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

Philipp Henzler1* Jeremy Reizenstein2 Patrick Labatut2 Roman Shapovalov2
Tobias Ritschel1 Andrea Vedaldi2 David Novotny2

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

1University College London 2Facebook AI Research

Training data: Object-centric videos

Test input

Output: Monocular 3D reconstruction

Figure 1: We present a novel deep architecture that contributes Warp-conditioned Ray Embedding (WCR) 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. For additional material please visit: 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,
knowledge of the object masks, object-centric viewpoints, etc. Methods such as [8, 7, 45, 54, 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, 38, 63] 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, 62] predict a point clouds, Park et al. [43, 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, 58] 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. [45, 61, 54, 24, 22, 53].

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, 56, 3], CMR [23] used NR-SfM and 2D keypoints to initialize the camera poses on the CUB [57] dataset based on differentiable mesh rendering [26, 32, 6]. The texturing model of CMR was improved in DIB-R [6].
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, [60] 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. [65] 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. [41, 42] 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, 37]. While these methods can ‘hallucinate’ high-quality images, they, unlike us, are unable to also perform reconstruction of the objects given an input image.

**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 [55] and differentiable volume rendering from [19, 54]. NSVF [31] combined NeRF and voxel grids to improve the scalability and expressivity of the model whereas Yariv et al. [63] uses sphere tracing to render signed distance fields. GRAF [50] 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, 46, 47, 64, 59] utilize sampled per-pixel encodings similar to us. [64] averages features over multiple views and [59] 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 [64], 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 \( \{I^p \}^n_{p=1} \) of video sequences. Each video \( I^p = \{I^p_t \}_{0 \leq t < T^p} \) consists of \( T^p \in \mathbb{N} \) color frames \( I^p_t \in \mathbb{R}^{3 \times H \times W} \). While we do not use any manual annotations for the videos, we do pre-process them using a Structure-from-Motion algorithm (COLMAP [48]). In this manner, for each video frame \( I^p_t \), we obtain sequence-specific camera poses \( g^p_t \in SE(3) \) and the camera intrinsics \( K^p_t \in \mathbb{R}^{3 \times 3} \). We further obtain a segmentation mask \( m^p_t \in \mathbb{R}^{1 \times H \times W} \) 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 \( \Psi : \mathbb{R}^3 \times \mathbb{S}^2 \times \mathcal{Z} \to \mathbb{R}^3 \times \mathbb{R}_+ \quad \Psi(x, r, z) = (c, \sigma) \),

which labels each 3D scene point \( x \in \mathbb{R}^3 \) and viewing direction \( r \in \mathbb{S}^2 \) with an RGB triplet \( c(x, r, z) \in \mathbb{R}^3 \) and an occupancy value \( \sigma(x, z) \in \{0, 1\} \) representing the opaqueness of the 3D space. Furthermore, the implicit function \( \Psi \) is conditioned on a latent code \( z \in \mathcal{Z} \) that captures the factors of variation of the object. By changing \( z \) 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 \( \{I^\text{src}_1, \ldots, I^\text{src}_n\} \) of the new instance and produces the latent code \( z \in \mathcal{Z} \).
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 \( I_{\text{tgt}} \).

In the following, we describe the main building blocks of our method. The rendering step follows Emission-Absorption raymarching \([34, 19, 36, 53]\) as detailed in section 3.1. Section 3.2 describes the specifics of the surface function \( \Psi \), and section 3.3 introduces the main technical contribution — a novel Warp-Conditioned Ray Embedding that defines the image encoder \( \Phi \).

### 3.1. Implicit surface rendering

In order to render a target image \( \hat{I}_{\text{tgt}} \), we emit 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 \( \Omega = \{0, \ldots, W - 1\} \times \{0, \ldots, H - 1\} \) 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 \( \hat{x}(u, Z) = Z \cdot K^{-1}[u^T 1]^T \), where \( K \in \mathbb{R}^{3 \times 3} \) are the camera intrinsics. The camera’s pose is given by an Euclidean transformation \( \hat{x} = g^\text{tgt}(x) \) maps points \( x \) expressed in the world reference frame to points \( \hat{x} \) 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 \( X_u = (x(u, Z_i))_{i=0}^{N_Z+1} \) for depth values \( Z_0 \leq \cdots \leq Z_{N_Z} \) obtaining their colors and occupancies:

\[
(c_i, \sigma_i) = \Psi(x(u, Z_i), r, z), \quad i = 0, \ldots, N_Z.
\]

