Unbiased 4D: Monocular 4D Reconstruction with a Neural Deformation Model

Erik C.M. Johnson, Marc Habermann, Soshi Shimada, Vladislav Golyanik, Christian Theobalt
Max Planck Institute for Informatics, Saarbrücken, Germany
ecmjohnson@gmail.com, {mhaberma, sshimada, golyanik, theobalt}@mpi-inf.mpg.de

Abstract

Capturing general deforming scenes from monocular RGB video is crucial for many computer graphics and vision applications. However, current approaches suffer from drawbacks such as struggling with large scene deformations, inaccurate shape completion, or requiring 2D point tracks. In contrast, our method, Ub4D, handles large deformations, performs shape completion in occluded regions, and can operate on monocular RGB videos directly by using differentiable volume rendering. This technique includes three new—in the context of non-rigid 3D reconstruction—components, i.e., 1) A coordinate-based and implicit neural representation for non-rigid scenes, which in conjunction with differentiable volume rendering enables an unbiased reconstruction of dynamic scenes, 2) a proof that extends the unbiased formulation of volume rendering to dynamic scenes, and 3) a novel dynamic scene flow loss, which enables the reconstruction of larger deformations by leveraging the coarse estimates of other methods. Results on our new dataset, which will be made publicly available, demonstrate a clear improvement over the state of the art in terms of surface reconstruction accuracy and robustness to large deformations.

1. Introduction

Reconstructing the deforming 3D geometry of an object from image data is a long-standing and important problem in computer vision with many applications in the movie and game industries, as well as VR and AR. Especially interesting and the subject of this work is the 4D reconstruction from a single RGB video, as this is the most intuitive and user-friendly capture setup. Over the last decade, many monocular 4D reconstruction approaches have been proposed; they can be categorized into dense non-rigid structure from motion (NRSfM) methods, shape-from-template (SfT) approaches, and neural template-free approaches.

NRSfM methods [1, 4, 8, 13, 22, 24, 37, 39] usually assume dense and coherent 2D point tracks connecting the frames of the video. While accurate results can be obtained, it is usually hard to satisfy this assumption in real-world captures, limiting the use case in practice. SfT methods [7, 21, 33, 36, 50, 50] assume an object template is given. While this provides a strong prior for this highly ill-posed task, initial reconstruction errors in the template can lead to tracking errors. Importantly, topological changes cannot be captured by such methods. Last, template-free learning-based approaches have shown compelling results for category-specific (*e.g.,* humans [31, 32]) and general scenes with small deformations [29, 40]. However, generalization beyond categories and accurate reconstruction of an explicit geometry remains a challenge.

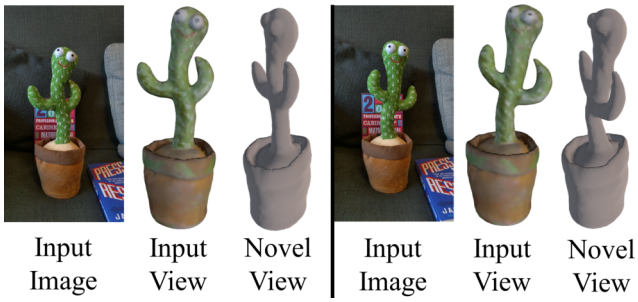

Figure 1. We present a new method for 4D reconstruction of dynamic scenes using a single RGB video. In contrast to previous work, our method completes the object as it is observed from different view angles, can handle small- and large-scale deformations of arbitrary objects due to our separation of non-rigid deformations using a canonical space, our unbiased volume rendering formulation, and an optional scene flow loss.

To this end, we propose Unbiased 4D (Ub4D), *i.e.,* a novel method for the 4D reconstruction of a deforming object given a single RGB video of the object; see Figure 1. In contrast to previous method classes such as non-rigid structure-from-motion and shape-from-template, Ub4D completes the shape as it is being observed from other view angles. Using a signed distance field (SDF) network, we represent the object of interest as an implicit SDF in canonical space. In order to obtain the deformed per-frame geometry, we propose a bending network, which deforms the current frame into a shared canonical space. To supervise the SDF and bending network, we impose an unbiased volume rendering loss, which extends prior work [43] to dynamic scenes. Notably, we also prove the correctness of our formulation. In particular, we compare the rendered images and object segmentation masks with the ground truth images and masks. Similar to previous works [40], this formulation alone still struggles with larger

scene deformations. Thus, when the scene contains large scene deformations, we propose a scene flow loss, which attaches free space to a set of tracked 3D points in order to guide the scene deformations predicted by the bending network. In summary, our primary technical contributions are as follows:

- Ub4D, *i.e.,* a new approach for dense 4D reconstruction from monocular image sequences based on an implicit surface representation and a dynamic bending network (Sec. 3).
- Extending the unbiased formulation of volume rendering [43] to general deforming scenes (Sec. 3.2).
- A new scene flow loss leveraging coarse geometric proxies (dense and sparse), which further increases the robustness to large-scale scene deformations (Sec. 3.3).
- A new synthetic benchmark dataset for general and large-scale deforming scenes (Sec. 4).

We demonstrate that our method outperforms the previous state of the art in terms of accuracy and robustness to large scale scene deformations (Secs. 4.1-4.4). Our code and the new dataset will be made publicly available.

