u \in \Omega$, we then 'shoot' a ray seeking the surface intersection. To do so, we sample points $\mathcal{X}_u = (\mathbf{x}(u, Z_i))_{i=0}^{N_Z+1}$ for depth values $Z_0 \leq \dots \leq Z_{N_Z}$ obtaining their colors and occupancies:

$$(\mathbf{c}_i, \sigma_i) = \Psi(\mathbf{x}(u, Z_i), \mathbf{r}, \mathbf{z}), \quad i = 0, \dots, N_Z. \qquad (1)$$

The probability of the ray *not* intersecting the surface in the interval $(Z_{i+1}, Z_i]$ is set to $T_i = e^{-(Z_{i+1}-Z_i)\sigma_i(\mathbf{x}(u,Z_i),\mathbf{z})}$ (transmission probability). Summing over all possible intersections $Z_0, \dots, Z_i$, the probability $p(Z = Z_i|u)$ of a ray terminating at depth Z_i is thus defined as:

$$p(Z = Z_i|u) = \left(\prod_{j=0}^{i-1} T_j\right)(1 - T_i)\,, \quad \hat{m}_u = 1 - \prod_{i=0}^{N_Z-1} T_i,$$

with the overall probability of intersection $\hat{m}_u$. Given the distributions of ray-termination probabilities $p(Z|u)$, the rendered color $\hat{\mathbf{c}}_u(\mathcal{X}_u, \mathbf{r}, \mathbf{z}) \in \mathbb{R}^3$ and opacity $\hat{\sigma}_u(\mathcal{X}_u, \mathbf{z}) \in \mathbb{R}$ are defined as an expectation over the outputs of the implicit function within the range $[0, \dots, N_Z - 1]$:

$$\hat{\mathbf{c}}_u = \sum_{i=0}^{N_Z-1} p(Z = Z_i|u)\mathbf{c}_i, \quad \hat{\sigma}_u = \sum_{i=0}^{N_Z-1} p(Z = Z_i|u)\sigma_i.$$

Since we are only interested in rendering the interior of the object, the colors $\mathbf{c}_u$ are softly-masked with $\hat{m}_u$ leading to the final target image render $\hat{I}^{\text{tgt}} \in \mathbb{R}^{3\times H\times W}$:

$$\hat{I}^{\text{tgt}} = I(g^{\text{tgt}}, \mathbf{z}) = \hat{m} \odot \hat{\mathbf{c}}. \qquad (2)$$

Note that the reconstruction depends on the target viewpoint g^{tgt} and the object code $\mathbf{z}$, which is viewpoint independent.

3.2. Neural implicit surface

Next, we detail the implicit surface function Ψ. Similar to previous methods [37, 40, 35], we exploit the representational power of deep neural networks and define Ψ as a deep multi-layer perceptron (MLP): $(\mathbf{c}, \sigma) = \Psi_{\text{nr}}(\mathbf{x}, \mathbf{r}, \mathbf{z})$. The network Ψ_{nr} follows a design similar to [37]. In particular, the world-coordinates $\mathbf{x}$ are preprocessed with the *harmonic encoding* $\gamma_{N_f^x}(\mathbf{x}) = [\sin(\mathbf{x}), \cos(\mathbf{x}), \dots, \sin(2^{N_f^x}\mathbf{x}), \cos(2^{N_f^x}\mathbf{x})] \in \mathbb{R}^{2N_f^x}$ before being input to the first layer of the MLP. In order to enable modelling of viewpoint dependent color variations, we further use the harmonic encoding of the target ray direction $\gamma_{N_f^r}(\mathbf{r}^{\text{tgt}}(\mathbf{x})) \in \mathbb{R}^{2N_f^r}$ as input (see Figure 2).

3.3. Warp-conditioned ray embedding

An important component of our method is the design of the latent code $\mathbf{z}$. A naïve solution is to first map a source image I^{src} to a D-dimensional vector $\mathbf{z}_{\text{CNN}} = \Psi_{\text{CNN}}(I^{\text{src}}) \in \mathbb{R}^D$ with a deep convolutional neural network Ψ_{CNN}, followed by appending a copy of $\mathbf{z}_{\text{CNN}}$ to each positional embedding $\gamma(\mathbf{x})$ to form an input to the neural occupancy function Ψ_{nr}. This approach, successfully utilized in [54, 32] for synthetic datasets where the training shapes are approximately rigidly aligned, is however insufficient when facing more challenging in-the-wild scenarios.

To show why there is an issue here, recall that our inputs are *videos* $\mathcal{V}^p$ of different object instances, each consisting of a sequence $(I_t^p)_{0\leq t<T^p}$ of video frames, together with viewpoint transformations $g_t^p \in SE(3)$ recovered by SfM. Crucially, due to the global coordinate frame and scaling ambiguity of the SfM reconstructions [15], there is no relationship between the camera positions g^p and g^q reconstructed for two different videos $p \neq q$. Even two identical videos $\mathcal{V}^p = \mathcal{V}^q$, reconstructed using SfM from two different random initializations, will result in two different sets of cameras $(g_t^p)_{0\leq t<T^p}$, $(g_t^q = g^\star g_t^p)_{0\leq t<T^p}$, related by an unknown similarity transformation $g^\star \in S(3)$. Since the frames $I_t^p = I_t^q$ are identical, the reconstruction network Φ_{CNN} must assign to them identical codes: $\mathbf{z}_{\text{CNN},t} = \mathbf{z}_{\text{CNN,t}}^p = \Phi_{\text{CNN}}(I_t^p) = \Phi_{\text{CNN}}(I_t^q) = \mathbf{z}_{\text{CNN,t}}^q$. Plugging this in eq. (2), means that two identical frames are reconstructed from the same code $\mathbf{z}_{\text{CNN},t}$ but two different viewpoints $g_t^p \neq g_t^q$: $\hat{I}_t^p = I(g_t^p, \mathbf{z}_{\text{CNN},t}) = I(g_t^q, \mathbf{z}_{\text{CNN},t}) = \hat{I}_t^q$. While of course we do not work with identical copies of the same videos, this extreme case demonstrates a fundamental issue with the naïve model, where different object instances must be reconstructed with respect to unrelated viewpoints.

We can partially tackle this issue by using a variant of [41] to approximately align the viewpoint of different video sequences before training (see supplemental).

Next, we introduce a more fundamental change to the model that also helps addressing this issue. The idea is to change the implicit surface (1)

$$\Psi_{\text{WCR}}(\mathbf{x}, \mathbf{z}(\mathbf{x})), \tag{3}$$

such that the code $\mathbf{z}$ is a *function of the queried ray point* $\mathbf{x}$ in world coordinates. Given a source image I_t^{src} with viewpoint g_t, the projection of this point in the image is: $u_t(\mathbf{x}) = \pi_t(\mathbf{x}) = \pi(K g_t \mathbf{x})$ where π denotes the perspective projection operator $\mathbb{R}^3 \to \Omega$. In particular, if $\mathbf{x}$ is also a point on the surface of the object, then $u_t(\mathbf{x})$ is the image of the corresponding point in the source view I_t^{src}.