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

\[
p(Z = Z_i | u) = \prod_{j=0}^{i-1} T_j (1 - T_i), \quad \bar{m}_u = 1 - \sum_{i=0}^{N_Z-1} T_i,
\]

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

\[
\hat{c}_u = \sum_{i=0}^{N_Z-1} p(Z = Z_i | u) c_i, \quad \bar{\sigma}_u = \sum_{i=0}^{N_Z-1} p(Z = Z_i | u) \sigma_i.
\]

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

\[
\hat{I}_{\text{tgt}} = I(g^\text{tgt}, z) = \hat{m} \odot \hat{c}.
\]

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

### 3.2. Neural implicit surface

Next, we detail the implicit surface function \( \Psi \). Similar to previous methods \([36, 39, 35]\), we exploit the representational power of deep neural networks and define \( \Psi \) as a deep multi-layer perceptron (MLP): \( (c, \sigma) = \Psi_w(x, r, z) \). The network \( \Psi_w \) follows a design similar to \([36]\). In particular, the world-coordinates \( x \) are preprocessed with the harmonic encoding \( \gamma_{N_f}(x) = [\sin(x), \cos(x), \ldots, \sin(2^{N_f} x), \cos(2^{N_f} x)] \in \mathbb{R}^{2N_f} \) 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 \( \gamma_{N_f}(x) \in \mathbb{R}^{2N_f} \) as input (see Figure 2).

### 3.3. Warp-conditioned ray embedding

An important component of our method is the design of the latent code \( z \). A naive solution is to first map a source image \( I^\text{src} \) to a \( D \)-dimensional vector \( z_{\text{CNN}} = \Phi_{\text{CNN}}(I^\text{src}) \in \mathbb{R}^D \) with a deep convolutional neural network \( \Phi_{\text{CNN}} \), followed by appending a copy of \( z_{\text{CNN}} \) to each positional embedding \( \gamma(x) \) to form an input to the neural occupancy function \( \Psi_w \). This approach, successfully utilized in \([53, 32]\) 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 \( V^p \) of different object instances, each consisting of a sequence \( (I^p_t)_{0 \leq t \leq T^p} \) of video frames, together with viewpoint transformations \( g^p \in SE(3) \) 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 \( V^p = V^q \), reconstructed using SfM from two different random initializations, will result in two different sets of cameras \( (g^p_t)_{0 \leq t \leq T^p}, (g^q_t)_{0 \leq t \leq T^q} \), related by an unknown similarity transformation \( g^* \in S(3) \).

Since the frames \( I^p_t = I^q_t \) are identical, the reconstruction network \( \Phi_{\text{CNN}} \) must assign to them identical codes: \( z_{\text{CNN},t} = \Phi_{\text{CNN}}(I^p_t) = \Phi_{\text{CNN}}(I^q_t) = z^t_{\text{CNN},t} \). Plugging this in eq. (2), means that two identical frames are reconstructed from the same code \( z_{\text{CNN},t} \) but two different viewpoints \( g^p_t \neq g^q_t ; \hat{I}^p_t = I(g^p_t, z_{\text{CNN},t}) = I(g^q_t, z_{\text{CNN},t}) = I^q_t \).

While of course we do not work with identical copies of the same videos, this extreme case demonstrates a fundamental issue with the naive model, where different object instances must be reconstructed with respect to unrelated viewpoints.

We can partially tackle this issue by using a variant of \([40]\) 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 embedding $z$ such that the code $z$ is a function of the queried ray point $x$ in world coordinates. Given a source image $I_{t}^{src}$ with viewpoint $g_t$, the projection of this point in the image is: $u_t(x) = \pi_t(x) = \pi(Kg_t x)$ where $\pi$ denotes the perspective projection operator $\mathbb{R}^3 \rightarrow \Omega$. In particular, if $x$ is also a point on the surface of the object, then $u_t(x)$ is the image of the corresponding point in the source view $I_{t}^{src}$.