2. Related Work

Several method classes for 3D reconstruction of non-rigidly deforming surfaces from monocular images are known. They differ in the assumptions they make about the available priors and types of motions and deformations.
Non-Rigid Structure from Motion (NRSfM) operates on point tracks over the input monocular views [4, 39]. It factorizes them into camera poses and deformable (per-frame) geometry of observed surfaces. Assuming that accurate point tracks can be obtained is a restrictive assumption. If points of the input views are tracked densely, NRSfM can then even be used to obtain dense surfaces [1, 8, 24]. Both neural NRSfM methods for the sparse [13, 22, 41] and dense [37] cases were recently proposed in the literature. Deep NRSfM [13, 22, 37, 41] is related to NRSfM in that it lifts 2D input points in 3D and does not rely on 3D supervision. Ub4D is similar to NRSfM in that 1) it has the least number of assumptions (no training datasets, no 3D priors) and 2) requires camera or object movement while recording the scene. It differs from NRSfM in that it operates directly on images with no need for 2D correspondences.
Shape from Template (SfT). This class of techniques assumes a 3D shape prior called a *template*. SfT is then posed as the problem of tracking and deforming the template so that the new states plausibly reproject to the input images [9, 21, 33, 50]. While some approaches have demonstrated accurate results, even for larger deformations, they come at the cost of being category-specific (*e.g.,* they only work for humans [3, 10]). Further, the assumption of a known 3D template is limiting when dealing with unknown objects. Moreover, obtaining the template usually requires a separate step, which can be difficult. ϕ-SfT [11] explains 2D observations through physics-based simulation of the deformation process. In contrast to them, we do not model physics laws explicitly. Moreover, we target a different class of non-rigid objects (thin surfaces [11] *vs* articulated objects). Deep SfT or direct surface regression methods assume multiple states available for training [7, 36]. Our approach differs from SfT in that it only requires 2D images as input. Nonetheless, it can benefit from a subset of frames observing static scene states to initialise the canonical volume. Note that—in contrast to SfT techniques—observing a scene under rigidity assumption [50] or having a template in advance from elsewhere is not a strict requirement for us.

Monocular 3D Mesh Reconstruction. 3D mesh reconstruction methods deform an initial mesh to match image observations [12, 15, 42, 45]. They are exclusively neural techniques, usually trained using 2D image collections. One of their limitations is that large image sets are not available for all object categories (*e.g.,* consider rarely observed biological species). Moreover, the methods, which do not require 2D image priors, might capture coarse articulations but fail to reconstruct fine surface details [12, 45]. Starting from a sphere mesh is a restricting assumption. Even though many watertight meshes are, in theory, topologically equivalent to a sphere, a practical attempt to guide sphere deformations by image cues can converge to local minima.

Free Viewpoint Video and Neural Surface Extraction. Coordinate-based volumetric neural representations learned from 2D observations, such as NeRF [20], can be used to render high-quality novel views of rigid [5, 17, 49] and non-rigid scenes [16, 18, 25, 27–29, 40, 44]. While they have shown impressive results, the volumetric representation they use lacks surface constraints so that it is difficult to extract high-quality surfaces from the learned representation. Some works [23, 43, 47, 48] propose to represent 3D scenes as a neural SDF and use volume rendering to learn the representation. We are inspired by the recent progress in neural volumetric representations learned without 3D supervision. Even though our goal is not novel view rendering and editing, we show that a NeRF-inspired component can be useful for monocular non-rigid 3D reconstruction. Moreover, surface extraction methods [23, 43, 47, 48] have focused on rigid objects so far. One concurrent work of Anonymous et al. [14] investigates how to improve the training time of NeuS [43] and how network weights can be incrementally refined throughout a dynamic scene. However, this is only distantly related to our work since they assume a multi-view camera setup and, in stark contrast to our method, do not treat the dynamic scene jointly but on a frame-by-frame basis. Thus, we demonstrate that the

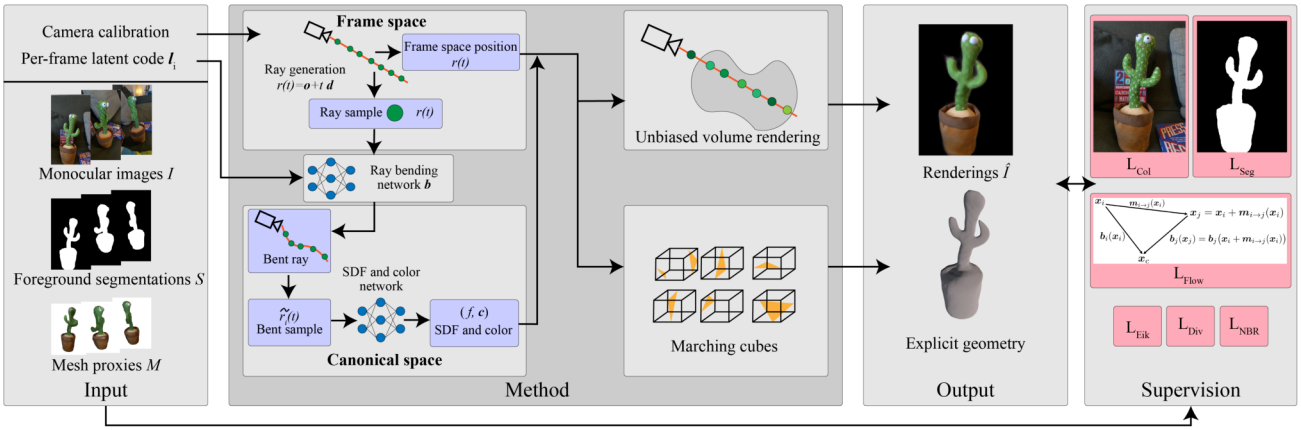

Figure 2. Ub4D takes as input a sequence of images recorded with a single RGB camera and foreground masks. Each frame is also equipped with a learnable latent code, and our method learns a canonical and colored SDF representing the static scene. Our bending network maps the frame space to canonical space and volume rendering and marching cubes can produce per-frame renderings and geometries. We weakly supervise our scene representation with image-based losses and spatiotemporal priors including our novel scene flow loss.

problem we are interested in, *i.e.,* monocular *non-rigid* 3D reconstruction, significantly benefits from advances in another, distantly related research direction.

3. Method

The goal of Ub4D is to reconstruct the dense and deforming surface of an object from a single RGB video. Therefore, our method takes as input the monocular image sequence $\{I_i, S_i : i \in [1, N_f]\}$ of the segmented object consisting of N_f RGB images I_i and respective segmentation masks S_i. We assume the extrinsic and intrinsic camera parameters are known. Optionally, corresponding per-frame coarse geometric proxies with N_v vertices can be provided $\{M_i : i \in [1, N_f]\}$, where $M_i = \{\boldsymbol{v}_i^{(k)} : k \in [1, N_v]\}$ and $\boldsymbol{v}_i^{(k)}$ denotes vertex k of the mesh in frame i. Note that we only use corresponding vertices, i.e. no connectivity information, which allows the use of sparse point sets (e.g. skeleton) as the geometric proxy. Given these inputs, Ub4D outputs an explicit geometry for every frame; see Fig. 2.