More specifically, we task a convolutional neural network Φ to map the image I_t^{src} to a feature field $\Phi(I_t^{\text{src}}) \in \mathbb{R}^{D \times H \times W}$ (see supplementary for details). In this way, for each pixel u in the source view we obtain a corresponding embedding vector $\mathbf{z}[u]$ (using differentiable bilinear interpolation). Finally, we set:

$$\mathbf{z}(\mathbf{x}) = \mathbf{z}[u] = \Phi(I_t)[\pi_t(\mathbf{x})] \in \mathbb{R}^D, \tag{4}$$

and call it **Warp-Conditioned Ray Embedding (WCR)**.

As shown in fig. 2, by using eqs. (3) and (5) during ray marching, the implicit surface network Ψ can in this way pool information from relevant 2D locations in the source view I_t. Importantly, this occurs in a manner which is invariant to the global viewpoint ambiguity. In fact, if the geometry is now changed by the application of an arbitrary similarity transformation $g^\star$, then the 3D point changes as $\mathbf{x}' = g^\star \mathbf{x}$, but the viewpoint also changes as $g_t' = g_t(g^\star)^{-1}$, so that $g_t' \mathbf{x}' = g_t'(g^\star)^{-1} g^\star \mathbf{x} = g_t \mathbf{x}$ and the encoding of the points $\mathbf{x}$ and $\mathbf{x}'$ is the same: $\mathbf{z}[\pi_t(\mathbf{x})] = \mathbf{z}[\pi_t'(\mathbf{x})']$.

Finally, note that the network eq. (3) combines two sources of information: (1) codes $\mathbf{z}(\mathbf{x})$ that capture the appearance of each point in a manner which is invariant from the global coordinate transforms; and (2) the absolute location of the 3D point $\mathbf{x}$ (internally encoded by using position-sensitive coding $\gamma(\mathbf{x})$) which allows to to resolve misalignments by localizing the implicit surface equivariantly with changes of the global coordinates.

Multi-view aggregation. Having described WCR for a single source image we now extend to the more common case with multiple source images. For a set of source views $\{I_t^{\text{src}}\}_{t=1}^{N_{\text{src}}}$ with their warp-conditioned embeddings $\mathbf{z}_t^{\text{src}}(\mathbf{x})$, source rays $\mathbf{r}_t^{\text{src}}(\mathbf{x})$, and the target ray $\mathbf{r}^{\text{tgt}}(\mathbf{x})$ (see Figure 2), we calculate the aggregate WCR $\mathbf{z}(\mathbf{x}) \in \mathbb{R}^{D+1}$:

$$\mathbf{z}(\mathbf{x}) = \text{cat}(\mathbf{z}^\mu(\mathbf{x}), \mathbf{z}^\sigma(\mathbf{x}), \mathbf{z}_{\text{CNN}}), \tag{5}$$

as a concatenation (cat) of the angle-weighted mean and variance embedding $\mathbf{z}^\mu(\mathbf{x}) \in \mathbb{R}^D$ and $\mathbf{z}^\sigma(\mathbf{x}) \in \mathbb{R}_+$ respectively, and a plain average $\mathbf{z}_{\text{CNN}} = N_{\text{src}}^{-1} \sum_t \mathbf{z}_{\text{CNN},t}$ over global source embeddings $\mathbf{z}_{\text{CNN},t}$.

Here, the mean embedding $\mathbf{z}^\mu(\mathbf{x}) = \sum_{t=1}^{N_{\text{src}}} w_t(\mathbf{x}) \mathbf{z}_t^{src}(\mathbf{x})$ is a weighted average of the source embeddings $\mathbf{z}_t^{src}(\mathbf{x})$ with the weight $w_t(\mathbf{x})$ defined as

$$w_t(\mathbf{x}) = W(\mathbf{x})^{-1}(1 + \mathbf{r}_t^{\text{src}}(\mathbf{x}) \cdot \mathbf{r}^{\text{tgt}}(\mathbf{x})).$$

$W(\mathbf{x}) = \sum_{t=1}^{N_{\text{src}}} w_t(\mathbf{x})$ is a normalization constant ensuring the weights integrate to 1. This gives more weight to the source-view features that are imaged from a viewpoint which is closer to the target view. The variance embedding $\mathbf{z}^\sigma(\mathbf{x}) \in \mathbb{R}$ is defined in a similar manner as an average over dimension-specific $w_t(\mathbf{x})$-weighted standard deviations of the set of source embeddings $\{\mathbf{z}_t^{\text{src}}(\mathbf{x})\}_{t=1}^{N_{\text{src}}}$.

3.4. Overall learning objective

For training, we optimize the loss $\mathcal{L} = \lambda \mathcal{L}_{\text{mask}} + \mathcal{L}_{\text{rgb}}$ where $\lambda = 0.05$. $\mathcal{L}_{\text{mask}}$ is defined as the binary cross-entropy between the rendered opacity and ground truth mask. For the appearance loss $\mathcal{L}_{\text{rgb}}$ we use the mean-squared error between the masked target view and our rendering.

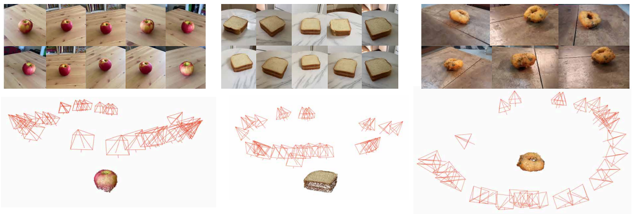

Figure 3: In order to study learning 3D object categories in the wild, we crowd-sourced a large collection of object-centric videos from Amazon Mechanical Turk. The top row shows frames from three example videos, the bottom two rows show SfM reconstructions of the videos together with tracked cameras.

4. Experiments

We discuss implementation details, data and evaluation protocols (section 4.1) and assess our method and baselines on the tasks of novel-view synthesis and depth prediction.

Implementation details. As noted in section 3.3, although WCR is in principle capable of dealing with the scene misalignments by itself, we found it beneficial to approximately "synchronize" the viewpoints of different videos in pre-processing, using a modified version of the method from [41]. First, we use the scene point clouds from SfM to register translation and scale by centering (subtracting the mean) and dividing by average per-dimension variance, resulting in adjusted viewpoints $\bar{g}_t$. We then proceed with training the rotation part of the viewpoint factorization branch of the VpDR network from [41], in order to align the rotational components of the viewpoints.

