# KeyTr: Keypoint Transporter for 3D Reconstruction of Deformable Objects in Videos

David Novotny dnovotny@fb.com
Ignacio Rocco irocco@fb.com
Samarth Sinha sinhasam@fb.com
Alexandre Carlier alexandre.carlier01@gmail.com
Gael Kerchenbaum gael.kerchenbaum@gmail.com
Roman Shapovalov romansh@fb.com
Nikita Smetanin nikitasmetanin@fb.com
Natalia Neverova nneverova@fb.com
Benjamin Graham benjamingraham@fb.com
Andrea Vedaldi vedaldi@fb.com

Meta AI

## Abstract

*We consider the problem of reconstructing the depth of dynamic objects from videos. Recent progress in dynamic video depth prediction has focused on improving the output of monocular depth estimators by means of multi-view constraints while imposing little to no restrictions on the deformation of the dynamic parts of the scene. However, the theory of Non-Rigid Structure from Motion prescribes to constrain the deformations for 3D reconstruction. We thus propose a new model that departs significantly from this prior work. The idea is to fit a dynamic point cloud to the video data using Sinkhorn's algorithm to associate the 3D points to 2D pixels and use a differentiable point renderer to ensure the compatibility of the 3D deformations with the measured optical flow. In this manner, our algorithm, called Keypoint Transporter, models the overall deformation of the object within the entire video, so it can constrain the reconstruction correspondingly. Compared to weaker deformation models, this significantly reduces the reconstruction ambiguity and, for dynamic objects, allows Keypoint Transporter to obtain reconstructions of the quality superior or at least comparable to prior approaches while being much faster and reliant on a pre-trained monocular depth estimator network. To assess the method, we evaluate on new datasets of synthetic videos depicting dynamic humans and animals with ground-truth depth. We also show qualitative results on crowd-sourced real-world videos of pets.*

## 1. Introduction

We are interested in the problem of reconstructing 3D dynamic scenes from casually recorded videos. A scene is *dynamic* if it contains moving objects, including deforming ones such as people or animals. This is a very challenging reconstruction scenario which has traditionally been addressed by making use of specialized hardware, such as multi-camera domes or 3D scanners. However, with advancements in virtual and augmented reality, we can envisage a future in which non-experts may wish to create content to experience in 3D. In these scenarios, dynamic

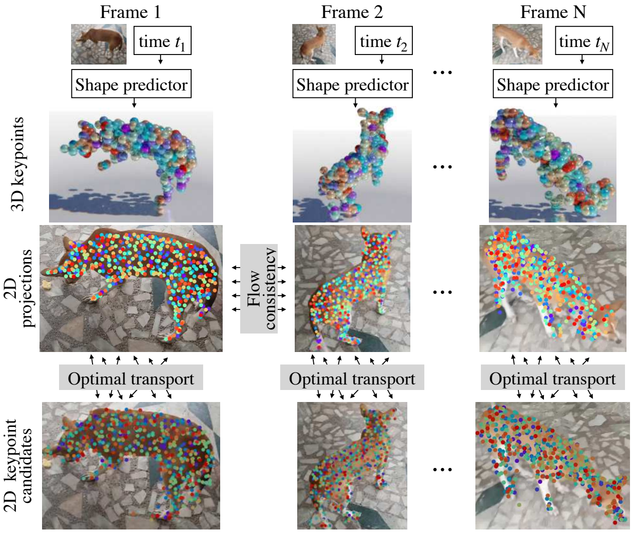

Figure 1. **Keypoint Transporter** reconstructs the depth of 3D non-rigid objects from a casually recorded video. Unlike prior work, we model the deformations of the object globally from the beginning to the end of the video. The key technical contribution is a robust mechanism to track these long-range deformations. This is based on estimating a dynamic cloud of 3D keypoints that are (1) encouraged to optimally cover a set of candidate 2D points in every frame via differentiable optimal transport and (2) to describe a 2D trajectory compatible with the measured optical flow.

3D scenes must be reconstructed from limited observations, such as a monocular video captured by a phone camera.

Casual 3D reconstruction is much more challenging than reconstruction in a controlled capture setup. While the problem has many interesting aspects, including reconstructing the shape and appearance of the visible parts of the scene and extrapolating the parts that are not visible (for new-view synthesis), in this work we focus on the task of reconstructing depth from videos. Furthermore, we focus specifically on reconstructing the depth of dynamic objects which deform over time as these are often the focus of attention, while being challenging to reconstruct.

Several works have recently considered the problem of estimating depth in casual videos of dynamic scenes [20,29, 49]. The general approach is to first apply a deep network such as MiDaS [23] in order to obtain a per-frame depth estimation. Given this initial, and often unreliable, depth estimate, principles from multi-view geometry are then applied to refine the solution, leveraging the information contained in the whole video. The latter usually starts by estimating image correspondences by an off-the-shelf optical flow method such as RAFT [44]. The simplest approach, adopted by Consistent Video Depth (CVD) [29] assumes dynamic objects behave rigidly across neighbouring video frames, which limits its applicability to slowly moving objects. The follow up Robust CVD [20] does not apply geometric constraints to the dynamic objects. Other approaches such as Dynamic Video Depth (DVD) [49] explicitly estimate and constrain the 3D deformation of the dynamic parts of the scene by means of a small neural network.

While prior works have settled on variants of the pipeline discussed above, in this paper we take a step back and question their assumptions. Since the problem is to reconstruct the 3D shape of a non-rigid object, we re-consider Non-Rigid Structure from Motion (NRSfM) methods [4, 5, 11, 33]. The key lesson form NRSfM is that reconstructing a deformable object is possible only if the *space of deformations* is *sufficiently* constrained. The simplest of such constraints is to assume that the 3D deformations span a low-rank linear subspace. By comparison, (Robust) CVD do not model non-rigid deformations explicitly, and DVD only enforces local smoothness of the 3D deformation field.

We thus propose *Keypoint Transporter (KeyTr)*: built on the idea of constraining the deformations that the object undergoes *throughout* the video. This is a significant departure from recent works because it requires to track deformations across the entire video (not just instantaneously as in, *e.g.*, DVD). In contrast to traditional NRSfM, which utilizes 2D feature trackers, KeyTr works by maintaining a set of 3D keypoints that can be deformed within a low-rank subspace. The keypoints and deformations are learned so that: (1) the 2D projections of the deformed keypoints cover the object region well in each frame, and (2) their 2D trajectories are compatible with the measured optical flow. The latter constraint is enforced in a differentiable and occlusion-aware manner by using an off-the-shelf point cloud renderer.