3.1. Non-Rigidity Model

We model temporal non-rigid deformations as a vector field projecting points from the frame space into a canonical space. One can conceptualize this by considering it as a bending of the straight rays originating from the camera. Given a straight ray with an origin $\boldsymbol{o} \in \mathbb{R}^3$ and a viewing direction $\boldsymbol{d} \in \mathbb{R}^3$ as $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{d}$, we bend this ray with a frame-specific bending network $\boldsymbol{b}_i : \mathbb{R}^3 \to \mathbb{R}^3$ as $\tilde{\boldsymbol{r}}_i(t) = \boldsymbol{r}(t) + \boldsymbol{b}_i(\boldsymbol{r}(t))$ where i denotes the frame. This bent ray is a directed parametric path in $\mathbb{R}^3$ like the straight ray, but, where the derivative of the straight ray is constant (*i.e.* $\frac{\mathrm{d}\boldsymbol{r}(t)}{\mathrm{d}t} = \boldsymbol{d}$), the bent ray has an instantaneous direction at each point along it of $\frac{\mathrm{d}\tilde{\boldsymbol{r}}_i(t)}{\mathrm{d}t} = \boldsymbol{d} + \frac{\partial \boldsymbol{b}_i}{\partial \boldsymbol{r}(t)}\boldsymbol{d}$. We desire that this bending network transforms points from frame space into a single canonical representation of the object shared by all frames of the input.

While this discussion presents the bending network as a per-frame vector field throughout $\mathbb{R}^3$, it is implemented using a per-frame latent code $\boldsymbol{l}_i \in \mathbb{R}^{64}$. This latent code is given as input along with a point in space to a Multi-Layer Perceptron (MLP) $\boldsymbol{b} : (\mathbb{R}^3, \mathbb{R}^{64}) \to \mathbb{R}^3$ and the latent codes are optimized during training. Importantly, the latent code $\boldsymbol{l}_i$ passed to the bending network is the *only* frame-specific element in our method and no other network receives it.

This is similar to the non-rigidity model employed in NR-NeRF [40]. However, we propose a different regularization to enable the modeling of larger deformations, which also removes the need to learn a rigidity score throughout the scene. Whereas NR-NeRF [40] penalizes the bending network output for its *absolute* length, we instead enforce that the deformation of the current frame is similar to that of the neighboring frames. This assumes that neighboring frames represent similar object states, which is a more reasonable assumption for dynamic scenes compared to the absolute amount of deformation. More specifically, for N_s samples along a straight ray $\boldsymbol{r}$, we penalize the bending network as:

$$L_{\text{NBR}} = \frac{1}{N_s} \sum_{z=1}^{N_s} \sum_{j \in \mathcal{N}(i)} \omega_i^{(z)} \, ||\boldsymbol{b}_i(\boldsymbol{r}(t^{(z)})) - \boldsymbol{b}_j(\boldsymbol{r}(t^{(z)}))||_2^2, \quad (1)$$

where $\omega_i^{(z)}$ is the visibility weight at sample z along the bent ray (see Sec. 3.2) and $\mathcal{N}(i)$ are the neighbours of frame i. We also penalize the divergence of the bending network as:

$$L_{\text{DIV}} = \frac{1}{N_s} \sum_{z=1}^{N_s} \omega_i^{(z)} \, |\nabla \cdot \boldsymbol{b}_i(\boldsymbol{r}(t^{(z)}))|^2, \quad (2)$$

where we use the unbiased, approximated divergence as presented in Tretschk *et al.* [40].

3.2. Rendering Method

Recent studies on *static* scene reconstruction have demonstrated that volume rendering enables more stable training compared to surface rendering [43, 47, 48]. Therefore, we extend the volume rendering method proposed in NeuS [43] to *dynamic* scenes. The full proof of unbiasedness is in the supplement. Let $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ be a Signed Distance Field (SDF) modeled by an MLP that takes as input sampled points $\tilde{\boldsymbol{r}}_i(t)$ on the bent ray. Then, NeuS [43] shows that we can compute the opaque density as:

$$\rho_i(\tilde{\boldsymbol{r}}_i(t)) = \max\left\{ \frac{-\frac{\mathrm{d}\Phi_s}{\mathrm{d}t}\big(f(\tilde{\boldsymbol{r}}_i(t))\big)}{\Phi_s\big(f(\tilde{\boldsymbol{r}}_i(t))\big)}, 0 \right\}, \tag{3}$$

where Φ_s is the logistic Cumulative Distribution Function (CDF) with standard deviation s^{-1}. This is in contrast to the original formulation operating on unbent ray sample points rather then bent ones. To calculate the color of each camera ray, we employ a hierarchical sampling with N_s samples in total (samples in coarse and fine stages) along the *bent* ray $\{\tilde{\boldsymbol{r}}_i(t^{(z)}) : z \in \mathbb{Z}, z \in [1, N_s]\}$ where $t^{(z)} < t^{(z+1)}, \forall z$. Then, the color of the ray can be computed as:

$$\hat{I}(\tilde{\boldsymbol{r}}_i) = \sum_{z=1}^{N_s-1} \omega_i^{(z)}\, \boldsymbol{c}\Big(\tilde{\boldsymbol{r}}_i\big(t^{(z)}\big), \tilde{\boldsymbol{r}}_i\big(t^{(z+1)}\big) - \tilde{\boldsymbol{r}}_i\big(t^{(z)}\big)\Big), \tag{4}$$

where $\boldsymbol{c}(\cdot)$ is a color function modeled by an MLP, which takes as input the point position $\tilde{\boldsymbol{r}}_i(t)$ and the viewing direction of the ray *at that point*, which is approximated with a forward difference. The weight $\omega_i^{(z)}$ is occlusion-aware and *unbiased* with respect to the object's surface (see our supplementary material for the proof), which is formulated based on the opaque density $\rho_i(\tilde{\boldsymbol{r}}_i(t))$ from Equation (3) as follows:

$$\omega_i^{(z)} = T_i^{(z)}\, \alpha_i^{(z)}, \text{ with } T_i^{(z)} = \prod_{\zeta=1}^{z-1}(1 - \alpha_i^{(\zeta)}), \text{ and} \tag{5}$$

$$\alpha_i^{(z)} = \max\left\{ \frac{\Phi_s\Big(f\big(\tilde{\boldsymbol{r}}_i(t^{(z)})\big)\Big) - \Phi_s\Big(f\big(\tilde{\boldsymbol{r}}_i(t^{(z+1)})\big)\Big)}{\Phi_s\Big(f\big(\tilde{\boldsymbol{r}}_i(t^{(z)})\big)\Big)}, 0 \right\}, \tag{6}$$

$$\alpha_i^{(z)} = 1 - \exp\left(- \int_{t^{(z)}}^{t^{(z+1)}} \rho(t)\mathrm{d}t \right). \tag{7}$$

Importantly, the discrete opacity $\alpha_i^{(z)}$ derivation [43] still applies in the case of a bent ray as replacing the constant viewing direction with $\frac{\mathrm{d}\tilde{\boldsymbol{r}}_i(t)}{\mathrm{d}t}$ does not affect the analysis.

In addition to the color, we can determine if a ray intersects the object by computing the sum of the weights:

$$\hat{S}(\tilde{\boldsymbol{r}}_i) = \sum_{z=1}^{N_s-1} \omega_i^{(z)}, \tag{8}$$

where $\hat{S}$ approaches 1 for a ray intersecting the object and otherwise $\hat{S}$ approaches 0.

Supervision. We supervise the dynamic scene representation by ℓ_1-distance between the color of each bent ray $\tilde{\boldsymbol{r}}_i^{(p)}$ and the corresponding ground-truth color $I_i^{(p)}$ of pixel p:

$$L_{\text{COL}} = \frac{1}{N_p} \sum_{p=1}^{N_p} \Big| \hat{I}\big(\tilde{\boldsymbol{r}}_i^{(p)}\big) - I_i^{(p)} \Big|, \tag{9}$$

where N_p is the number of pixels sampled from frame i. To more explicitly ensure that our approach solely focuses on reconstructing the foreground object, we also define a segmentation loss L_{SEG} as the binary cross entropy between the estimated segmentation $\hat{S}\big(\tilde{\boldsymbol{r}}_i^{(p)}\big)$ and the ground truth object segmentation $S_i^{(p)}$. Finally, we enforce f to be an SDF with the Eikonal loss defined as follows:

$$L_{\text{EIK}} = \frac{1}{N_p N_s} \sum_{p=1}^{N_p} \sum_{z=1}^{N_s} (|\nabla f(\tilde{\boldsymbol{r}}_i^{(p)}(t^{(z)}))| - 1)^2. \tag{10}$$

3.3. Optional Scene Flow Loss

So far, very large scene deformations remain a challenge for Ub4D since it can create erroneous multiple geometries in the canonical space to best explain the monocular observations. This is particularly noticeable for scenes containing large translations like *RootTrans* (see Figure 5-(b)). To resolve this, we accept an additional input in the form of a coarse and coherent per-frame geometric proxy. From these coarse 3D correspondences, we can compute an estimate of the scene flow, which can then be used to regularize the bending network. This greatly reduces the effect of duplicate geometries in the canonical space. *Note that the scope of this work is not how such coarse proxies are obtained, rather we focus on how these enable Ub4D to densely track larger scene deformations.*

Consider a function $\boldsymbol{m}_{i\rightarrow j} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ that returns the scene flow estimate at a point from frame i to j. The scene flow allows us to transform points from a frame i into any other frame j as $\boldsymbol{x}_j = \boldsymbol{x}_i + \boldsymbol{m}_{i\rightarrow j}(\boldsymbol{x}_i)$. Given that the bending network projects a point $\boldsymbol{x}_i$ in frame space into canonical space resulting in the point $\boldsymbol{x}_c$, it follows:

$$\begin{aligned} \boldsymbol{x}_c &= \boldsymbol{x}_i + \boldsymbol{b}_i(\boldsymbol{x}_i) = \boldsymbol{x}_j + \boldsymbol{b}_j(\boldsymbol{x}_j) \\ &= \boldsymbol{x}_i + \boldsymbol{m}_{i\rightarrow j}(\boldsymbol{x}_i) + \boldsymbol{b}_j\big(\boldsymbol{x}_i + \boldsymbol{m}_{i\rightarrow j}(\boldsymbol{x}_i)\big). \end{aligned} \tag{11}$$

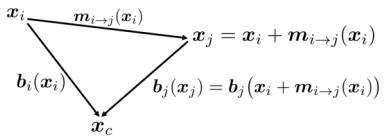

Figure 3. Graphical depiction of the relationship between the scene flow from frame i to j (*i.e.,* $\boldsymbol{m}_{i\to j}(\boldsymbol{x}_i)$) and the bending network projecting both points to the same canonical position $\boldsymbol{x}_c$.

Intuitively, this means that a point in frame i and its corresponding point in frame j determined through the scene flow $\boldsymbol{m}_{i\to j}(\boldsymbol{x}_i)$ should be mapped to the same point $\boldsymbol{x}_c$ in canonical space by the bending network (see Fig. 3). We can then formulate it as a loss for a set $\mathcal{X}$ of sampled points:

$$L_{\text{FLO}} = \frac{1}{|\mathcal{X}|} \sum_{\boldsymbol{x}\in\mathcal{X}} ||\boldsymbol{m}_{i\to j}(\boldsymbol{x}) + \boldsymbol{b}_j\big(\boldsymbol{x} + \boldsymbol{m}_{i\to j}(\boldsymbol{x})\big) - \boldsymbol{b}_i(\boldsymbol{x})||_2^2. \tag{12}$$