	AMT								**Freiburg Cars**							
	Train-test				**Test**				**Train-test**				**Test**			
Method	$\ell_1^{\textbf{RGB}}$	$\ell_1^{\textbf{VGG}}$	**IoU**	$\ell_1^{\textbf{Depth}}$	$\ell_1^{\textbf{RGB}}$	$\ell_1^{\textbf{VGG}}$	**IoU**	$\ell_1^{\textbf{Depth}}$	$\ell_1^{\textbf{RGB}}$	$\ell_1^{\textbf{VGG}}$	**IoU**	$\ell_1^{\textbf{Depth}}$	$\ell_1^{\textbf{RGB}}$	$\ell_1^{\textbf{VGG}}$	**IoU**	$\ell_1^{\textbf{Depth}}$
Mesh	0.10	1.17	0.60	5.13	0.10	1.16	0.60	5.09	0.14	2.03	0.60	1.19	0.17	2.17	0.56	**1.06**
Voxel	0.06	1.05	0.78	2.14	0.09	1.13	0.66	3.07	0.05	1.58	0.89	0.59	0.16	2.05	0.51	2.18
Voxel+MLP	0.06	1.04	0.78	1.95	0.09	1.13	0.65	2.87	0.05	1.47	0.88	**0.48**	0.16	2.06	0.54	1.97
MLP	0.04	0.90	0.87	1.38	0.09	1.13	0.65	3.59	**0.04**	**1.39**	0.87	0.59	0.15	2.03	0.47	2.52
Ours	**0.03**	**0.86**	**0.88**	**1.31**	**0.05**	**0.93**	**0.83**	**1.90**	**0.04**	**1.39**	**0.90**	**0.48**	**0.12**	**1.89**	**0.62**	1.60

Table 1: **Novel-view synthesis on AMT Objects and Freiburg Cars**. Each row evaluates either a baseline or **our** method. Results are reported for two perceptual metrics ℓ_1^{RGB}, ℓ_1^{VGG}, depth error ℓ_1^{Depth}, and intersection-over-union (IoU). For training we randomly selected between 1 and 7 source images. For testing we separately calculated the error metrics for 1, 3, 5 and 7 source images respectively and provide the average among those. For a more detailed evaluation we refer to the supplemental. Lower is better for $\ell_1^{\textbf{RGB}}$**,** $\ell_1^{\textbf{VGG}}$, and $\ell_1^{\textbf{Depth}}$, whereas higher is better for **IoU**. The best result is **bolded**.

4.1. AMT Objects and other benchmarks

One of our main contributions is to introduce the **AMT Objects** dataset, a large collection of object-centric videos that we collected (fig. 3) using Amazon Mechanical Turk. The dataset contains 7 object categories from the MS COCO classes [30]: apple, sandwich, orange, donut, banana, carrot and hydrant. For each class, we ask Turkers to collect a video by looking 'around' a class instance, resulting in a turntable video. For reconstruction, we uniformly sampled 100 frames from each video, discarding any video where COLMAP preprocessing was unsuccessful. The dataset contains 169-457 videos per class. For each class, we randomly split videos into training and testing videos in an 8:1 ratio.

We also consider the **Freiburg Cars** [52], consisting of 45 training and 5 testing videos of various parked cars.

For every video, we define three disjoint sets of frames on which we either train or evaluate: (1) *train-train*, (2) *train-test* and (3) *test*. For each training video, we form the train-test set by randomly selecting 16 frames and a disjoint train-train set containing the complement of train-test. While the train-train frames are utilized for training, the train-test frames are never seen during training and only serve for evaluation. The evaluation on the test set is the most challenging since it is conducted with views of previously unseen object instances.

Evaluation protocol. Recall that, at test time, our network takes as input a certain number of source images I^{src} and reconstructs a target image $\hat{I}^{\text{tgt}}$ seen from a different viewpoint. We assess the view synthesis and depth reconstruction quality of this prediction. To this end, for each object category, we randomly extract a batch of 8 different images from the *train-test* and *test* respectively. To increase view variability we repeat this process 5 times for every object. For each batch one of the images is picked as a target image I^{tgt} and from the remaining images we individually select 1,3,5,7 images and perform the forward pass to generate $\hat{I}^{\text{tgt}}$ for each selection.

In order to assess the quality of view synthesis, we calculate the $\ell_1^{\textbf{RGB}}$ error, between the target and predicted image. We also use the $\ell_1^{\textbf{VGG}}$ perceptual metric, which computes the ℓ_1 distance between the two images encoded by means of the VGG-19 network [53] pretrained on ImageNet. For depth reconstruction, we compute the $\ell_1^{\textbf{Depth}}$ distance between ground truth depth map (obtained from COLMAP SfM) and the predicted one in the target view. Finally we report Intersection-over-Union (**IoU**) between the predicted object mask and the object mask obtained by Mask-RCNN in the target view.

4.2. Baselines

In this section we detail the baselines we compare with. The first is **MLP**, corresponding to a naïve version of the latent global encoding $\mathbf{z}_{\text{CNN}}$ already discussed in section 3.3. Here, the N^{src} source images $\{I_t^{src}\}_{t=1}^{N^{\text{src}}}$ are first independently mapped to embedding vectors $\{\mathbf{z}_t \in \mathbb{R}^{256}\}_{t=1}^{N^{\text{src}}}$ by a ResNet50 [17] encoder and subsequently averaged to form an encoding of the object $\mathbf{z}_{\text{CNN}} = \frac{1}{N^{\text{src}}} \sum_{t=1}^{N^{\text{src}}} \mathbf{z}_t$. A copy of $\mathbf{z}_{\text{CNN}}$ is then concatenated to each positional embedding $\gamma(\mathbf{x})$ of each target ray point $\mathbf{x}$. MLP renders with the EA ray marcher (section 3.1).

The second baseline is **Voxel**, which closely resembles [55]. This uses the same encoding scheme as **MLP**, but differs by the fact that the object is represented by a voxel grid. Specifically, $\mathbf{z}_{\text{CNN}}$ is decoded with a series of 3D convolution-transpose layers to a 128^3 voxel grid containing RGB and opacity values. **Voxel** also renders with EA.

Next, **Voxel+MLP** is inspired by Neural Sparse Voxel fields [31] and marries NeRF [36] with voxel grids. As in **Voxel**, $\mathbf{z}_{\text{CNN}}$ is first 3D-deconvolved into a 128^3 volume of 32-dimensional features. Each target view ray point $\mathbf{x}$ is then described with a positional embedding $\gamma(\mathbf{x})$, and a latent feature $\mathbf{z}_{\text{g}}(\mathbf{x}) \in \mathbb{R}^{32}$ trilinearly sampled at the voxel grid location $\mathbf{x}$. The rest is the same as in **MLP**.

Finally, the **Mesh** baseline uses the soft-rasterization of [6] as implemented in PyTorch3D [45] with the top-k face