Compared to prior methods that model only local deformations, we empirically show that globally modelling and constraining object deformations significantly reduces the reconstruction ambiguity. Because of this, we show that KeyTr obtains superior or at least comparable reconstruction quality *without using a pre-trained depth estimation model* such as MiDaS; instead, our model reconstructs videos individually from scratch without any prior learning.

We also introduce several new datasets for measuring the quality of dynamic video depth algorithms. We quantitatively evaluate on synthetic datasets of humans and animals and we further evaluate the reconstructions qualitatively, on real videos of pets collected for the study.

## 2. Related work

**Non-rigid Structure from Motion.** NRSfM simultaneously estimates the viewpoints and 3D structure of a dynamic scene. This is naturally modelled by matrix factorization [6]. Subsequent research has focused on reducing the ambiguity incurred in decomposing viewpoint and object deformation by restricting the rank of the deformation space [2, 10, 11, 51] or of the 3D point trajectories [3, 4]. Other works enforced smoothness in the spatio-temporal domain [1, 14, 21, 22], sparse [50] or Gaussian [46] priors, or minimized canonicalization loss [33]. In this spirit, we also constrain the space of deformations to be low-rank.

**Learning deformable meshes.** Several authors considered learning 3D shape predictors specialized to individual object categories. For instance, CMR [17] and DIB-R [7] learn to reconstruct a mesh of genus-0 topology from a collection of category-specific images supervised with sparse 2D keypoints. UMR [25] prescinds from keypoint supervision by leveraging an unsupervised parts detector trained on a large dataset. There is also a large amount of work on reconstructing important categories such as humans.

More relevant to this work are approaches that learn to 'overfit' a neural network to a single video: Kokkinos and Kokkinos [19] use Laplacian mesh editing of a known category-level template, while LASR [48] learns a parametric skinning model. Our method is equally general to LASR and, while it does not obtain a full reconstructed mesh, is more robust.

**Dynamic new-view synthesis.** Also relevant to our reconstruction problem are works on dynamic video new-view synthesis (NVS). Neural Volumes [27] recover a time-dependent voxelized reconstruction of a dynamic object by learning codes and decoders. Inspired by neural radiance fields (NeRF) [31], which replace voxels with a continuous representation of shape and appearance, several works

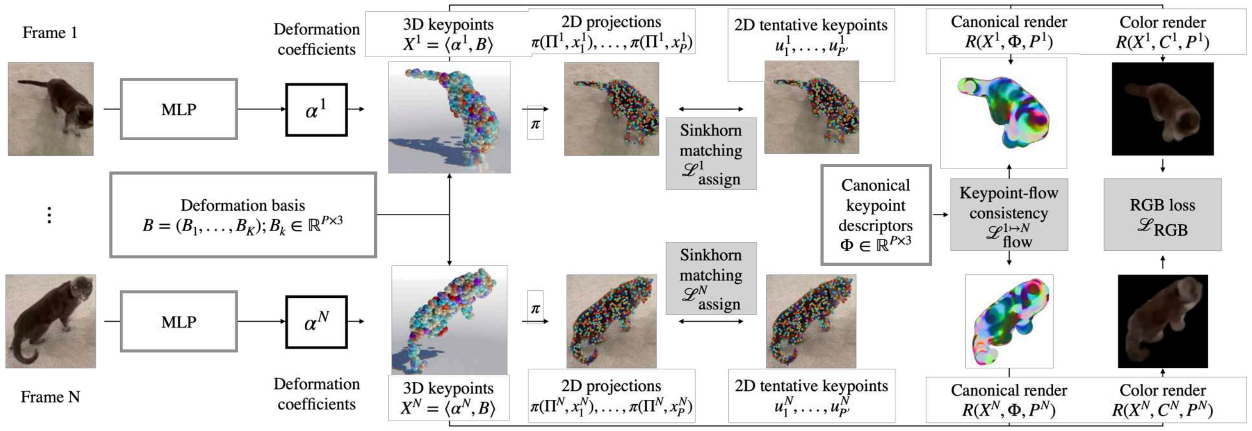
Frame 1
MLP
Deformation coefficients
$\alpha^1$
3D keypoints
$X^1 = \langle \alpha^1, B \rangle$
2D projections
$\pi(\Pi^1, x_1^1), \ldots, \pi(\Pi^1, x_P^1)$
$\pi$
Sinkhorn matching $\mathscr{L}^1_{\text{assign}}$
2D tentative keypoints
$u_1^1, \ldots, u_{P'}^1$
Canonical render
$R(X^1, \Phi, P^1)$
Color render
$R(X^1, C^1, P^1)$
⋮
Deformation basis
$B = (B_1, \ldots, B_K); B_k \in \mathbb{R}^{P \times 3}$
Canonical keypoint descriptors
$\Phi \in \mathbb{R}^{P \times 3}$
Keypoint-flow consistency
$\mathscr{L}^{1 \mapsto N}_{\text{flow}}$
RGB loss
$\mathscr{L}_{\text{RGB}}$
Sinkhorn matching $\mathscr{L}^N_{\text{assign}}$
Frame N
MLP
$\alpha^N$
$\pi$
Deformation coefficients
3D keypoints
$X^N = \langle \alpha^N, B \rangle$
2D projections
$\pi(\Pi^N, x_1^N), \ldots, \pi(\Pi^N, x_P^N)$
2D tentative keypoints
$u_1^N, \ldots, u_{P'}^N$
Canonical render
$R(X^N, \Phi, P^N)$
Color render
$R(X^N, C^N, P^N)$

Figure 2. **An overview of Keypoint Transporter.** We represent the object shape in the $i$-th frame $I^i$ as a cloud of keypoints $X^i$ whose deformations span a low-rank linear deformation basis $B$. We minimize $\mathcal{L}^i_{\text{assign}}$ to obtain the assignment between the 2D keypoint projections $\pi(\Pi^i, x_p)$ and a set of candidate 2D candidate locations $u^i_{p'}$ sampled from each image. Furthermore, $\mathcal{L}^i_{\text{flow}}$ encourages keypoints to track unique locations on the object surface. Finally, we also optimize a complementary color reconstruction loss $\mathcal{L}^i_{\text{RGB}}$.

proposed to train radiance fields on videos of dynamic scenes. These generally encode time using a positional encoder [26, 36] or otherwise [34, 47] and use a neural network for warping 3D coordinates back to a canonical (time-invariant) reconstruction of the scene or object. However, "cancelling" non-rigid deformations in this manner is ambiguous, so these methods regularize the learned deformation field assuming elasticity [34] or penalizing the field's divergence [47]. In practice, they manage to reconstruct well only the videos that render limited motions, such as changing facial expressions in the case of Nerfies [34].