The scene flow from the proxies can only be exactly computed at the proxy vertices $\boldsymbol{v}_i^{(k)}$ but our implicit surface representation can be evaluated at any point in 3D space. Thus, we extrapolate this scene flow to any point in $\mathbb{R}^3$ with a convex combination over the vertices using a kernel function depending on the distance to the vertices inspired by the spatial weighting approach in bilateral filtering [38]:

$$\boldsymbol{m}'_{i\to j}(\boldsymbol{x}) = \frac{\sum_{k=1}^{N_v} w_{\lambda_1}\big(||\boldsymbol{x} - \boldsymbol{v}_i^{(k)}||_2\big)\,\big(\boldsymbol{v}_j^{(k)} - \boldsymbol{v}_i^{(k)}\big)}{\sum_{k=1}^{N_v} w_{\lambda_1}\big(||\boldsymbol{x} - \boldsymbol{v}_i^{(k)}||_2\big)}, \tag{13}$$

where $w_{\lambda_1}(x) = e^{-\lambda_1 x^2}$ is a kernel function with a parameter λ_1 affecting the weighting of vertex flow estimates. Additionally, we add an attenuation term, so that the scene flow falls off as the distance to the nearest vertex increases:

$$\boldsymbol{m}_{i\to j}(\boldsymbol{x}) = w_{\lambda_2}\Big(\min_{k=1}^{N_v} ||\boldsymbol{x} - \boldsymbol{v}_i^{(k)}||_2\Big)\,\boldsymbol{m}'_{i\to j}(\boldsymbol{x}) \tag{14}$$

where $w_{\lambda_2}(x) = e^{-\lambda_2 x^2}$ is a kernel function with λ_2 as a scale parameter defining the extent of the kernel.

3.4. Surface Extraction

To convert our deforming and implicit scene representation into an explicit geometry, we use the Marching Cubes algorithm [19] with a threshold of 0. For points sampled in frame i, we transform them from the frame space into the canonical space, *i.e.,* $\boldsymbol{x}_c = \boldsymbol{x}_i + \boldsymbol{b}_i(\boldsymbol{x}_i)$, where $\boldsymbol{x}_i$ is a point sampled for marching cubes and $\boldsymbol{x}_c$ is the canonical space point at which we then evaluate the SDF. We restrict the selection of frame march points to the camera frustum of the given frame since any space not seen in that frame is unconstrained by our reconstruction losses and may contain aberrant geometry.

4. Experimental Results

We first introduce the datasets we are using for evaluation as well as the evaluation metrics. Next, we compare our method to previous works on monocular 4D scene reconstruction (Sec. 4.1). We validate the design choice of using an SDF network rather than a density network (Sec. 4.2). Finally, we ablate important design choices (Sec. 4.3) and show more qualitative results on real world data (Sec. 4.4). All tests were performed using a single NVIDIA Quadro RTX 8000 with 48 GB RAM.

Dataset. We aim at reconstructing the *full* deforming geometry and, thus, the monocular capture requires sufficient camera motion around the dynamic object to observe every part at least once. However, we found that existing datasets either capture static scenes with a circulating camera path around the object or dynamic scenes with very limited camera motion. Therefore, we create our own dataset of dynamic objects with sufficient camera motion, which contains synthetic scenes for quantitative evaluations and real scenes for qualitative results.

For the synthetic evaluation, we create two scenes in Blender [6] showing a deforming cactus, referred to as *Cactus*, and a moving human, referred to as *RootTrans*. Each of the scenes has an image resolution of 1024×1024 and is 150 frames long. We define a moving camera viewing the dynamic object and provide the camera parameters as input to our method. To generate the optional proxy geometries, which are used in our proposed scene flow loss for capturing large deformation, we leverage a human capture method [26] for the *RootTrans* sequence (further details are included in our supplementary material). For the *Cactus* sequence, we use a highly downsampled version of the ground-truth geometry as a coarse proxy. A visualization of the proxies is shown in Figure 4.

For the evaluation of our method on real data, we capture two sequences: one of a moving human, called the *Humanoid* sequence, and one of a deforming cactus toy, called *RealCactus*. We capture these sequences at resolutions of 960×1280 and 1080×1920, respectively, with a mobile phone camera. Again each sequence contains around 150 frames. To obtain the camera parameters, we use the rigid Structure from Motion (SfM) software COLMAP [34, 35]. As with the *RootTrans* synthetic sequence, we generate proxy geometries for the *Humanoid* sequence using the same human capture method [26]. However, unlike the *RootTrans* sequence, we only input the *joint positions* (i.e. 12 vertices) as the proxy, rather than the full posed SMPL-X [26] model. This shows that our proxy need not include any information about the location of the surface. To demonstrate the *optional* nature of the scene flow loss, the *RealCactus* sequence does not use any proxy geometry at all. For the foreground masks, we manually labeled a few frames and then trained a segmentation network, based on

	Method	CD (↓)	E2G (↓)	G2E (↓)
Cactus	D-NeRF [29]	[113.38]	-	7.99
	NR-NeRF [40]	[96.96]	-	9.89
	LASR [45]	20.23	12.09	8.14
	ViSER [46]	14.34	7.93	6.41
	N-NRSfM [37]	[102.00]	5.74	-
	DDD [50]	34.71	6.98	27.72
	Ub4D (ours)	**3.06**	2.42	0.64
	Ub4D after ICP	2.71	2.24	0.46
RootTrans	D-NeRF [29]	[23.50]	-	8.43
	NR-NeRF [40]	[1.94]	-	0.33
	LASR [45]	0.39	0.08	0.31
	ViSER [46]	0.37	0.20	0.17
	N-NRSfM [37]	[0.38]	0.09	-
	DDD [50]	0.26	0.10	0.16
	Ub4D (ours)	**0.23**	0.14	0.09
	Ub4D after ICP	0.03	0.02	0.02

Table 1. **Quantitative comparison to previous work.** We include the unidirectional breakdown of the CD as well, since some methods produce geometries for which the CD is unfair as a metric (these are denoted by placing the CD in square brackets and omitting the unfair metric). See our supplement for further discussion.

the UNet architecture [30], on those labeled frames, which then provides masks for all frames in a semi-automated fashion.