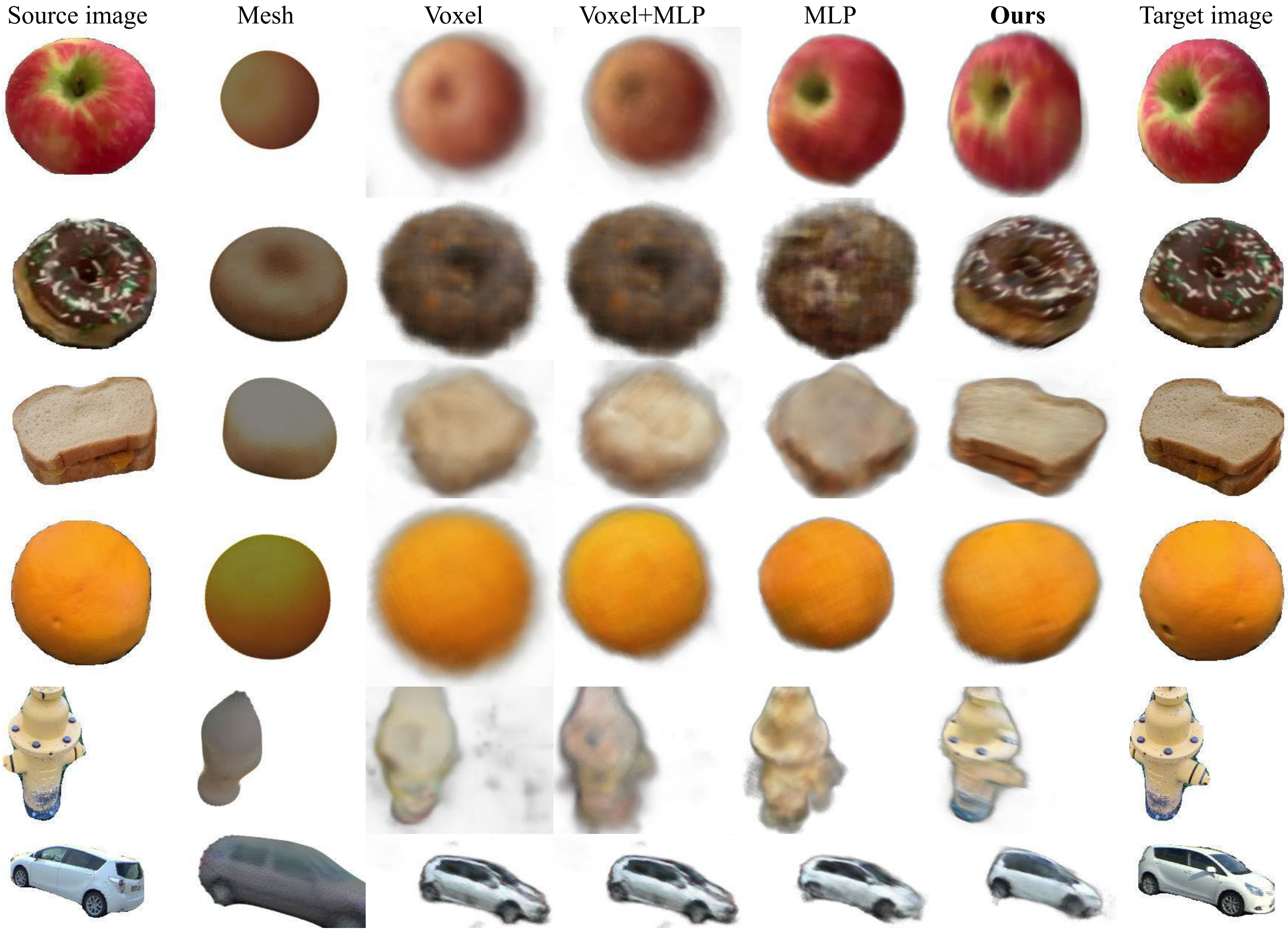

Figure 4: **Monocular reconstruction on Freiburg Cars and AMT Objects.** In reach row, a single source image (1st column) is processed by one of the evaluated methods (Mesh, Voxel, MLP+Voxel, MLP, **Ours** - columns 2 to 5) to generate a prescribed target view (last column). We show results on the test split.

		AMT								**Freiburg Cars**							
		Train-test				**Test**				**Train-test**				**Test**			
	Method	1	3	5	7	1	3	5	7	1	3	5	7	1	3	5	7
$\ell_1^{\mathbf{RGB}}$	Mesh	.096	.096	.096	.096	.102	.102	.102	.102	.141	.141	.140	.140	.166	.166	.166	**.166**
	Voxel	.062	.061	.061	.061	.091	.091	.091	.091	.055	.055	.055	.054	.159	.159	.158	.158
	Voxel+MLP	.059	.059	.058	.059	.090	.090	.090	.090	.045	.045	.045	.045	.158	.157	.158	.157
	MLP	**.037**	.036	.036	.036	.088	.088	.088	.088	**.041**	**.041**	**.041**	.041	.152	.152	.152	.152
	Ours	.038	**.032**	**.031**	**.030**	**.058**	**.046**	**.043**	**.042**	.046	**.041**	**.041**	**.040**	**.130**	**.120**	**.115**	**.114**

Table 2: . We evaluate the impact of increasing the number of source views during test time for the ℓ_1^{RGB} metric. Target renders and the corresponding metrics are produced for 1, 3, 5 and 7 source images. The best result is **bolded** where lower is better.

accumulation. The scene encoding $\mathbf{z}_{\text{CNN}}$ is converted with a pair of linear layers to: (1) a set $\{v_i(\mathbf{z}) \in \mathbb{R}^3\}_{i=1}^{N_{\text{vertex}}}$ of 3D vertex locations of the object mesh, and (2) a 128×128 UV map of the texture mapped to the surface of the mesh, which is rendered in order to evaluate the reconstruction losses from section 3.4. The mesh is initialized with an icosahedral sphere with 642 vertices.

4.3. Quantitative Results

Table 1 presents quantitative results on Freiburg Cars and the AMT Objects, respectively. In terms of all perceptual metrics ($\ell_1^{\mathbf{RGB}}$, $\ell_1^{\mathbf{VGG}}$) as well as depth and IoU, our method is on par with the MLP on the *train-test* split, and outperforms all other baselines on the *test* split of both datasets. This indi-

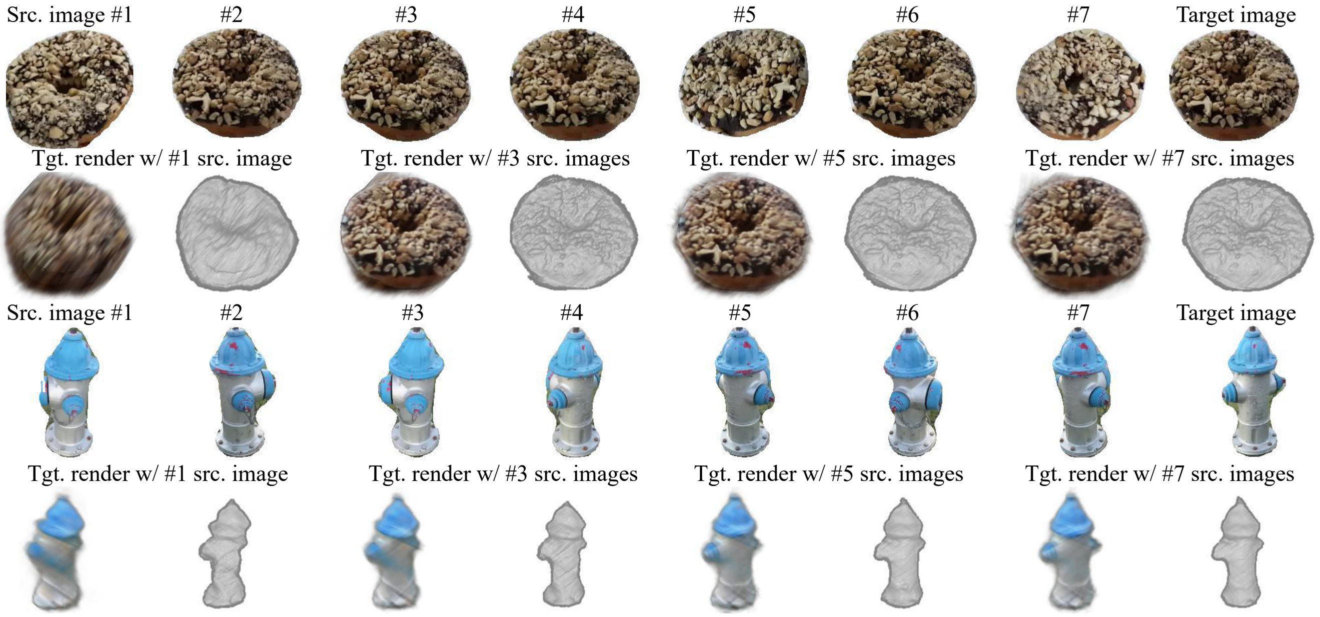
Src. image #1
#2
#3
#4
#5
#6
#7
Target image
Tgt. render w/ #1 src. image
Tgt. render w/ #3 src. images
Tgt. render w/ #5 src. images
Tgt. render w/ #7 src. images
Src. image #1
#2
#3
#4
#5
#6
#7
Target image
Tgt. render w/ #1 src. image
Tgt. render w/ #3 src. images
Tgt. render w/ #5 src. images
Tgt. render w/ #7 src. images

