

Self-supervised Neural Articulated Shape and Appearance Models

Fangyin Wei^{1*} Rohan Chabra² Lingni Ma² Christoph Lassner² Michael Zollhoefer²
Szymon Rusinkiewicz¹ Chris Sweeney² Richard Newcombe² Mira Slavcheva²

¹Princeton University ²Reality Labs Research

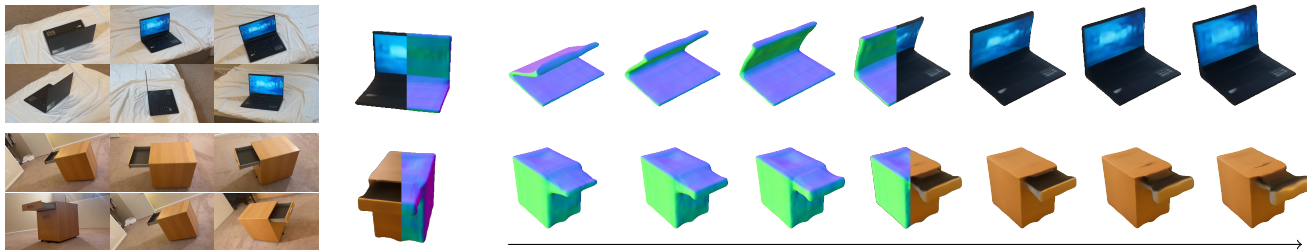


Figure 1. Our self-supervised method learns the shape and appearance of articulated object classes. After training from multi-view synthetic images of different states of object instances, our model can reconstruct and animate objects from static real-world images. *Left*: input real views of a static object. *Middle*: 3D reconstruction with shape and color. *Right*: animation driven by the learned articulation codes.

Abstract

Learning geometry, motion, and appearance priors of object classes is important for the solution of a large variety of computer vision problems. While the majority of approaches has focused on static objects, dynamic objects, especially with controllable articulation, are less explored. We propose a novel approach for learning a representation of the geometry, appearance, and motion of a class of articulated objects given only a set of color images as input. In a self-supervised manner, our novel representation learns shape, appearance, and articulation codes that enable independent control of these semantic dimensions.

Our model is trained end-to-end without requiring any articulation annotations. Experiments show that our approach performs well for different joint types, such as revolute and prismatic joints, as well as different combinations of these joints. Compared to state of the art that uses direct 3D supervision and does not output appearance, we recover more faithful geometry and appearance from 2D observations only. In addition, our representation enables a large variety of applications, such as few-shot reconstruction, the generation of novel articulations, and novel view-synthesis. Project page: <https://weify627.github.io/nasam/>.

1. Introduction

Reconstructing articulated 3D objects from image observations in terms of their underlying geometry, kinemat-

ics, and appearance is one of the fundamental problems of computer vision with many important applications, *e.g.* in robotics and augmented/virtual reality. This inverse graphics problem is highly challenging and of underconstrained nature, since image formation, *i.e.*, mapping from the 3D world to discrete 2D pixel measurements, tightly entangles all visible properties of an object—and finding non-visible properties, such as kinematics, requires additional information such as information over time.

Most approaches that tackle this inverse problem rely on object/class-specific priors learned from large datasets with available 3D ground truth, which are challenging and expensive to collect. The learned manifold of shape, appearance, and motion is often encoded via a low-dimensional latent space. In the devised approaches, this learned low-dimensional prior is then used to better constrain the inverse reconstruction problem.

The majority of previous techniques in the literature has focused on reconstructing classes of static objects; dynamic objects, especially with controllable articulation, are less explored. For example, occupancy networks condition the decision boundary of a neural classifier on a shape code to represent a class of static objects [33]. Approaches such as DeepSDF follow a similar principle, but employ a learned continuous signed distance field (SDF) [42] to model the object’s surface implicitly as the zero level-set of a coordinate-based neural network. DISN [61] further improves this technique and can recover more details. All the mentioned approaches require dense 3D ground truth geometry for training and do not model object appearance. One exception is IDR [72] which employs inverse differ-

*Work begun during internship at Reality Labs Research.

entiable rendering to reconstruct the shape (using an SDF) and view-dependent appearance of a single object, but this approach does not generalize to an entire object class.