For additional evaluation of our method, we leverage the *Lego* object[1] made available by Mildenhall *et al.* [20], which we animate over time by lifting the boom and tilting the bucket (see Fig. 5-(a)), to obtain a dynamic scene. Further, we defined a monocular camera path for 150 frames, rendered monocular images and masks at a resolution of 800×800, and extracted the camera extrinsics and intrinsics. This scene does not use 3D proxy for our method.

Evaluation Metrics. To quantitatively compare Ub4D to the state-of-the-art methods, we compute the Chamfer distance (CD) between the estimated geometry and the ground-truth geometry. For a more fair comparison with previous monocular non-rigid 3D reconstruction methods, we break the CD down into its two components: measuring from estimate to ground-truth (*E2G*) and from ground-truth to estimate (*G2E*). The reported numbers are in relative distance units since synthetic scenes do not have an interpretable physical scale. NR-NeRF [40] and D-NeRF [29] are given the same camera parameters used by our method. Some methods either assume a fixed camera [50] or predict the camera poses [37,45,46]. For these cases, we apply ICP [2] to rigidly align their meshes with the ground truth before computing metrics.

[1]Released under CC-BY-3.0 and modifications are made. Originally created by Blend Swap user Heinzelnisse.

	Comparison	CD (↓)	E2G (↓)	G2E (↓)
Cactus	w/o L_{FLO}	8.32 †	4.67 †	3.65 †
	w/o L_{EIK}	5.47	3.58	1.89
	w/o L_{FLO}, L_{NBR}, & L_{DIV}	5.34 †	3.27 †	2.07 †
	Ub4D (Ours)	**3.06**	2.42	0.64
RootTrans	w/o L_{FLO}	9.42	4.83	4.59
	w/o L_{EIK}	0.29	0.16	0.13
	w/o L_{FLO}, L_{NBR}, & L_{DIV}	4.30 †	3.17 †	1.13 †
	Ub4D (Ours)	**0.23**	0.14	0.09

Table 2. **Quantitative ablation study.** "†" denotes frames that do not produce any geometry (due to frustum culling). Note that our full method provides the best result for both scenes.

4.1. Quantitative Comparison

We compare our method to NR-NeRF [40], D-NeRF [29], N-NRSfM [37], DDD [50], LASR [45], and ViSER [46]. NR-NeRF and D-NeRF are novel view synthesis methods that permit extracting geometry by using marching cubes on their density networks with a threshold. N-NRSfM is a Non-Rigid Structure-from-Motion (NRSfM) method, which uses an auto-decoder to deform a mean shape based on a learned per-frame latent representation. DDD is template-based; it deforms the template to minimize an energy formulation. For DDD, we provide the first frame's ground-truth mesh as a template. Both LASR and ViSER do not require a template and recover a rigged mesh that is animated over the image sequence. Several other 4D reconstruction techniques with source code available online, such as Shimada *et al.* [36] or Ngo *et al.* [21], do not work under our assumptions; so, we do not include them. For each related method, we follow the original papers to set the hyper-parameters.

The quantitative results on the synthetic sequences are reported in Table 1 and a qualitative comparison is shown in Fig. 4. Ub4D outperforms the state-of-the-arts both quantitatively and qualitatively. We found that prior work struggles with large scale deformations resulting in overly noisy results [29], [40], experiences tracking errors [50], has a limited resolution [45], [46], or only reconstructs the visible geometry [37] while ours accurately captures the large deformations of the entire geometry. Also note that although we rigidly align the results for other methods with the ground truth, our method still achieves the most accurate results. For completeness, we also report our results after ICP.

4.2. Comparison to Volume-based Representations

Like some previous works [43,48], our method leverages an SDF representation to model the surface of the object. An alternative approach is predicting volume density with a network [20, 29, 40]. While a volume density representa-