Figure 5: **Reconstruction with multiple source views.** The top row shows all available source images (top left) for a given target image (top right). The bottom row contains results conditioned on 1, 3, 5 or 7 source images. In addition to the rendered images we also provide shaded surface renderings.

cates significantly better ability of our warp-conditioned embedding to generalize to previously unseen object instances.

We further find that our method is better at leveraging multiple source views $N_{\text{src}} > 1$, outperforming all baselines for the ℓ_1^{RGB} error, see Table 2. When increasing the number of source images our method performance for all metrics improves whereas for all baselines it stays more or less constant. This further shows the effectiveness of the warp-conditioned embedding (WCR).

Regarding depth reconstruction, there is a fundamental issue with this evaluation. Recall that the target scene is rendered w.r.t. a corresponding camera estimated from SfM in pre-processing. SfM estimates a set of camera poses for source and target views that are internally consistent, but there is an overall scale ambiguity which cannot be fixed. When the network observes a single view, it has no way of establishing which scale was picked by SfM, which in turn means that, when the target camera matrix is plugged in the renderer, the result will be off because the assumed scale is incorrect. To that, we must also add the fact that ℓ_1^{Depth} is computed w.r.t. the COLMAP-MVS estimate of depth [50], which tends to be noisy. Unlike depth, the perceptual metrics are more reliable since color values captured by a camera are accurate. Note however, that our monocular results are still valid, since the viewpoints and scales have been approximately synchronized using the method from [42]. To conclude, we outperform other baselines on multi-source view perceptual metrics, which, as we argued above, are the most indicative.

4.4. Qualitative Results

Fig. 4 provide qualitative comparisons for monocular novel-view synthesis. It shows that our method produces significantly more detailed novel views, probably due to its ability to retrieve spatial encodings from the given source view. Fig. 5 further demonstrates the reconstruction improvement when multiple source views $N^{\text{src}} > 1$ are available.

5. Discussion

Limitations Even though our method demonstrates the capability to reconstruct 3D objects from only images, on a challenging in the wild dataset, our method still has several limitations. First, the differentiable rendering is very slow, making it impossible to use in real-time applications. Secondly, due to our template-free approach the object silhouettes tend to be blurry. Furthermore, our method is not always able to handle misalignments in the dataset, but nonetheless yields promising results on a noisy real world dataset. Lastly, despite no manual labeling necessary our method still relies on additional masks as input obtained from Mask-RCNN.

Conclusion In this paper, we have presented a method that is able to reconstruct category-specific 3D shape and appearance from videos of object categories in the wild alone, without requiring manual annotations. We demonstrated that our main contribution, warp-conditioned ray embedding, can successfully deal with the inherent ambiguities present in the video SfM reconstructions that provide our supervisory signal, outperforming alternatives on a novel dataset of crowd-sourced object videos.

References

[1] Matan Atzmon and Yaron Lipman. Sal: Sign agnostic learning of shapes from raw data. In *Proc. CVPR*, pages 2565–2574, 2020. 2

[2] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432, 2001. 3

[3] Joao Carreira, Abhishek Kar, Shubham Tulsiani, and Jitendra Malik. Virtual view networks for object reconstruction. In *Proc. CVPR*, 2015. 2

[4] Thomas J Cashman and Andrew W Fitzgibbon. What shape are dolphins? building 3d morphable models from 2d images. *PAMI*, 35(1):232–244, 2013. 2

[5] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv:1512.03012*, 2015. 1, 2

[6] Wenzheng Chen, Huan Ling, Jun Gao, Edward Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. In *Proc. NeurIPS*, pages 9609–9619, 2019. 2, 6, 12

[7] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Proc. ECCV*, pages 628–644. Springer, 2016. 2

[8] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proc. ICCV*, pages 605–613, 2017. 2

[9] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *Proc. 3DV*, pages 402–411. IEEE, 2017. 3

[10] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *Proc. CVPR*, pages 4857–4866, 2020. 2

[11] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Proc. ICCV*, pages 7154–7164, 2019. 2

[12] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *Proc. ECCV*. Springer, 2016. 2

[13] Georgia Gkioxari, Justin Johnson, and Jitendra Malik. Mesh R-CNN. In *Proc. ICCV*, 2019. 2

[14] Shubham Goel, Angjoo Kanazawa, and Jitendra Malik. Shape and viewpoint without keypoints. *Proc. ECCV*, 2020. 3

[15] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 4

[16] K. He, G. Gkioxari, and P. Dollár and. R. Girshick. Mask R-CNN. In *Proc. ICCV*, 2017. 3

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. 6

[18] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (SIGGRAPH Asia Conference Proceedings)*, 37(6), 2018. 3

[19] Philipp Henzler, Niloy Mitra, and Tobias Ritschel. Escaping plato's cave using adversarial training: 3d shape from unstructured 2d image collections. In *Proc. ICCV*, 2019. 3, 4

[20] Zeng Huang, Tianye Li, Weikai Chen, Yajie Zhao, Jun Xing, Chloe LeGendre, Linjie Luo, Chongyang Ma, and Hao Li. Deep volumetric video from very sparse multi-view performance capture. In *Proc. ECCV*, pages 336–354, 2018. 3

[21] Wei-Chih Hung, Varun Jampani, Sifei Liu, Pavlo Molchanov, Ming-Hsuan Yang, and Jan Kautz. Scops: Self-supervised co-part segmentation. In *Proc. CVPR*, 2019. 3

[22] Eldar Insafutdinov and Alexey Dosovitskiy. Unsupervised learning of shape and pose with differentiable point clouds. In *Proc. NeurIPS*, pages 2802–2812, 2018. 2

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

[24] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *Proc. NeurIPS*, 2017. 2

[25] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proc. CVPR*, 2019. 3

[26] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proc. CVPR*, 2018. 2

[27] Nilesh Kulkarni, Abhinav Gupta, David F. Fouhey, and Shubham Tulsiani. Articulation-aware canonical surface mapping. In *Proc. CVPR*, pages 449–458, 2020. 3