**Dynamic video depth prediction.** More directly related to our work are methods that estimate depth from videos. The simplest approach is to apply a monocular depth estimator such as Monodepth2 [12] to every frame. These monocular estimators are often learned in a self-supervised manner, e.g. using datasets of web videos [13]. However, self-supervision assumes a rigid scene, so dynamic objects are often explicitly discounted [12, 13]. Although depth supervision is difficult to obtain for real videos, 3D films can be exploited. Specifically, MiDaS [23] trained a supervised estimator of the disparity up to a scale and a shift.

Monocular depth estimators cannot guarantee consistency of the predicted depth within a video, nor they use the whole video to improve the prediction. Follow-up methods [20, 29, 49] leverage multi-view geometry to refine the output of (or re-train) monocular predictors. They do so by establishing 2D correspondences with an optical-flow method [44] and compute a reprojection loss. The latter requires per-frame camera poses, which can be obtained from an SfM method (COLMAP [39]), as in CVD [29], or estimated via bundle adjustment, as in Robust CVD [20]. While CVD assume that the dynamic portions of the scene behave rigidly across neighbouring video frames, Robust CVD relaxes them and Dynamic Video Depth (DVD) [49] explicitly predicts the 3D scene flow with an MLP in a manner similar to the dynamic NeRF methods [26, 34, 36, 47]. However, these methods can only model and regularize the deformation locally, at each time instant. While KeyTr also enforces deformations to be consistent with the optical flow, it differs significantly in using a much stronger model of deformation for the entire video; furthermore, it does not use the input from a monocular depth estimator at all.

## 3. Keypoint Transporter

Given a video consisting of $N$ RGB frames $(I^i)_{i=1}^N$, $I^i \in \mathbb{R}^{3 \times H \times W}$ and corresponding masks $(M^i)_{i=1}^N$, $M^i \in \{0,1\}^{H \times W}$ outlining the object of interest, our goal is to predict depth maps $(D^i)_{i=1}^N$, $D^i \in \mathbb{R}^{H \times W}$ containing, for each pixel that belongs to the object, the $z$ component of the corresponding 3D point location in the camera coordinates.

We further assume that frames are labelled with camera projection matrices $(\Pi^i)_{i=1}^N$, $\Pi^i \in \mathbb{R}^{3 \times 4}$, for example obtained from a standard SfM method (COLMAP [39, 40] or ORB-SLAM [32]), and that the optical flow fields $F^{i \mapsto j} \in \mathbb{R}^{2 \times H \times W}$ between pairs of frames $I^i$ and $I^j$ are provided by an off-the-shelf method such as RAFT [44].

**Method overview** Keypoint Transporter (Fig. 2) amounts to estimating a time-varying point cloud $X^i$ that: (1) deforms in a simple manner (*i.e.*, according to a low rank model); (2) covers the object masks $M^i$ well; and (3) deforms consistently with the measured optical flow $F^{i \mapsto j}$. Additionally, (4) we also encourage the colors of the points to be consistent through time and with the images. The rest of the section describes these four ideas in detail.

**Notation.** We assume the perspective camera model with left-multiplication of points by projection matrices. The perspective projection function $u = \pi(\Pi^i, x)$ maps points $x \in \mathbb{R}^3$ in world coordinates to points $u \in \{1, \ldots, H\} \times$

$\{1, \ldots, W\}$ in the camera plane such that $\Pi^i[x; 1] = d_u[u; 1]$, where $d_u = D^i[u] \in \mathbb{R}_+$ is the depth of $u$.

## 3.1. Representing the shape and its deformations

The key component of KeyTr is a deformable 3D point cloud $X^i$ representing the shape of the reconstructed object. Namely, the shape of the object in the $i$-th frame is given by an ordered collection $(x_1^i, \ldots, x_P^i) = X^i \in \mathbb{R}^{P\times 3}$ of $P = 500$ three-dimensional keypoints. The point cloud deforms according to the linear model:

$$X^i = \sum_{k=1}^{K} \alpha_k^i B_k. \tag{1}$$

In this equation, inspired by NRSfM [5,33,45], $B_k \in \mathbb{R}^{P\times 3}$ is the $k$-th element of the deformation basis $B$. The basis contains only a small number $K \ll P$ of elements. It is fixed and shared across all video frames, whereas the coefficients $(\alpha_k^i)_{k=1,\ldots,K}$ express the specific deformation of the object observed in frame $I^i$. In this manner, the possible deformations of the object span a linear subspace of rank $K$. By increasing or decreasing $K$, we encourage more flexible or rigid deformations of the shape, respectively.

Rather than fitting the coefficients $\alpha$ directly, we further constrain them and set them to be the output $\alpha^i = \psi(\gamma(t^i))$ of a small multi-layer perceptron (MLP) $\psi$ that takes as input the timestamp $t^i \in \mathbb{R}$ of the $i$-th frame. Similar to NSFF [24], we pre-process the timestamp with a harmonic positional embedding $\gamma(t)$ before passing into $\psi$. The architecture of $\psi$ closely follows C3DPO's shape predictor [33]. Following [42], in order to prevent reconstructions lying behind cameras, among other losses, our training minimizes the negative-depth penalty $\mathcal{L}^i_{\text{d+}} = \frac{1}{P}\sum_{j=1}^{P} \min(d_{u(x_j^i)}, 0)^2$, where $d_{u(x_j^i)}$ is the depth of the point $x_j^i \in X^i$ in camera $\Pi^i$.

## 3.2. Optimal keypoint transport

In NRSfM, the standard approach for defining a set of keypoints $X$ is to start from a set of 2D feature tracks generated by an off-the-shelf 2D tracker. However, such trackers are usually fragile and often result in incorrect or interrupted tracks. We prefer instead to optimize the 3D keypoints $X$ directly. In this section we define *which* object points should be tracked. We do so in two steps: first, we define a set of $P'$ candidate points locations $\Omega^\star(I^i) = \{u_1^i, \ldots, u_{P'}^i\}$ in each image $I^i$ and then we assign the 3D keypoints $X^i$ to them to provide a good coverage of all such locations.

**Candidate point locations $\Omega^\star(I^i)$.** There are several options for defining the candidate locations $\Omega^\star(I^i)$. While a standard solution would be to select points that yield good features to track (e.g., Harris corners [15], MSER regions [30], or SIFT keypoint detections [28]), such keypoints usually do not cover the surface of the object uniformly and focus only on certain well-textured regions, which may lead to sparse reconstructions. Instead, we sample uniformly at random $P' = 1000$ points $\Omega^\star(I^i)$ from the set of foreground pixels $\Omega^{\text{fg}}(I^i) = \{u \mid M^i[u] = 1\}$.