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

$$z_t(x) = \Phi(I_{t})[\pi_t(x)] \in \mathbb{R}^{D},$$

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

Intuitively, as shown in fig. 2, by using eqs. (3) and (4) during ray marching, the implicit surface network $Ψ_{\text{WCR}}$ can pool information from relevant 2D locations $u_t$ in the source view $I_{t}^{src}$. 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 $g^*$, then the 3D point changes as $x' = g^* x$, but the viewpoint also changes as $g'_t = g_t(g^*)^{-1}$, so that $g'_t x' = g'_t (g^*)^{-1} g^* x = g_t x$ and the encoding of the points $x$ and $x'$ is the same: $\Phi(I_{t})[\pi_t(x)] = \Phi(I_{t})[\pi'_t(x')]$

Finally, note that the network eq. (3) combines two sources of information: (1) codes $z(x)$ 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 $x$ (internally encoded by using position-sensitive coding $\gamma(x)$). The combination of 1) and 2) above allows 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 $\{I_{t}^{src}\}_{t=1}^{N_{src}}$ with their warp-conditioned embeddings $z_{t}^{src}(x)$, source rays $r_{t}^{src}(x)$, and the target ray $r_{tgt}(x)$ (see Figure 2), we calculate the aggregate WCR $z(x, \{I_{t}^{src}\})$:

$$z(x, \{I_{t}^{src}\}) = \text{cat}(z^\mu(x, \{I_{t}^{src}\}), z^\sigma(x, \{I_{t}^{src}\}), z_{\text{CNN}}(\{I_{t}^{src}\})),$$

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

The mean $z^\mu(x, \{I_{t}^{src}\}) = \sum_{t=1}^{N_{src}} w_t(x) z_{t}^{src}(x)$ is a weighted average of the source embeddings $z_{t}^{src}(x)$ with the weight $w_t(x)$ defined as

$$w_t(x) = \frac{1}{\bar{w}(t)} \cdot \frac{\gamma(x)}{\bar{\gamma}}(1 + r_{t}^{src}(x) \cdot r_{tgt}(x)).$$

$\bar{w}(t)$ 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 $z^\sigma$ is defined analogously as an average over dimension-specific $w_t(x)$-weighted standard deviations of the source embedding set $\{z_{t}^{src}(x)\}_{t=1}^{N_{src}}$.

### 3.4. Overall learning objective

For training, we optimize the loss $L = \lambda L_{\text{mask}} + L_{\text{rgb}}$ where $\lambda = 0.05$. $L_{\text{mask}}$ is defined as the binary cross-entropy between the rendered opacity and ground truth mask. For the appearance loss $L_{\text{rgb}}$ 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.](image-url)

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 [40]. 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 $\hat{g}_t$. We then proceed with training the rotation part of the viewpoint factorization branch of the VpDR network from [40], in order to align the rotational components of the viewpoints.
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 pre-processing 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** [51], 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 \(I^{tgt}\) 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 \(I^{tgt}\) for each selection.

In order to assess the quality of view synthesis, we calculate the \(\ell^{RGB}_1\) error, between the target and predicted image. We also use the \(\ell^{VGG}_1\) perceptual metric, which computes the \(\ell_1\) distance between the two images encoded by means of the VGG-19 network [52] pretrained on ImageNet. For depth reconstruction, we compute the \(\ell^{Depth}_1\) 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.

| Method     | \(\ell^{RGB}_1\) | \(\ell^{VGG}_1\) | IoU | \(\ell^{Depth}_1\) | \(\ell^{RGB}_1\) | \(\ell^{VGG}_1\) | IoU | \(\ell^{Depth}_1\) |
|------------|-----------------|-----------------|-----|-------------------|-----------------|-----------------|-----|-------------------|
| Mesh       | 0.10            | 1.17            | 0.60| 5.13              | 0.10            | 1.16            | 0.60| 5.09              |
| Voxel      | 0.06            | 1.05            | 0.78| 2.14              | 0.09            | 1.13            | 0.66| 3.07              |
| Voxel+MLP  | 0.06            | 1.04            | 0.78| 1.95              | 0.09            | 1.13            | 0.65| 2.87              |
| MLP        | 0.04            | 0.90            | 0.87| 1.38              | 0.09            | 1.13            | 0.65| 3.59              |
| Ours       | **0.03**        | **0.86**        | **0.88**| **1.31**        | **0.05**        | **0.93**        | **0.83**| **1.90**        |