[28] Nilesh Kulkarni, Abhinav Gupta, and Shubham Tulsiani. Canonical surface mapping via geometric cycle consistency. In *Proc. ICCV*, 2019. 3

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

[30] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *Proc. ECCV*, 2014. 6

[31] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *Proc. NeurIPS*, 2020. 2, 3, 6

[32] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3D reasoning. *arXiv.cs*, abs/1904.01786, 2019. 2, 4

[33] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. *arXiv.cs*, abs/2008.02268, 2020. 2

[34] Nelson L. Max. Optical models for direct volume rendering. *IEEE Trans. Vis. Comput. Graph.*, 1(2):99–108, 1995. 4

[35] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proc. CVPR*, pages 4460–4470, 2019. 2, 4

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

[37] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proc. ECCV*, 2020. 4, 12

[38] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. HoloGAN: Unsupervised learning of 3D representations from natural images. *arXiv.cs*, abs/1904.01326, 2019. 3

[39] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. CVPR*, pages 3504–3515, 2020. 2

[40] Michael Niemeyer, Lars M. Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proc. ICCV*, 2019. 4

[41] David Novotný, Diane Larlus, and Andrea Vedaldi. Learning the semantic structure of objects from web supervision. In *Proceedings of the ECCV workshop on Geometry Meets Deep Learning*, 2016. 4, 5

[42] David Novotny, Diane Larlus, and Andrea Vedaldi. Learning 3d object categories by looking around them. In *Proc. ICCV*, 2017. 3, 8

[43] David Novotný, Diane Larlus, and Andrea Vedaldi. Capturing the geometry of object categories from video supervision. *PAMI*, 2018. 3

[44] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proc. CVPR*, pages 165–174, 2019. 2

[45] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 6

[46] Danilo Jimenez Rezende, SM Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. In *Proc. NeurIPS*, pages 4996–5004, 2016. 2

[47] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proc. ICCV*, October 2019. 3

[48] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In *Proc. CVPR*, June 2020. 3

[49] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proc. CVPR*, 2016. 3

[50] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proc. ECCV*, 2016. 8

[51] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Proc. NeurIPS*, 2020. 3

[52] Nima Sedaghat and Tomas Brox. Unsupevised generation of a viewpoint annotated car dataset from videos. In *Proc. ICCV*, 2015. 6

[53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. ICLR*, 2015. 6

[54] Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Multi-view consistency as supervisory signal for learning shape and pose prediction. In *Proc. CVPR*, pages 2897–2905, 2018. 2, 4

[55] Shubham Tulsiani, Tinghui Zhou, Alexei A. Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proc. CVPR*, pages 209–217, 2017. 2, 3, 6

[56] Shubham Tulsiani, Tinghui Zhou, Alexei A. Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proc. CVPR*, 2017. 2

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. NeurIPS*, 2017. 3

[58] Sara Vicente, João Carreira, Lourdes Agapito, and Jorge Batista. Reconstructing PASCAL VOC. In *Proc. CVPR*, 2014. 2

[59] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 2

[60] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proc. ECCV*, 2018. 2

[61] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. *arXiv*, 2021. 3

[62] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *Proc. CVPR*, 2020. 3

[63] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3D object reconstruction without 3D supervision. In *Proc. NIPS*, pages 1696–1704, 2016. 2

[64] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proc. ICCV*, pages 4541–4550, 2019. 2

[65] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Proc. NIPS*, 2020. 2, 3

[66] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. *arXiv*, 2020. 3

[67] Yuxuan Zhang, Wenzheng Chen, Huan Ling, Jun Gao, Yinan Zhang, Antonio Torralba, and Sanja Fidler. Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *Proc. ICLR*, 2021. 3

Unsupervised Learning of 3D Object Categories from Videos in the Wild

Supplementary material

A. Additional implementation details

In this section, we provide more detailed information about the dense image descriptors, as well as the view aggregation block Θ and the neural radiance field Ψ. Furthermore, we give more insights into the training process.

A.1. Dense image descriptors

This section describes in more detail the dense pixel-wise embeddings $\Phi(I_t)$ introduced in Section 3.3 in the main paper.

For a given source image I_t, the embedding $\mathbf{z}_t = \Phi(I_t)$ is composed of 3 different types of features: 1) learned 32-dimensional dense pixel-wise features $\Phi(I_t) \in \mathbb{R}^{3\times H\times W}$ output by a deep convolutional encoder network 2) raw image rgb colors $I_t \in \mathbb{R}^{3\times H\times W}$, and, 3) the segmentation mask $m_t \in \mathbb{R}^{1\times H\times W}$.

Dense feature extractor Φ. The architecture of the U-Net Φ is defined as follows (a detailed visualisation is present in Fig. 7). A source image $I^{src} \in \mathbb{R}^{3\times H\times W}$ is fed into a ResNet-50 which returns spatial features from intermediate convolutional layers (*layer1, layer2, layer3, layer4, layer5* and the final linear fully connected ResNet layer which outputs global features, i.e. non-spatial. Each spatial feature layer is then passed through a 1x1 convolution to equalize the size of all feature channels to 32, and bilinearly upsampled according to the spatial size of the source image. The resulting 6 different feature tensors of the same spatial size and dimension. The spatial feature encoding $\mathbf{e} = \Phi(I^{src}) \in \mathbb{R}^{(32\times 6)\times H\times W}$ is then formed as a concatenation of the 6 tensors.

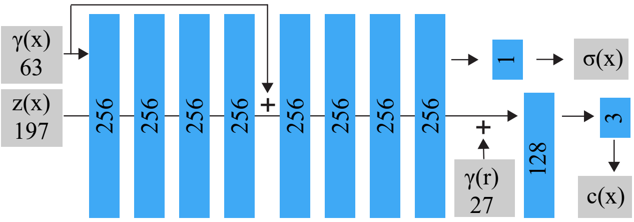

Figure 6: The neural radiance field Ψ is represented by an MLP. It takes as input the warp-conditioned embedding $z(x)$, the harmonic positional embedding $\gamma(x)$ and to account for view point variations the harmonic directional embedding $\gamma(\mathbf{r})$. It returns the rgb and opacity values.

Neural radiance field Ψ. Our scene is represented by a neural radiance field Ψ similar to [37] with the only difference that we additionally condition the field with our warp-conditioned embedding (wce), see Fig. 6.

A.2. Training details

We trained the both the U-Net encoder Φ and the neural radiance field Ψ architecture followed by differentiable rendering end-to-end using the Adam optimizer. We set the batch size to 8 and the learning rate to 1e-4. Our method as well as all baselines were trained on an NVIDIA Tesla V100 for 7 days. For all raymarching baselines and our method, we shoot 1024 rays per iteration through random image pixels in Monte-Carlo fashion. For each ray we first uniformly sample 128 times along the ray in order to retrieve a coarse rendering (voxel or mlp based depending on the method used). In the second pass we sample each ray 128 times based on probabilistic importance sampling similar to [37] (again voxel or mlp based depending on the method used).

For the mesh baseline we additionally used a negative IoU loss L_{iou}, a Laplacian loss L_{lap} and smoothness loss L_{sm} according to [6] and weighted them with 1.0, 19.0, 1.0 respectively.