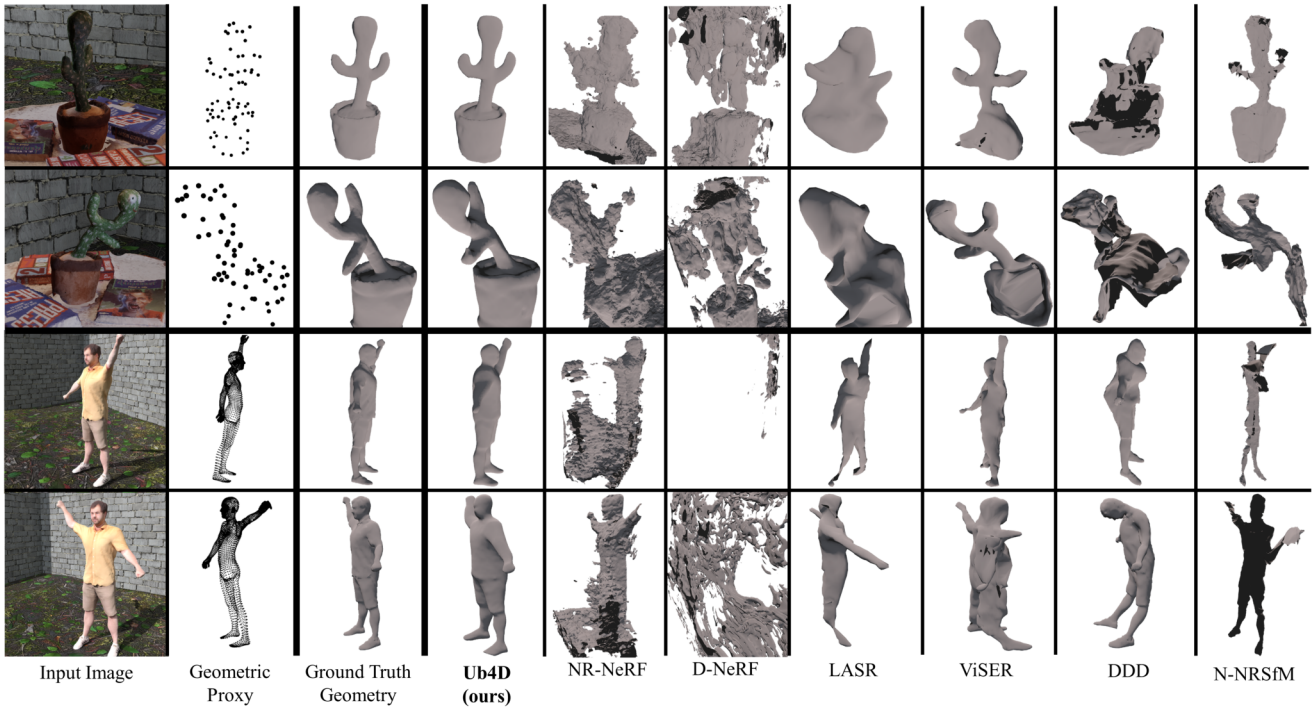

Figure 4. Qualitative comparison with our synthetic sequences rendered from novel views. Note that competing methods struggle with reconstructing the dense and deforming surface, while Ub4D captures the large scale deformations as well as medium scale details.

tion has been used for the problem of novel view synthesis, extracting a surface from such a density representation with marching cubes results in noisy and inaccurate geometry as demonstrated in Fig. 4 and further in Fig. 5-(a) where we qualitatively compare to NR-NeRF [40]. In contrast, an SDF representation removes the need to determine a threshold when extracting the explicit geometry and must add a zero crossing, *i.e.,* a surface, in order to satisfy the reconstruction losses. Further, this example shows the limited ability of NR-NeRF to model large deformations as they penalize the absolute offset length, which our neighbouring frame regularization allows us to handle.

4.3. Ablation Study

We validate our design decisions through an ablation study on the *Cactus* and *RootTrans* sequences and report the metrics in Table 2. Our full supervision consists of six loss terms: L_{COL}, L_{SEG}, L_{EIK}, L_{FLO}, L_{NBR} and L_{DIV}. We compare the full method to removing the terms: 1) L_{FLO}, which is our novel flow loss, 2) L_{EIK}, which directly regularizes the SDF and color network and indirectly regularizes the bending network and 3) L_{FLO}, L_{NBR}, and L_{DIV}, which are all direct bending network regularizers. Most importantly, the full combination of losses provides the best result validating the contribution of each term.

Concerning 1), our flow loss especially helps for the large root translation and arm motion of the *RootTrans* sequence. Without using this loss, multiple different geometries are synthesized, which fit the reconstruction losses. Then, the bending network can "switch" between the different copies throughout the sequence. This results in overfitting to the camera pose by exploiting monocular depth ambiguities to generate geometry that is not seen in other views. Figure 5-(b) shows this overfitting to the camera

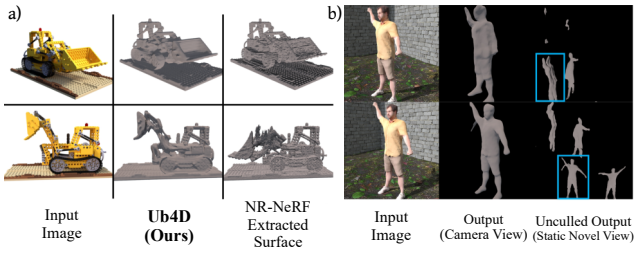

Figure 5. (a) Comparison of Ub4D using an SDF scene representation (without scene flow loss) to a density-based scene representation, called NR-NeRF [40]. To generate surfaces for NR-NeRF, we apply marching cubes [19] with a threshold of 50. The density-based representation leads to an overall noisier surface compared to our approach. We also penalize bending using neighbouring frame offsets allowing Ub4D to accurately reconstruct large deformations. (b) Qualitative ablation of the scene flow loss (L_{FLO}) on the *RootTrans* sequence. Right column shows the scene from a static novel view with the geometry in the camera frustum highlighted with a blue box. Note that without the proposed scene flow loss using proxy geometry, Ub4D can produce multiple distinct copies of the character at different scales by exploiting the monocular depth ambiguity.

Figure 6. Qualitative results of our method for a non-humanoid object, *RealCactus*, left, and a human character, *Humanoid*, right. The scene flow loss is only used for the human character and only uses a sparse skeleton geometric proxy (12 vertices). This shows that the geometric proxy need not be dense or provide any information about the surface. Note that the recovered geometry nicely overlays onto the input image but also looks plausible from a novel 3D viewpoint.

pose with multiple distinct geometries being used over the sequence to satisfy the reconstruction losses.

Regarding 2), we found that not using L_{EIK} leads to overall noisier surfaces and thus the quality is reduced. Finally concerning 3), without any explicit regularization of the bending network, the deformations can be almost arbitrary again leading to overfitting to individual frames by violating 3D consistency resulting in a reduced accuracy. We even observed that the network was not able to produce any geometry for some frames, which validates the necessity of explicit regularization of the bending network.

4.4. Qualitative Results on Real Word Scenes

We next demonstrate that Ub4D works well on real-world scenes. Fig. 6 visualizes our *RealCactus* sequence depicting a dancing cactus toy and the *Humanoid* sequence where a person is moving their arms and legs. Although in both cases the dynamic scenes contain large deformations, Ub4D robustly and accurately reconstructs individual frame geometries, which also contain medium frequency details.

5. Discussion and Possible Extensions

Ub4D significantly outperforms all competing methods in our evaluations, both numerically and qualitatively. We hypothesise that it is partially due to its shape completion property. In fact, we found that none of the existing methods can deal with captures that include object motion *and* severe camera motions while our method leverages such recording conditions to its benefit, inspired by classical non-rigid structure from motion algorithms. In the case of severe scene deformations Ub4D can leverage a geometric proxy.Note that our scene flow loss is versatile: The proxy can be either a full estimated mesh or even just a few points (see *Humanoid* results Fig. 6); as long as it *roughly* describes the deformation of the scene, the model is guided towards a better local minimum in the reconstruction losses. Future work involves exploring this direction further with the main question being: How sparse can the proxy be and could it even be a 2D entity in the image plane? Along these lines, we see multiple avenues for future research, including tracking a generic proxy along with learning the SDF and using 2D image features for initializing a sparse proxy.