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 \(\ell^{RGB}_1\), \(\ell^{VGG}_1\), depth error \(\ell^{Depth}_1\), 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 \(\ell^{RGB}_1\), \(\ell^{VGG}_1\), and \(\ell^{Depth}_1\), whereas higher is better for IoU. The best result is bolded.
4.3. Quantitative Results

Table 1 presents quantitative results on Freiburg Cars and the AMT Objects, respectively. In terms of all perceptual metrics ($\ell_{\text{RGB}}^1$, $\ell_{\text{VGG}}^1$) as well as depth and IoU, our method is on par with the MLP on the train-test split. On the test split, we outperform all other baselines in $\ell_{\text{RGB}}^1$, mean absolute error (MAE), and mean squared error (MSE) for Freiburg Cars. For the AMT Objects, our method outperforms all baselines in terms of $\ell_{\text{RGB}}^1$, $\ell_{\text{VGG}}^1$, and IoU.

4.4. Qualitative Results

Figure 4: Monocular reconstruction on Freiburg Cars and AMT Objects. In each row, a single source image (1st column) is processed by one of the evaluated methods (Mesh, Voxel, MLP+Voxel, MLP, Ours - columns 2 to 6) to generate a prescribed target view (last column). We show results on the test split.

Table 2: We evaluate the impact of increasing the number of source views during test time for the $\ell_{\text{RGB}}^1$ 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.

Source image | Mesh | Voxel | Voxel+MLP | MLP | Ours | Target image
---|---|---|---|---|---|---
[Image of source images]

Figure 4: Monocular reconstruction on Freiburg Cars and AMT Objects. In each row, a single source image (1st column) is processed by one of the evaluated methods (Mesh, Voxel, MLP+Voxel, MLP, Ours - columns 2 to 6) to generate a prescribed target view (last column). We show results on the test split.

Table 2: We evaluate the impact of increasing the number of source views during test time for the $\ell_{\text{RGB}}^1$ 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.

Table 2: We evaluate the impact of increasing the number of source views during test time for the $\ell_{\text{RGB}}^1$ 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.

4.3. Quantitative Results

Table 1 presents quantitative results on Freiburg Cars and the AMT Objects, respectively. In terms of all perceptual metrics ($\ell_{\text{RGB}}^1$, $\ell_{\text{VGG}}^1$) as well as depth and IoU, our method is on par with the MLP on the train-test split. On the test split, we outperform all other baselines in $\ell_{\text{RGB}}^1$.
\[ \ell^\text{VGG}_1 \] and IoU on all 7 classes of AMT Objects and Freiburg Cars. This indicates 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 \( N_{\text{src}} \geq 1 \), outperforming all baselines for the \( \ell^\text{RGB}_1 \) 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 \( \ell^\text{Depth}_1 \), our method outperforms all alternatives on all datasets except the test split of Freiburg Cars, where we are 2nd after Mesh. Here, we note that \( \ell^\text{Depth}_1 \) is only an approximate measure because: 1) the predicted depth is compared to the COLMAP-MVS estimate of depth [49], which tends to be noisy and; 2) the scale ambiguity in SfM reconstructions that supervise learning leads to a significantly unconstrained problem of estimating the scale of a testing scene given a small number of source views, which is challenging to resolve for any method.

### 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 \( N_{\text{src}} \geq 1 \) are available.

### 5. Discussion and conclusions

#### Limitations.

Even though our method outperforms baselines on the vast majority of metrics and datasets, there are still several limitations. First, the execution of the deep MLP at every 3D ray-location in a rendered frame is relatively slow (depending on the number of source views rendering takes between 3 and 8 sec for a \( 128 \times 256 \) image on average), which makes a real-time deployment challenging. Secondly, due to our template-free approach, the object silhouettes can be blurry. Lastly, despite no manual labeling is necessary, our method still relies on segmentation masks that were automatically generated with Mask-RCNN.