B. Additional qualitative results

Additional qualitative results are available presented in Fig. 9 and Fig. 8. Also, we provide more qualitative results on our project webpage: `https://henzler.github.io/publication/unsupervised_videos/`. The page contains comparison of our method to baselines by showing the scenes from the train-test or test subsets rendered from a viewpoint that rotates around the object of interest.

C. Test-time view ablation

Furthermore, we also provide a view ablation of our method at test time. Recall that we randomly sample between 1 and 7 source images during training. During test time we evaluated our method separately on 1, 3, 5 and 7 views as input. In the main paper we provide an average of those numbers. In Table 2 we give insight into how changing the number of source views affects performance. Not surprisingly, increasing the numbers of source views consistently improves all metrics.

		AMT								Freiburg Cars							
		Train-test				Test				Train-test				Test			
	Method	1	3	5	7	1	3	5	7	1	3	5	7	1	3	5	7
ℓ_1^{VGG}	Mesh	1.163	1.167	1.168	1.169	1.160	1.161	1.163	1.163	2.030	2.029	2.028	2.023	2.170	2.168	2.166	2.167
	Voxel	1.052	1.051	1.051	1.051	1.127	1.127	1.127	1.127	1.581	1.581	1.580	1.580	2.050	2.050	2.046	2.046
	Voxel+MLP	1.041	1.040	1.040	1.040	1.131	1.130	1.130	1.130	1.469	1.468	1.468	1.468	2.067	2.063	2.063	2.064
	MLP	**0.900**	0.899	0.899	0.899	1.130	1.130	1.130	1.131	**1.391**	1.389	1.389	1.389	2.027	2.025	2.024	2.025
	Ours	0.905	**0.846**	**0.837**	**0.832**	**1.007**	**0.921**	**0.896**	**0.883**	1.450	**1.381**	**1.372**	**1.359**	**1.945**	**1.897**	**1.874**	**1.863**
IoU	Mesh	0.599	0.599	0.599	0.598	0.598	0.598	0.598	0.598	0.601	0.604	0.605	0.606	0.556	0.556	0.556	0.556
	Voxel	0.776	0.777	0.777	0.777	0.660	0.660	0.660	0.661	0.891	0.892	0.892	0.893	0.517	0.511	0.509	0.510
	Voxel+MLP	0.775	0.776	0.777	0.776	0.652	0.654	0.654	0.654	0.878	0.878	0.878	0.878	0.540	0.541	0.542	0.541
	MLP	**0.871**	0.871	0.872	0.872	0.654	0.653	0.653	0.653	0.872	0.872	0.872	0.872	0.472	0.470	0.472	0.471
	Ours	0.866	**0.884**	**0.886**	**0.889**	**0.774**	**0.788**	**0.787**	**0.787**	**0.889**	**0.897**	**0.898**	**0.897**	**0.600**	**0.624**	**0.629**	**0.632**
ℓ_1^{Depth}	Mesh	5.138	5.119	5.128	5.130	5.100	5.101	5.090	5.086	1.202	1.185	1.178	1.177	**1.062**	**1.061**	**1.063**	**1.063**
	Voxel	2.150	2.141	2.140	2.141	3.069	3.064	3.067	3.065	0.591	0.590	0.585	0.583	2.133	2.181	2.207	2.200
	Voxel+MLP	1.958	1.942	1.942	1.941	2.881	2.868	2.861	2.864	**0.478**	0.479	0.479	0.479	1.972	1.979	1.968	1.968
	MLP	**1.389**	1.378	1.377	1.377	3.583	3.587	3.590	3.593	0.595	0.593	0.594	0.593	2.521	2.530	2.519	2.520
	Ours	1.593	**1.291**	**1.201**	**1.172**	**2.186**	**1.847**	**1.802**	**1.776**	0.535	**0.467**	**0.457**	**0.453**	1.606	1.595	1.589	1.603

Table 3: We complement the evaluation in the main paper regarding the impact of increasing the number of source views during test time for the metrics: ℓ_1^{VGG}, ℓ_1^{Depth}, **IoU**. We compare the difference for 1, 3, 5 and 7 source images. The best result is **bolded** where lower is better for ℓ_1^{VGG}, ℓ_1^{Depth} and higher is better for **IoU**.

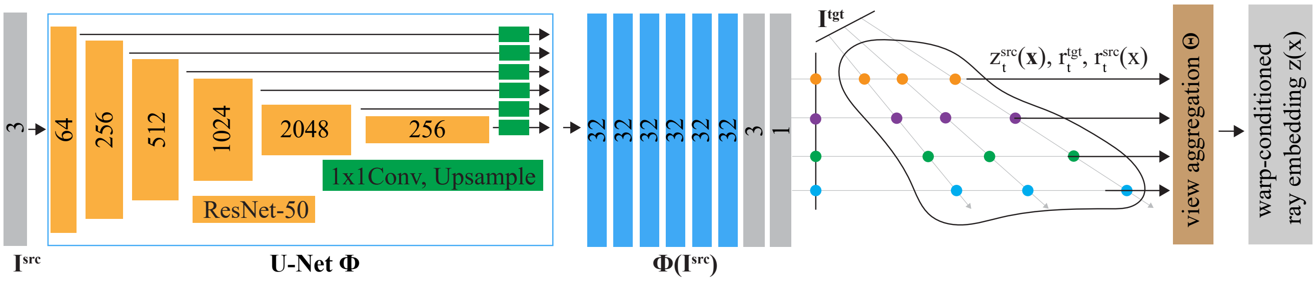

Figure 7: The input to the dense feature extractor Φ is a source image from a given view. It first makes use of a ResNet-50 to retrieve the layer-wise features. Then each layer is independently fed to a 1x1 convolution followed by bilinear upsmapling to the original input resolution. The resulting feature blocks are concatenated with the input image I^src and its corresponding object mask m^src. In case there are multiple source images available, this process is repeated for each of them. Once all per-view features are obtained the warp-conditioned embedding (wce) is retrieved after applying the view-aggregation Θ.