6. Conclusion

We presented, Ub4D, a method of a new class for 3D reconstruction of deformable scenes from a single RGB camera. It represents the scene as a learned static canonical volume with an implicit surface. A bending network warping the frames into this canonical volume accounts for the scene deformation. Our optional scene flow loss improves the reconstruction accuracy and robustness in the case of large deformations given a coarse proxy geometry. The qualitative and quantitative comparisons to different method types show that our approach is a clear step towards dense and deformable tracking of general and largely deforming scenes using a single RGB camera.

References

[1] Mohammad D. Ansari, Vladislav Golyanik, and Didier Stricker. Scalable dense monocular surface reconstruction. In *International Conference on 3D Vision (3DV)*, 2017. 1, 2

[2] Paul J. Besl and Neil D. McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: Control Paradigms and Data Structures*, 1992. 6

[3] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J. Black. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *European conference on computer vision (ECCV)*, 2016. 2

[4] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. In *Computer Vision and Pattern Recognition (CVPR)*, 2000. 1, 2

[5] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo radiance fields (srf): Learning view synthesis from sparse views of novel scenes. In *Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[6] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, 2018. 5

[7] David Fuentes-Jimenez, Daniel Pizarro, David Casillas-Perez, Toby Collins, and Adrien Bartoli. Texture-generic deep shape-from-template. *IEEE Access*, 9:75211–75230, 2021. 1, 2

[8] Ravi Garg, Anastasios Roussos, and Lourdes Agapito. Dense variational reconstruction of non-rigid surfaces from monocular video. In *Computer Vision and Pattern Recognition (CVPR)*, 2013. 1, 2

[9] Marc Habermann, Weipeng Xu, Helge Rhodin, Michael Zollhoefer, Gerard Pons-Moll, and Christian Theobalt. Nrst: Non-rigid surface tracking from monocular video. In *German Conference on Pattern Recognition (GCPR)*, 2018. 2

[10] Marc Habermann, Weipeng Xu, Michael Zollhöfer, Gerard Pons-Moll, and Christian Theobalt. Livecap: Real-time human performance capture from monocular video. *ACM Transactions on Graphics (TOG)*, 38(2):14:1–14:17, 2019. 2

[11] Navami Kairanda, Edgar Tretschk, Mohamed Elgharib, Christian Theobalt, and Vladislav Golyanik. ϕ-sft: Shape-from-template with a physics-based deformation model. In *Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[12] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[13] Chen Kong and Simon Lucey. Deep non-rigid structure from motion. In *International Conference on Computer Vision (ICCV)*, 2019. 1, 2

[14] SEE COVER LETTER. See cover letter, SEE COVER LETTER. 2

[15] Xueting Li, Sifei Liu, Shalini De Mello, Kihwan Kim, Xiaolong Wang, Ming-Hsuan Yang, and Jan Kautz. Online adaptation for consistent mesh reconstruction in the wild. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[16] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[17] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[18] Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. Neural actor: Neural free-view synthesis of human actors with pose control. In *ACM Transactions on Graphics (TOG)*, 2021. 2

[19] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH*, 21(4):163–169, 1987. 5, 7

[20] 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 *European Conference on Computer Vision (ECCV)*, 2020. 2, 6

[21] Dat Tien Ngo, Sanghyuk Park, Anne Jorstad, Alberto Crivellaro, Chang D. Yoo, and Pascal Fua. Dense image registration and deformable surface reconstruction in presence of occlusions and minimal texture. In *International Conference on Computer Vision (ICCV)*, 2015. 1, 2, 6

[22] David Novotny, Nikhila Ravi, Benjamin Graham, Natalia Neverova, and Andrea Vedaldi. C3dpo: Canonical 3d pose networks for non-rigid structure from motion. In *International Conference on Computer Vision (ICCV)*, 2019. 1, 2

[23] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *International Conference on Computer Vision (ICCV)*, 2021. 2

[24] Shaifali Parashar, Daniel Pizarro, and Adrien Bartoli. Isometric non-rigid shape-from-motion with riemannian geometry solved in linear time. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(10):2442–2454, 2018. 1, 2

[25] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *International Conference on Computer Vision (ICCV)*, 2021. 2

[26] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 5

[27] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *International Conference on Computer Vision (ICCV)*, 2021. 2

[28] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[29] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1, 2, 6

[30] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015. 6

[31] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *International Conference on Computer Vision (ICCV)*, 2019. 1

[32] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In *Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[33] Mathieu Salzmann, Julien Pilet, Slobodan Ilic, and Pascal Fua. Surface deformation models for nonrigid 3d shape recovery. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 29(8):1481–1487, 2007. 1, 2

[34] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[35] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 5

[36] Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Didier Stricker. Ismo-gan: Adversarial learning for monocular non-rigid 3d reconstruction. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019. 1, 2, 6

[37] Vikramjit Sidhu, Edgar Tretschk, Vladislav Golyanik, Antonio Agudo, and Christian Theobalt. Neural dense non-rigid structure from motion with latent space constraints. In *European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 6

[38] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *International Conference on Computer vision (ICCV)*, 1998. 5

[39] Lorenzo Torresani, Aaron Hertzmann, and Chris Bregler. Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(5):878–892, 2008. 1, 2

[40] 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 dynamic scene from monocular video. In *International Conference on Computer Vision (ICCV)*, 2021. 1, 2, 3, 4, 6, 7

[41] Chaoyang Wang and Simon Lucey. Paul: Procrustean autoencoder for unsupervised lifting. In *Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[42] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[43] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 1, 2, 4, 6

[44] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[45] 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 *Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 6

[46] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Ce Liu, and Deva Ramanan. Viser: Video-specific surface embeddings for articulated 3d shape reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 6

[47] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 4

[48] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Ronen Basri, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2, 4, 6

[49] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *International Conference on Computer Vision (ICCV)*, 2021. 2

[50] Rui Yu, Chris Russell, Neill DF Campbell, and Lourdes Agapito. Direct, dense, and deformable: Template-based non-rigid 3d reconstruction from rgb video. In *International Conference on Computer Vision (ICCV)*, 2015. 1, 2, 6