#### Conclusions.

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. Future work could include decomposition of shape, appearance and lighting allowing for more control over the rendered images.
References

[1] Matan Atzmon and Yaron Lipman. Sal: Sign agnostic learning of shapes from raw data. In Proc. CVPR, 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, 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

[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, Forrest Cole, Aneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In Proc. CVPR, pages 4857–4866, 2020. 2

[11] Kyle Genova, Forrest 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 Trans Graph (Proc. SIGGRAPH Asia), 2018. 3

[19] Philipp Henzl, 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, Chloé 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, 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 Thomas Funkhouser. Unconstrained Photo Collections. In Proc. 3DV, 2019. 2

[32] Zeng Huang, Tianye Li, Weikai Chen, Yajie Zhao, Jun Xing, and Jian Sun. Deep residual learning for image recognition. In Proc. ICCV, 2019. 2

[33] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In Proc. ICCV, 2019. 2, 4

[34] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In Proc. CVPR, 2021. 2

[35] Nelson L. Max. Optical models for direct volume rendering. IEEE Trans. Vis. Comput. Graph., 1995. 4

[36] 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

[37] Ben Mildenhall, Pratul P Sinha, 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, 4, 6, 11
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 Φ as well as 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 Φ(It) introduced in Section 3.3 in the main paper.

For a given source image It, the embedding field Φ(It) is composed of 3 different types of features: 1) learned 5 · 32-dimensional dense pixel-wise features output by a deep convolutional encoder network ΦU-Net, 2) raw image rgb colors It ∈ R^3×H×W, and 3) the segmentation mask mt ∈ R^1×H×W.

Dense feature extractor ΦU-Net. The architecture of the U-Net inside ΦU-Net is defined as follows (a detailed visualisation is present in Fig. 7). A source image It ∈ R^3×H×W, masked by m^w (retrieved from Mask-RCNN), is fed into a ResNet-50 which returns spatial features from intermediate convolutional layers (layer1, layer2, layer3, layer4, layer5), and the final linear ResNet layer which outputs global features zCNN, i.e. non-spatial. Each feature layer including the global one is then passed through a 1x1 convolution to equalize the size of all feature channels to 32. The spatial features are further bilinearly upsampled to the spatial size of the source image and concatenated along the channel dimension to create a dense embedding field ΦU-Net(It) ∈ R^5·32×H×W.

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 γ(x) and to account for view point variations the harmonic directional embedding γ(r). It returns the rgb and opacity values.

Neural radiance field Ψ. Our scene is represented by a neural radiance field Ψ similar to [36] with the only difference that we additionally condition the field with our warp-conditioned ray embedding, see Fig. 6.

A.2. Training details

We trained both the U-Net encoder ΦU-Net and the neural radiance field Ψ with 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 following [36].

For the mesh baseline we shoot rays for each pixel per iteration and use soft rasterization to predict the surface intersection. In addition to the losses used for the other baselines as well as our method, we additionally use a negative IoU loss L_iou, a Laplacian loss L_lap and smoothness loss L_sm according to [44] 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 3 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.
Table 3: We complement the evaluation of the impact of the number of source views during test time for the metrics: $\ell^1_{VGG}$, $\ell^1_{Depth}$, IoU. We report results for 1, 3, 5 and 7 source images. The best result is bolded where lower is better for $\ell^1_{VGG}$, $\ell^1_{Depth}$ and higher is better for IoU.

Figure 7: The input to the dense feature extractor $\Phi$ is a source image from a given view. It first makes use of a ResNet-50 ($\Phi_{U-Net}$) to retrieve the layer-wise features. Then, each layer is independently fed to a 1x1 convolution followed by bilinear upsampling 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 ray embedding is retrieved after applying the view-aggregation.
Figure 8: In each row, a single source image (1st column) is processed by one of the evaluated methods (Mesh, Voxel, MLP+Voxel, MLP, **Ours** - columns 2 to 6) to generate a prescribed target view (last column). We show results on the test split.
Figure 9: **Reconstruction with multiple source views.** The top row for each object shows all available source images (columns 1-7) 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 new RGB views we also provide shaded surface renderings.