**Optimal assignment of keypoints to candidate locations.** Next, we explain how the 3D keypoints $X^i$ are assigned to the candidate locations $\Omega^\star(I^i)$. To this end, let $\Omega^\Pi(I^i) = \{\pi(\Pi^i, x_1^i), \ldots, \pi(\Pi^i, x_P^i)\}$ be the 2D projections of the 3D keypoints $\{x_1^i, \ldots, x_P^i\}$. We seek the optimal assignments $A \in \{0,1\}^{P\times P'}$ that minimise our proposed 2D-3D *assignment loss* as follows:

$$\mathcal{L}^i_{\text{asgn}} = \min_A \sum_{\substack{p\in\{1,\ldots,P\},\\ p'\in\{1,\ldots,P'\}}} \rho_{p,p'}\ A_{p,p'}, \tag{2}$$

$$\text{s.t.}: \sum_p A_{p,p'} = 1, \quad \sum_{p'} A_{p,p'} = P'/P, \tag{3}$$

$$A_{p,p'} \in \{0,1\}, \quad \forall p\, \forall p', \tag{4}$$

where $\rho_{p,p'} = \|u_{p'}^i - \pi(\Pi^i, x_p^i)\|/\sigma \in \mathbb{R}_+$ is the Euclidean distance between the candidate location $u_{p'}^i$ and the projected keypoint $\pi(\Pi^i, x_p^i)$.[1]

Fortunately, Eq. (2) is an optimal transport problem and thus can be solved efficiently: if the assignment matrix is relaxed to the continuous range $\hat{A} \in [0,1]^{P\times P'}$, the problem becomes a linear programming (LP) problem which has a solution that coincides with the solution of the integer linear program (2) due to the weights being totally unimodular. Furthermore, while the resulting LP problem is still rather large, we can efficiently find approximate solutions using Sinkhorn's algorithm [9,41], which has the additional benefit of supporting back-propagation of the gradient of $\mathcal{L}^i_{\text{asgn}}$.

Recall that, for memory efficiency, we set the size $P'$ of $\Omega^\star(I^i)$ to 1000 points during every training iteration. In order to alleviate potential optimization issues due to the limited size of $\Omega^\star(I^i)$, we add to $\mathcal{L}^i_{\text{asgn}}$ an additional *Chamfer Distance* term $\mathcal{L}^i_{\text{CD}}$ computed between the full set of foreground pixels $\Omega^{\text{fg}}(I^i)$ and $\Omega^\Pi(I^i)$. Note that it is crucial to jointly optimize $\mathcal{L}^i_{\text{CD}}$ and $\mathcal{L}^i_{\text{asgn}}$ because minimizing only $\mathcal{L}^i_{\text{CD}}$ leads to a non-uniform foreground coverage resulting in large gaps between the reconstructed 3D points.

## 3.3. Consistency of deformation and optical flow

Minimization of the assignment loss $\mathcal{L}^i_{\text{asgn}}$ encourages the 3D keypoints $X$ to cover well the candidate 2D keypoint locations in each frame; however, it does not guarantee that the deformation of $X$ is temporally consistent. In

[1] Eq. (3) assumes that $P$ divides $P'$ (and $P' \geq P$); if not, one (arbitrarily) assigns each column to sum to either $\lfloor P'/P \rfloor$ or $\lfloor P'/P \rfloor + 1$, so that all points are assigned once.

fact, the candidate locations at different time steps are unrelated and, in general, the assignment process can match a given 3D point to different locations for different frames.

In order to encourage the 3D tracks to be compatible with the visual evidence, we employ a flow-consistency loss $\mathcal{L}_{\text{flow}}$. The challenge is how to implement such a loss efficiently, accounting for potential self-occlusions of the 3D points, and in a manner which is easily differentiable.

We show next how a differentiable renderer can be used for this purpose. In order to do so, we first define a set of fixed descriptors, one for each 3D keypoint: $\Phi = (\phi(x_1), \ldots, \phi(x_P))$, where $\phi : \mathbb{R}^3 \rightarrow \mathbb{R}^{D_\phi}$. The descriptors are arbitrary and only used to "color" each keypoint for the purpose of identification.[2] At the beginning of training, they are initialised by taking $P$ uniform random samples $\phi(x_p) \in \{\phi \in \mathbb{R}^{D_\phi} : \|\phi\| = 1\}$ from the $(D_\phi - 1)$-dimensional unit hypersphere. Given these descriptors, we use the PyTorch3D's soft point rasterizer [37] to render the point cloud $X^i$ from the viewpoint of camera $P^i$, resulting in the feature map $R(X^i, \Phi, P^i) \in \mathbb{R}^{D_\phi \times H \times W}$.

Now consider the maps $R(X^i, \Phi, P^i)$ and $R(X^j, \Phi, P^j)$ obtained from two frames $I^i$ and $I^j$. If the 3D trajectories are consistent with the optical flow field $F^{i \mapsto j} \in \mathbb{R}^{2 \times H \times W}$ mapping 2D pixel locations from image $I^i$ to image $I^j$, then the flow must match identical descriptors in the two maps, as these correspond to the same 3D point identity. This is captured by the *flow-consistency loss* $\mathcal{L}^{i \mapsto j}_{\text{flow}} = \Big\| M^i \odot \Big[ R(X^i, \Phi, P^i) - s\big(R(X^j, \Phi, P^j), F^{i \mapsto j}\big) \Big] \Big\|_\varepsilon$, where $s$ denotes differentiable bilinear image sampling[3] and $\|z\|_\varepsilon = \sum_i \varepsilon \left( \sqrt{1 + \left(\frac{z_i}{\varepsilon}\right)^2} - 1 \right)$ is an element-wise soft-Huber norm with the cut-off threshold $\varepsilon = 0.01$.

Note that it is important to use a rendering function $R$ that leverages z-buffering to correctly resolve rendering conflicts between points lying on the same projection ray because the consistency with optical flow $\mathcal{L}_{\text{flow}}$ should be enforced only for the surface points that are visible (and hence rendered) in camera $P^i$.

### 3.4. Low-rank shape appearance

The final supervisory signal aims at assigning an RGB value to each keypoint to match the colors observed in the images $I^i$. While we could make a color constancy assumption and attach a constant color $c_p \in \mathbb{R}^3$ to each keypoint, this would be a poor choice for dynamic objects as their exposure to light can change over time. Furthermore, real-life objects are often non-Lambretian. Instead, extending the ideas above, we allow the colors of points to change over time in a low-rank manner.

Formally, the colors $C^i$ of the keypoints in the $i$-th frame are obtained as a linear combination of a small number $K^c$ of color basis vectors $B^c_k \in \mathbb{R}^{K \times 3}$ as follows:

$$C^i = \tau \left( \sum_{k=1}^{K^c} \beta^i_k B^c_k \right), \tag{5}$$

where $\beta^i_k \in \mathbb{R}$ is the $k$-th color coefficient in frame $I^i$ and $\tau$ is the sigmoid activation function, ensuring that the final colors are bounded to $[0, 1]$. Similar to the shape coefficients $\alpha^i$ in the previous sections, the color coefficients $\beta^i$ are predicted by using a second branch of the time-conditioned MLP $\psi$ that already outputs $\alpha^i$.