Prior work on articulated deformations heavily focuses on humans and animals [9,22,31,35,40,50,69,74] due to the availability of large datasets and readily available (learned) priors. One exception is the A-SDF [37] technique which is focused on general articulated objects. It learns separate codes for shape and articulation and employs an SDF for representing the objects. This approach learns a geometry prior across a class of articulated objects but does not jointly learn an appearance prior. Additionally, it requires dense 3D ground truth for training. We provide comparisons with A-SDF in our experiments.

Looking at these related works in context raises the question: Is it possible to jointly learn a prior over the 3D geometry, kinematics, and appearance over an entire class of articulated objects from only photometric 2D observations without requiring access to 3D ground truth?

We propose a novel approach for learning the geometry, kinematics, and appearance manifold of a class of articulated objects given only a set of color images as input. Our novel 3D representation of articulated objects is learned in a self-supervised manner from only color observations without the need for explicit geometry supervision. It enables independent control of the learned semantic dimensions. Our model is trained end-to-end without requiring any articulation annotations. Experiments show that our approach performs well on both of the most widespread joint types: revolute and prismatic joints, as well as combinations thereof. We outperform the state of the art, even though these related approaches require access to ground truth geometry for explicit geometry supervision. In addition, our approach handles a larger variety of joint types than A-SDF [37]. Furthermore, our representation enables various applications, such as few-shot reconstruction, the generation of novel articulations, and novel view-synthesis. In summary, our contributions are:

- A novel approach that learns a representation of the geometry, appearance, and kinematics of a class of articulated objects with only a set of color images as input.
- We introduce an embedding space for geometry, kinematics, and view-dependent appearance that enables a large variety of applications, such as the generation of new articulations and novel view synthesis.
- Our model, trained only on synthetic data, enables few-shot reconstruction of real-world articulated objects via fine-tuning, as shown in Fig. 1.

2. Related Work

Articulated Object Modeling. In robotics, research on articulated objects focuses on kinematic models [1,21,48].

In the vision community, prior work on articulated deformations heavily focuses on humans and animals [9,22,31,35,40,50,69,74]. Less explored is how to represent general objects with piece-wise rigidity. Following the success of neural implicit representations, A-SDF [37] extends DeepSDF [42] with separate shape and articulation codes to model category-level articulation. By adding joint angles to the shape code, A-SDF learns a mapping to the corresponding deformed shape. Instead of using known joint angles and dense 3D supervision as in A-SDF, we learn the articulation codes without the ground-truth labeling and only from images. Li et al., [25] propose the normalized articulated object coordinate space (NAOCS) as a canonical representation for category-level articulated objects. This idea is further explored by CAPTRA [63] to track object articulation from point clouds, and adopted by StrobeNet [73] to reconstruct articulated objects by first aggregating NAOCS predictions from multi-view color observations. However, the reconstruction only provides geometry. Recently, LASR [70] proposes a template-free approach to reconstructing articulated shapes from monocular video. The algorithm jointly estimates the rest pose, skinning, articulation, and camera intrinsics by solving an inverse graphics problem resulting in coarse animatable meshes.

Deformation Fields for Shape Reconstruction. Shape deformation deals with deforming a shape to best fit a set of observations. DynamicFusion [38] relied on local depth correspondences, follow-up methods used sparse SIFT features [20], dense color tracking [16] or dense SDF alignment [54,55]. Such methods are subject to failure under fast motion due to the use of handcrafted functions for maintaining correspondences. Recently, the performance of non-rigid tracking has been improved by data-driven approaches with learned correspondences [4,5,26]. Lately, there has been an exploration of neural generative models for shape deformation. DIF [10] represents shapes by a template implicit field shared across the category, together with a 3D deformation field and a correction field dedicated to each shape instance. FiG-NeRF [67] approximates neural radiance fields of objects and simultaneously performs foreground/background separation. NPMs [41] is a neural parametric model that learns latent shape and pose spaces to model 3D deformable shapes. Unlike most of the above methods, we approximate geometry and view-dependent appearance of the shape. without any 3D supervision.