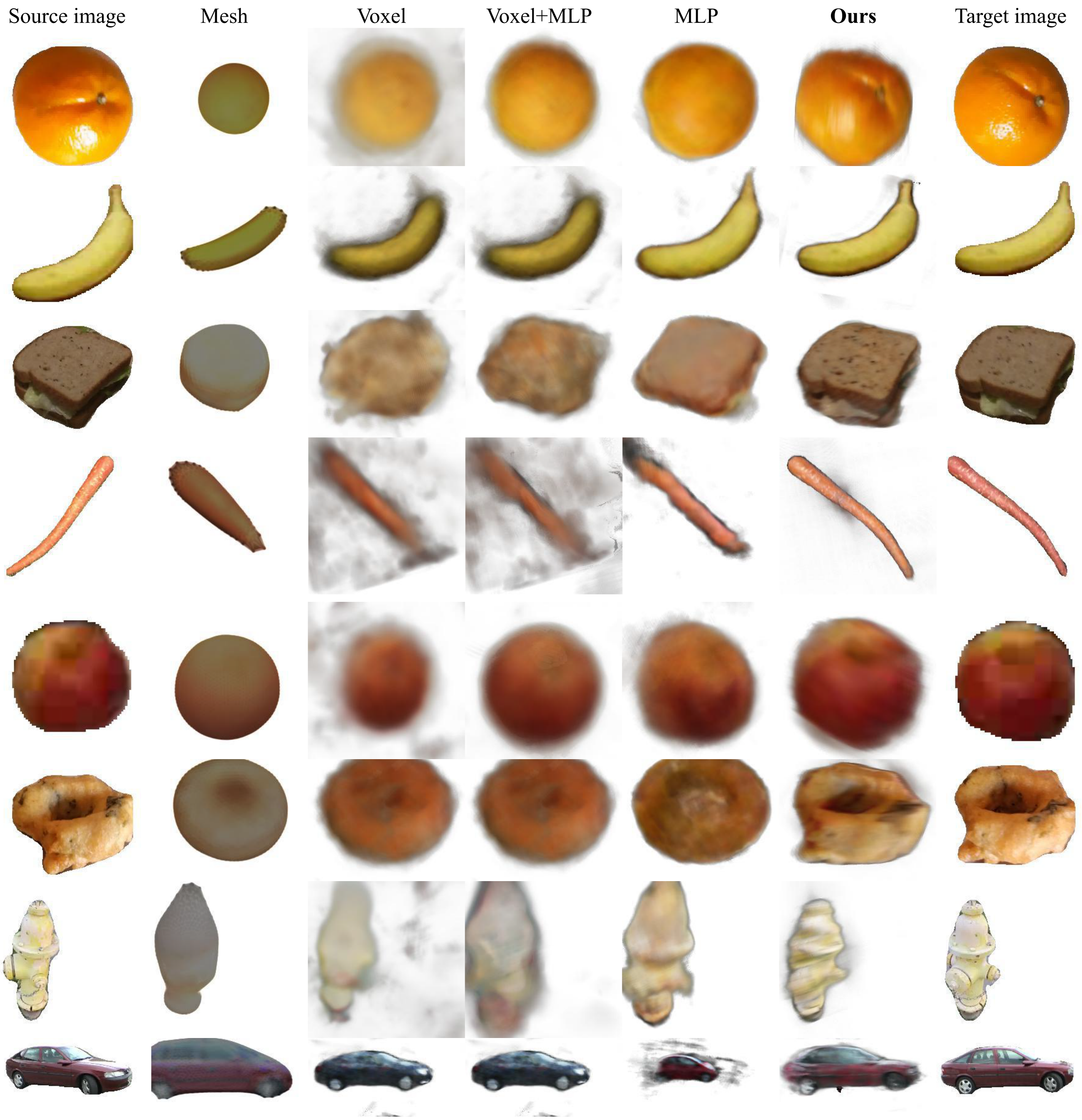

Figure 8: In reach row, a single source image (1st column) is processed by one of the evaluated methods (Mesh, Voxel, MLP+Voxel, MLP, **Ours** - columns 2 to 5) to generate a prescribed target view (last column). We show results on the test split.

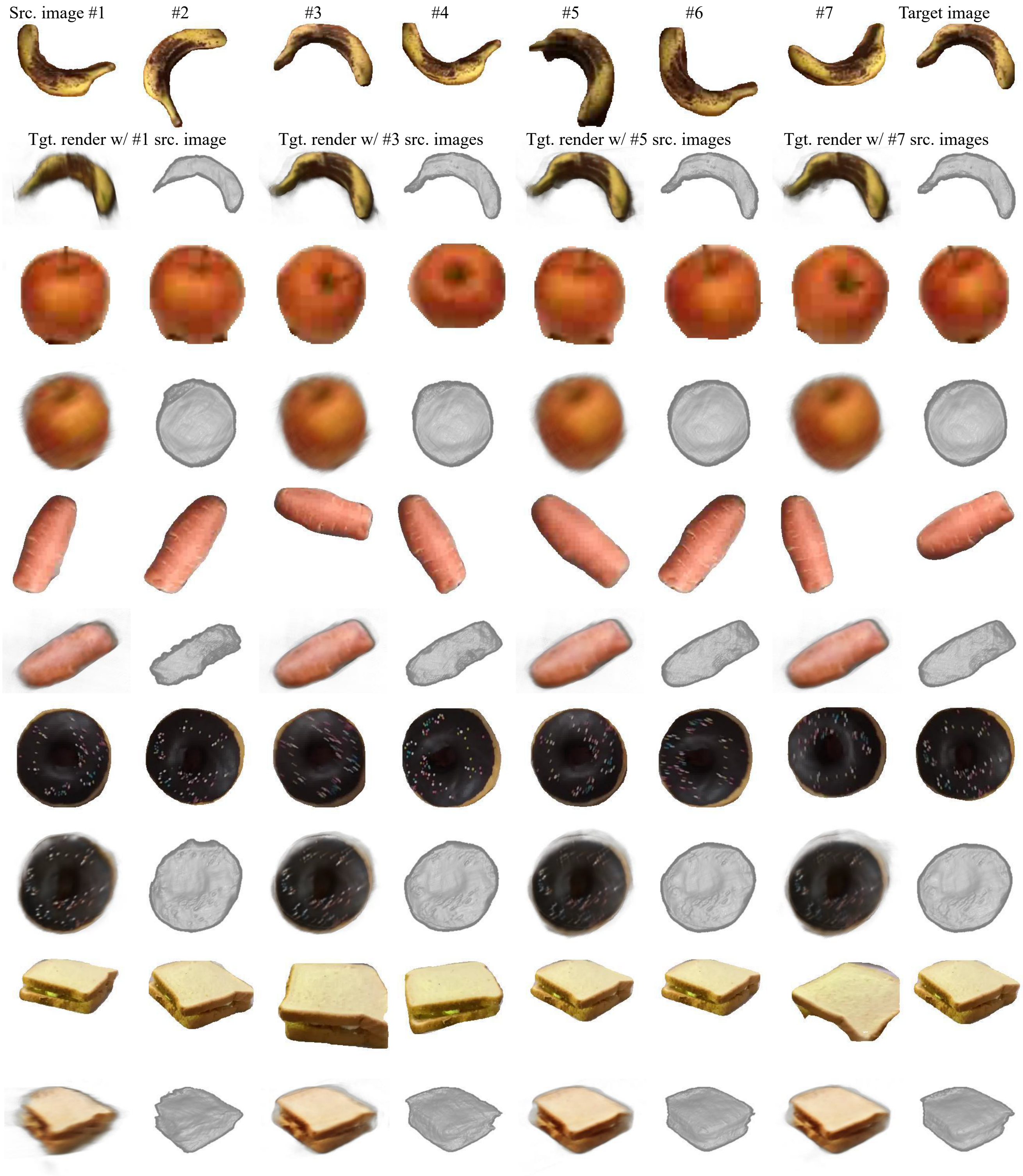
Src. image #1
#2
#3
#4
#5
#6
#7
Target image
Tgt. render w/ #1 src. image
Tgt. render w/ #3 src. images
Tgt. render w/ #5 src. images
Tgt. render w/ #7 src. images

Figure 9: **Reconstruction with multiple source views.** The top row for each object shows all available source images (top left) for a given target image (top right). The bottom row contains results conditioned on 1,3,5 or 7 source images. In addition to the volume rendered result we also provide shaded surface renderings.