Finally, given an RGB frame $I^i$, its corresponding camera pose $P^i$, the point cloud $X^i$ and the composed colors $C^i$, we define the RGB reconstruction loss $\mathcal{L}_{\text{rgb}}$ as follows:

$$\mathcal{L}^i_{\text{RGB}} = \|M^i \odot \left[R(X^i, C^i, P^i) - I^i\right]\|_\varepsilon, \tag{6}$$

using once again the soft point rasterizer $R$ [37].

### 3.5. Training details

Training of KeyTr optimizes the weights of the time-conditioned predictor $\psi$ and of the shape and color bases $B$ and $B^c$ using stochastic gradient descent with Adam optimizer (learning rate = 0.001) until convergence. During each training iteration, we randomly sample a time-ordered list of $N_{\text{batch}}$ indices $J = (i_1, \ldots, i_{N_{\text{batch}}})$ such that $1 \leq i_1 \leq \cdots \leq i_{N_{\text{batch}}} \leq N$, and backpropagate w.r.t. the batch loss, which is defined as follows:

$$\sum_{i \in J} (w_{\text{asgn}} \mathcal{L}^i_{\text{asgn}} + w_{\text{CD}} \mathcal{L}^i_{\text{CD}} + w_{\text{RGB}} \mathcal{L}^i_{\text{RGB}} + w_{\text{d+}} \mathcal{L}^i_{\text{d+}}) \\ + \sum_{j=1}^{N_{\text{batch}}-1} w_{\text{flow}} \mathcal{L}^{J_j \mapsto J_{i+1}}_{\text{flow}}, \tag{7}$$

with weights $w_{\text{asgn}} = 0.1, w_{\text{CD}} = 10.0, w_{\text{RGB}} = 1.0, w_{\text{flow}} = 1.0, w_{\text{d+}} = 100.0$.

## 4. New benchmarking data

Assessing dynamic video depth reconstruction quantitatively requires videos of deformable objects with known depth. Furthermore, in order to clearly isolate the performance of a depth estimation method from exogenous factors such as the object segmentation quality, this data should come with ground truth information for object masks and other parameters such as camera poses and calibrations.

In order to better test our algorithm quantitatively and qualitatively, we introduce a number of new synthetic and real video benchmarks with dynamic deformable objects, described below and illustrated in Fig. 3.

[2] Hence $\Phi$ can be thought as a random projection of the indicator function of each keypoint, which is a much lower dimensional representation of its identity because $D \ll P$.

[3] Implemented with the `grid_sample` function of PyTorch [35].

|  | Animals-in-motion |  |  | Humans-in-motion – `easy` |  |  | Humans-in-motion – `diff.` |  |  |
|---|---|---|---|---|---|---|---|---|---|
|  | $\ell_1^{\text{d-scaled}}$ | $\ell_1^{\text{d-seq-scaled}}$ | $\ell_1^{\text{d}}$ | $\ell_1^{\text{d-scaled}}$ | $\ell_1^{\text{d-seq-scaled}}$ | $\ell_1^{\text{d}}$ | $\ell_1^{\text{d-scaled}}$ | $\ell_1^{\text{d-seq-scaled}}$ | $\ell_1^{\text{d}}$ |
| MiDAS$^\dagger$ [23] | 0.567 | 0.873 | 5.826 | 0.870 | 1.058 | 5.830 | 0.131 | 0.247 | **0.485** |
| DVD$^\dagger$ [49] | 0.577 | 0.588 | 0.720 | 1.119 | 1.136 | 2.409 | 0.102 | 0.185 | 0.684 |
| RCVD$^\dagger$ [20] | 0.678 | 0.904 | 0.941 | 1.094 | 1.232 | 1.245 | 0.129 | 0.227 | - |
| NSFF$^\dagger$ [26] | 6.861 | 8.418 | 8.418 | 4.274 | 7.256 | 7.256 | 0.372 | 3.119 | 3.119 |
| LASR [48] | 1.630 | 1.635 | 1.799 | 7.349 | 7.397 | 7.452 | 1.615 | 1.723 | 3.303 |
| **KeyTr** | **0.135** | **0.171** | **0.196** | **0.098** | **0.161** | **0.183** | **0.086** | **0.181** | 0.780 |

Table 1. Quantitative comparison of depth estimation on all considered datasets (Animals-in-motion, Humans-in-motion). We report the absolute metric depth error $\ell_1^{\text{d}}$, the scale-invariant depth error $\ell_1^{\text{d-scaled}}$, and the sequence-scale-invariant $\ell_1^{\text{d-seq-scaled}}$. Methods labelled with $^\dagger$ require a monocular depth predictor supervised with ground truth annotations.

Figure 3. We evaluate on Humans-in-motion and Animals-in-motion synthetic datasets as well as the Pet AMT video datataset.

**Humans-in-motion.** This dataset contains 13 videos of animated synthetic humans performing simple full-body movements and actions. There are two subsets: `easy`, with a single person in a video without foreground occlusions, and `difficult`, with people that can be occluded by each other and objects in the environment. Similarly to [16], our dataset is based on 375 3D scans from the RenderPeople dataset,[4] animated using motion capture sequences. Scenes are rendered using Blender [8]. Background uses environment maps for the `easy` set and models from the Replica dataset [43] (scans of real indoor environments) for the `difficult` set. Each video sequence contains between 250 and 300 frames (≈10 sec long) and is rendered using randomly sampled smooth camera trajectories. All samples in this dataset contain ground-truth depth maps, object masks and camera trajectories.

**Animals-in-motion.** This dataset is similar to Humans-in-motion, but it contains 10 videos of animals (dog, horse, cow, sheep, chimp) modelled and manually animated by a 3D artist. Each video sequence contains 100 frames (≈3 sec long) and is rendered using randomly sampled camera trajectories. As in the previous case, this dataset contains per-frame ground-truth object masks, depth maps and camera trajectories for all sequences.

**Pet AMT videos.** In order to benchmark on challenging real-life data, we also collected videos of pets (cats and dogs) using Amazon Mechanical Turk (AMT) following the data collection protocol recently proposed in [38]. We instructed each user to arrange a pet-centric scene with the user slowly circling around the animal while keeping it fully within the field of view. After collection, we sample 300 uniformly spaced frames from each video and extract camera extrinsics using COLMAP [39]. We further generate foreground masks using the PointRend segmenter [18]. Since the dataset does not provide ground-truth depth, we only evaluate qualitatively.

## 5. Experiments

We compare our Keypoint Transporter to several baselines using the datasets introduced above. The baselines, together with the evaluation protocol, are detailed here.

**Evaluation protocol** We measure the depth reconstruction accuracy for the regions that contain a dynamic object (*e.g.*, a human or an animal). Formally, given a set of ground truth depth maps $\{D^i\}_{i=1}^N$, $D^i \in \mathbb{R}_+^{H\times W}$, their corresponding predictions $\{\hat{D}^i\}_{i=1}^N$, $\hat{D}^i \in \mathbb{R}_+^{H\times W}$, and the foreground object masks $M^i \in \{0,1\} \in \mathbb{R}^{H\times W}$, we evaluate the absolute metric depth error $\ell_1^{\text{d}} = \frac{1}{|M^i|}\sum_{u\in M^i}\left|D^i[u] - \hat{D}^i[u]\right|$, which is averaged over the set of foreground pixels $M^i$, and then over the frames and videos in the benchmark.

Since $\ell_1^{\text{d}}$ is often dominated by the incorrect scaling of the predicted depth map, we also consider a loss that is scale invariant. Specifically, we define $\ell_1^{\text{d-scaled}} = \arg\min_{s\in\mathbb{R}_+} \frac{1}{|M^i|}\sum_{u\in M^i}\left|D^i[u] - s\hat{D}^i[u]\right|^2$, thus finding the best scale $s$ that aligns the prediction with the ground truth. Intuitively, $\ell_1^{\text{d-scaled}}$ ignores the scale mismatch and, as such, solely evaluates the quality of the reconstructed shape.

Finally, we also report $\ell_1^{\text{d-seq-scaled}}$ which differs from $\ell_1^{\text{d-scaled}}$ by sharing the same scale value $s$ for all the frames of a given sequence. The benefit compared to per-frame rescaling is that a low error in this metric means that depth is reconstructed consistently for the duration of the video, still up to an overall scaling factor.

**Baselines** Our **KeyTr** is compared to several depth estimation and 3D reconstruction baselines, described next. **LASR [48]** optimizes a deformable mesh in a manner which is consistent with the estimated optical flow and foreground segmentation. For fairness, we modify it to use the ground

[4] http://renderpeople.com/

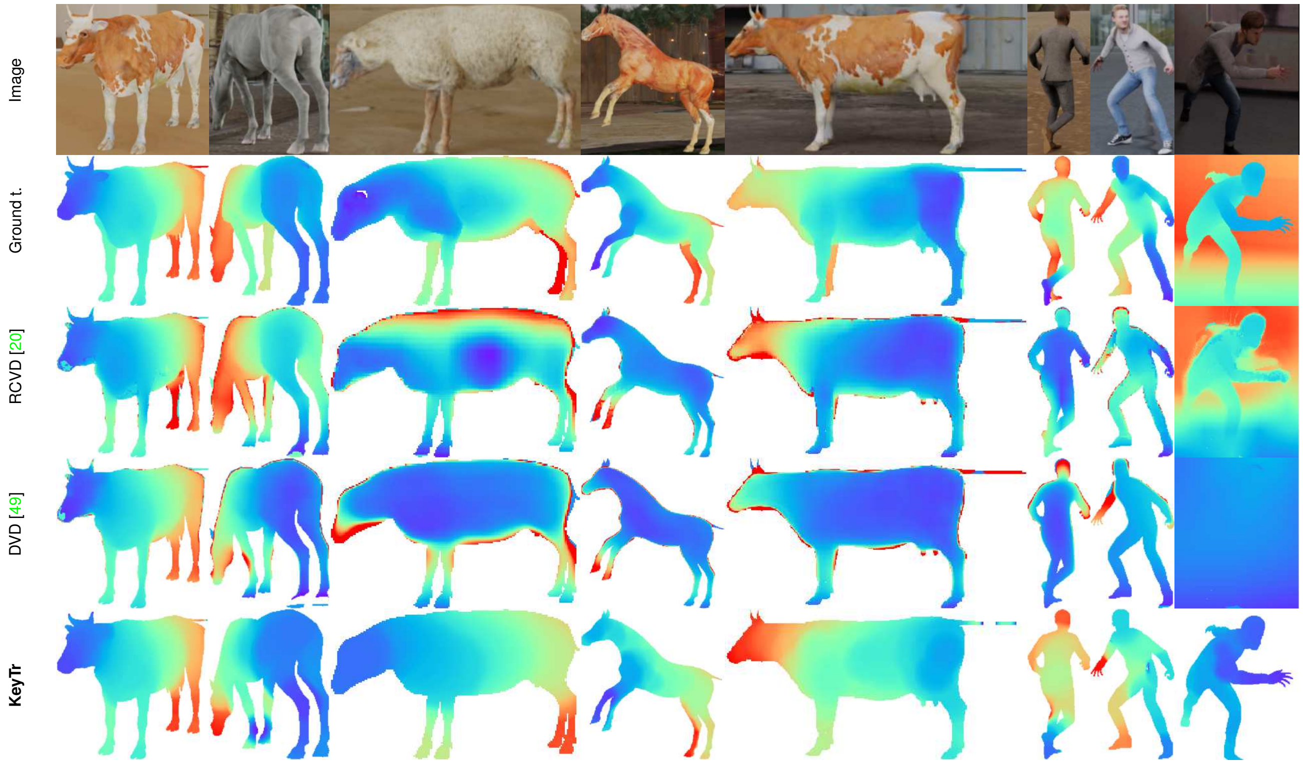

Figure 4. Qualitative evaluation of depth maps reconstructed by Keypoint Transporter and corresponding baselines from Animals-in-motion(first 5 columns) and Humans-in-motion(`easy` set, last 2 columns) video sequences using ground-truth object masks.

truth camera motions. **MiDaS** [23] is a state-of-the-art monocular depth predictor trained on a large variety of video datasets. Because frames are processed independently, depth predictions are not consistent through time, nor the scale is consistent with the ground truth camera motion, usually leading to poor performance in metric depth error $\ell_1^{\text{d}}$ and, to a lesser extent, $\ell_1^{\text{d-seq-scaled}}$. **DVD** [49] and **RCVD** [20] fine tune the output of MiDAS to be video-consistent and are described in detail in Sec. 1. **NSFF** [26] is an extension of NeRF [31] that can handle a dynamic scene by estimating a 3D flow field to represent the deformation of dynamic objects. While NSFF is meant for new-view synthesis, it is possible to extract depth by computing the expected termination of the camera rays in the volume.

## 5.1. Results

Table 1 contains quantitative comparisons of all methods. Our approach significantly outperforms DVD, RCVD, LASR and NSFF on the Humans-in-motion-`easy` and Animals-in-motion datasets, where our KeyTr produces an average error $<$ 20cm, while other methods have errors above 50cm, sometimes reaching several meters.

For the Humans-in-motion-`difficult` dataset, results are mixed. Most methods have a difficult time estimating the true metric depth of the videos, but perform reasonably up to a scaling factor. In this case, our approach is better for the scaled metrics $\ell_1^{\text{d-scaled}}$ and $\ell_1^{\text{d-seq-scaled}}$ (arguably the most useful ones in many applications), but DVD outperforms it in the non-scaled metric $\ell_1^{\text{d}}$.

| $\mathcal{L}_{\text{asgn}} + \mathcal{L}_{\text{CD}}$ | $\mathcal{L}_{\text{flow}}$ | $\mathcal{L}_{\text{rgb}}$ | $\ell_1^{\text{d-scaled}}$ | $\ell_1^{\text{d-seq-scaled}}$ | $\ell_1^{\text{d}}$ |
|---|---|---|---|---|---|
| | | ✔ | 0.87 | 0.91 | 0.91 |
| | ✔ | | 8.36 | 8.36 | 8.36 |
| | ✔ | ✔ | 8.00 | 8.04 | 8.00 |
| ✔ | | | 0.22 | 0.23 | 0.31 |
| ✔ | | ✔ | 0.19 | 0.21 | 0.27 |
| ✔ | ✔ | | 0.13 | 0.16 | 0.19 |
| ✔ | ✔ | ✔ | 0.14 | 0.17 | 0.20 |

Table 2. The ablation study evaluating the contribution of each loss term of our method on the Animals-in-motion dataset.

Additionally, Figures 4 and 5 contain qualitative evaluation of Keypoint Transporter on all considered datasets.

**Ablation study.** Next we evaluate the contribution of the components of KeyTr by turning them off and recording the change of performance. In more detail, we switch on/off the loss terms $\mathcal{L}_{\text{flow}}$, $\mathcal{L}_{\text{asgn}}$+$\mathcal{L}_{\text{CD}}$, $\mathcal{L}_{\text{rgb}}$. Table 2 contains the results on the synthetic Animals-in-motion dataset.

As shown in the table, assigning vertices to candidate object points is essential for performance ($\mathcal{L}_{\text{asgn}}$+$\mathcal{L}_{\text{CD}}$), as

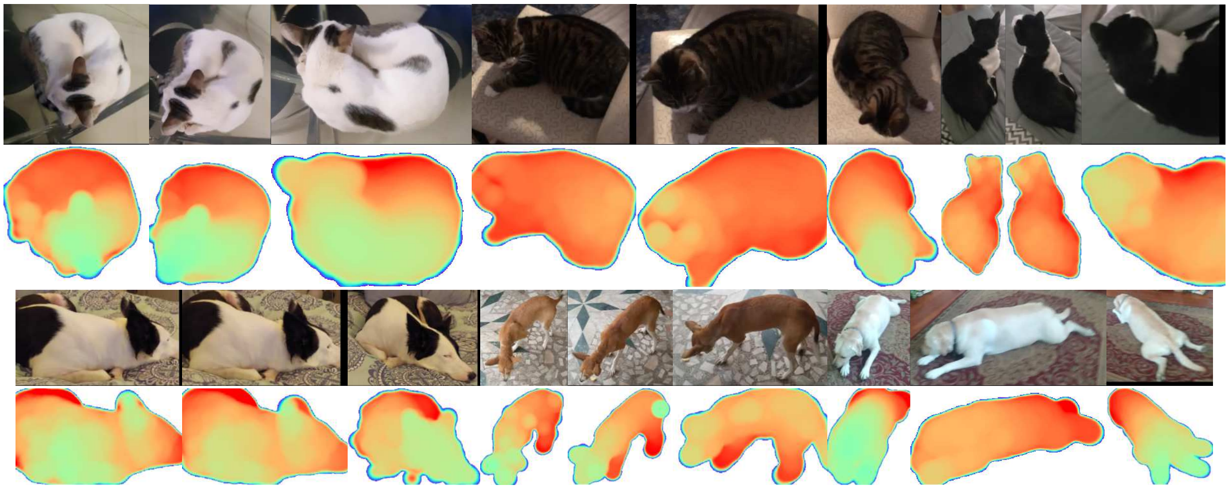

Figure 5. Depth predicted by our method (KeyTr) on real-life sequences from our collected Pet AMT videos of cats and dogs.

without the latter the learned point cloud does not cover the object properly. The RGB loss $\mathcal{L}_{\text{rgb}}$ is helpful in terms of depth estimation, but only when the flow loss $\mathcal{L}_{\text{flow}}$ is not used as well. Adding $\mathcal{L}_{\text{flow}}$ to $\mathcal{L}_{\text{asgn}}+\mathcal{L}_{\text{CD}}$ has the biggest effect, reducing the prediction error by 30-40%, depending on the metrics. However, also adding $\mathcal{L}_{\text{rgb}}$ to these two does not bring further improvements.

**Varying the number of basis shapes** We further analyse the regularisation effect of the number of vectors $K$ of the shape basis $B$. Figure 6 records the depth errors attained by our method on Animals-in-motion as a function of $K$. It is obvious that too low/high values of $K$ hurt all depth errors (especially the most sensitive metric error $\ell_1^{\text{d}}$) suggesting the benefits of our low-rank shape regularization.

## 6. Broader impact and limitations

Monocular depth reconstruction is a general-purpose technique, which incurs the usual risks of misuse, bias and lack of fairness typical of computer-vision methods based on machine learning. KeyTr, however, is less susceptible to bias since the model is trained from scratch on each new video. Limitations of our method include failures of reconstructing videos that exhibit little camera parallax compared to the object deformation, and ones with occlusions caused either by other parts of the scene or due to the object leaving the field of view. As for the data used in the paper, we use RenderPeople and Replica in a manner compatible with their licenses; these datasets were collected with full subject consent where applicable. Furthermore, we created the 3D animal assets ourselves. Real pet video data collection was reviewed and approved by our institutional review board.

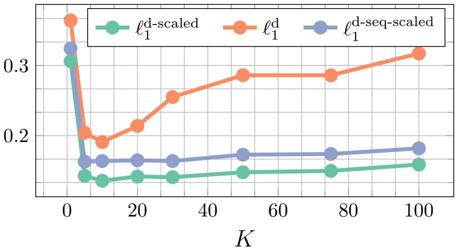

Figure 6. Absolute depth errors of our method (Keypoint Transporter) on Animals-in-motion as a function of the number of basis vectors $K$. The best performance is attained for $K = 10$.

## 7. Conclusions

While there has been significant progress in the development of methods that can estimate depth from casual videos of dynamic objects, we have identified common limitations shared by these approaches: their inability to track, model and constrain the deformation of the objects for the duration of the video. This has inspired us to design a very different algorithm with the main goal of explicitly capturing such deformations. By doing so, we have shown empirically that our method is often significantly more accurate than competitors in recovering the geometry of the dynamic objects despite using no learned geometric prior such as a monocular depth estimator. However, this has also limitations, such as obtaining poorer results when the camera parallax is very small. We argue that combining learned depth priors with our deformation-aware model might solve this and further improve the general performance of the system.

## References

[1] Antonio Agudo and Francesc Moreno-Noguer. Dust: Dual union of spatio-temporal subspaces for monocular multiple object 3d reconstruction. In *Proc. CVPR*, 2017. 2

[2] Antonio Agudo, Melcior Pijoan, and Francesc Moreno-Noguer. Image collection pop-up: 3D reconstruction and clustering of rigid and non-rigid categories. In *Proc. CVPR*, 2018. 2

[3] Ijaz Akhter, Yaser Sheikh, Sohaib Khan, and Takeo Kanade. Nonrigid structure from motion in trajectory space. In *Proc. NIPS*, 2009. 2

[4] Ijaz Akhter, Yaser Sheikh, Sohaib Khan, and Takeo Kanade. Trajectory space: A dual representation for nonrigid structure from motion. *PAMI*, 33(7):1442–1456, 2011. 2

[5] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3D shape from image streams. In *Proc. CVPR*, 2000. 2, 4

[6] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3D shape from image streams. In *Proc. CVPR*, 2000. 2

[7] Wenzheng Chen, Jun Gao, Huan Ling, Edward J. Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Learning to Predict 3D Objects with an Interpolation-based Differentiable Renderer. In *Proc. NeurIPS*, 2019. 2

[8] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 6

[9] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013. 4

[10] Yuchao Dai, Hongdong Li, and Mingyi He. A simple prior-free method for non-rigid structure-from-motion factorization. *International Journal of Computer Vision*, 107(2):101–122, 2014. 2

[11] Katerina Fragkiadaki, Marta Salas, Pablo Arbelaez, and Jitendra Malik. Grouping-based low-rank trajectory completion and 3D reconstruction. In *Proc. NIPS*, 2014. 2

[12] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel Brostow. Digging Into Self-Supervised Monocular Depth Estimation. In *IEEE International Conference on Computer Vision*, number 1, 2019. 3

[13] Ariel Gordon, Hanhan Li, Rico Jonschkowski, and Anelia Angelova. Depth from Videos in the Wild: Unsupervised Monocular Depth Learning from Unknown Cameras. In *IEEE International Conference on Computer Vision*, 2019. 3

[14] Paulo FU Gotardo and Aleix M Martinez. Non-rigid structure from motion with complementary rank-3 spaces. In *Proc. CVPR*, 2011. 2

[15] Christopher G. Harris and Mike Stephens. A combined corner and edge detector. In *Proc. of The Fourth Alvey Vision Conference*, 1988. 4

[16] Zeng Huang, Yuanlu Xu, Christoph Lassner, Hao Li, and Tony Tung. Arch: Animatable reconstruction of clothed humans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. MIT, 2020. 6

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

[18] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. MIT, 2020. 6

[19] Filippos Kokkinos and Iasonas Kokkinos. Learning monocular 3D reconstruction of articulated categories from motion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[20] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 3, 6, 7

[21] Suryansh Kumar, Anoop Cherian, Yuchao Dai, and Hongdong Li. Scalable dense non-rigid structure from motion: A grassmannian perspective. In *Proc. CVPR*, 2018. 2

[22] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Spatial-temporal union of subspaces for multi-body non-rigid structure-from-motion. *Pattern Recognition Journal*, 2017. 2

[23] Katrin Lasinger, René Ranftl, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *arXiv*, abs/1907.01341, 2019. 2, 3, 6, 7

[24] Tianye Li, Mira Slavcheva, Michael Zollhöfer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, and Zhaoyang Lv. Neural 3d video synthesis. *arXiv.cs*, abs/2103.02597, 2021. 4

[25] 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. Technical report, 2020. 2

[26] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. *arXiv.cs*, abs/2011.13084, 2020. 3, 6, 7

[27] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Transactions on Graphics*, 38(4), 2019. 2

[28] David G. Lowe. Object recognition from local scale-invariant features. In *Proc. ICCV*, 1999. 4

[29] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Trans. Graph.*, 39(4):71, 2020. 2, 3

[30] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proc. BMVC*, 2002. 4

[31] 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. 2, 7

[32] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. on Robotics*, 31(5), 2015. 3

[33] David Novotny, Nikhila Ravi, Benjamin Graham, Natalia Neverova, and Andrea Vedaldi. C3DPO: Canonical 3d pose networks for non-rigid structure from motion. In *Proc. ICCV*, 2019. 2, 4

[34] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B. Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Deformable neural radiance fields. *arXiv*, 2020. 3

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 5

[36] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. *arXiv.cs*, abs/2011.13961, 2020. 3

[37] 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. 5

[38] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the International Conference on Computer Vision (ICCV)*. MIT, 2021. 6

[39] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proc. CVPR*, 2016. 3, 6

[40] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proc. ECCV*, 2016. 3

[41] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967. 4

[42] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Proc. NeurIPS*, 2019. 4

[43] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 6

[44] Zachary Teed and Jia Deng. RAFT: recurrent all-pairs field transforms for optical flow. In *Proc. ECCV*, 2020. 2, 3

[45] L. Torresani, A. Hertzmann, and C. Bregler. Learning non-rigid 3D shape from 2D motion. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Proc. NeurIPS*, volume 16. MIT, Cambridge, MA, 2004. 4

[46] Lorenzo Torresani, Aaron Hertzmann, and Chris Bregler. Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors. *PAMI*, 30(5):878–892, 2008. 2

[47] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a deforming scene from monocular video. *arXiv*, 1(1), 2020. 3

[48] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Huiwen Chang, Deva Ramanan, William T. Freeman, and Ce Liu. LASR: Learning Articulated Shape Reconstruction from a Monocular Video. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 6

[49] Z. Zhang, F. Cole, R. Tucker, W. T. Freeman, and T. Dekel. Consistent depth of moving objects in video. In *Proc. SIGGRAPH*, 2021. 2, 3, 6, 7

[50] Xiaowei Zhou, Menglong Zhu, Spyridon Leonardos, and Kostas Daniilidis. Sparse representation for 3D shape estimation: A convex relaxation approach. *PAMI*, 2016. 2

[51] Yingying Zhu, Dong Huang, Fernando De La Torre, and Simon Lucey. Complex non-rigid motion 3D reconstruction by union of subspaces. In *Proc. CVPR*, 2014. 2