Neural Representations for Geometry. Neural scene representations have been found compact and powerful to model the geometry and motion of objects [24,43]. Implicit fields [7] use an implicit coordinate-based function and a latent code to model multiple object classes. Local Deep Implicit Functions [14] decompose space into a structured set of learned implicit functions to represent deformable shapes. Occupancy networks [33] use a local neural classi-

fier to represent objects. Approaches such as DeepSDF [42] follow a similar idea, but employ an SDF. DISN [61] further improves this technique and can recover more details. Local implicit grid representations [6, 8] decompose a scene into local parts for which it learns implicit representations. All of these methods only model the geometry of rigid objects.

Neural Representations for Appearance. Neural Radiance Fields (NeRF) model the appearance of static scenes via a coordinate-based scene representation [36]. Scene Representation Networks [53] map world coordinates to a feature representation that can be translated to a rendering. Differentiable Volumetric Rendering [39] predicts a texture field. Proxy-geometry, such as spheres or points can be used to speed up the rendering process [51, 60]. Point-based representations have been explored independently for view synthesis as well [2, 23, 64]. Neural Volumes breaks down the space into separate volumes with their own neural representations [29] and can model dynamic scenes. For an in-depth discussion of recent neural rendering approaches, we refer to a recent survey [57].

Neural Representations for Motion. A hallmark feature of dynamic scenes is that they can be analyzed using flow fields, and these in turn can be used to represent them [11, 27]. There are several works that build on top of neural radiance fields to capture scenes in motion [3, 13, 24, 44, 45, 47, 49, 65]. However, they do not allow to control the scene rendering. Tretschk et al. [58] allow to strengthen or weaken foreground motion. All of the aforementioned works in this paragraph do not allow to control the resulting reconstruction. Most prior work on controllable representations is around humans, for facial avatars [12, 17, 30, 62] or human bodies [19, 28, 40, 46, 56, 68]. LASR [70] is a very general method that creates meshes for arbitrary objects. It then provides a rigged model with an estimated skeleton to animate the object, however with very coarse results. D-NeRF [49] approximates the radiance field of a deforming shape by using time as an input to the system but only works for a single scene.

3. Method

Our goal is to build a representation that models the geometry and appearance of a class of articulated objects from RGB images without geometry priors. This representation must enable reconstructing unseen shapes and generating new articulations. To this end, we utilize a differentiable rendering system with implicit neural representations and learn a category-level embedding space with disentangled shape, appearance, and articulation representations. To enforce the disentanglement, different articulations of the same instance share the same shape code. The geometry is predicted by deforming a canonical shape conditioned on a learned articulation code. Without any joint annotations,

the model is able to learn a continuous space for articulation that allows for generating new articulations.

We define an ‘articulated object’ as an object which consists of several rigid parts connected by joints. We define further an *articulation* of an articulated object as a specific state of its joints (*e.g.*, a laptop with joint angles 40° and 90° are two distinct articulations). An object with a specific articulation is modeled by a representation set (θ, ϕ, ψ) where $\theta \in \mathbb{R}^m, \phi \in \mathbb{R}^n, \psi \in \mathbb{R}^q$ are the code for geometry, appearance, and articulation, respectively. In this section, we first review our backbone differentiable renderer and introduce a category-level embedding space (Sec. 3.1). Then we describe how to enforce disentanglement and the deformation field prediction (Sec. 3.2). In Sec. 3.3, we summarize the entire framework for training and inference.

3.1. Differentiable Renderer with Category Priors

We opt for Implicit Differentiable Renderer (IDR) [72] as the backbone differentiable renderer. While it originally works for a single object, we extend it to learn category-level geometry and appearance embedding. IDR is an end-to-end neural system that can learn 3D geometry, appearance, and camera extrinsics from masked 2D images and noisy camera pose initializations. There are three unknowns in IDR [72]: geometry with learnable parameters $\Theta \in \mathbb{R}^r$, appearance with learnable parameters $\Phi \in \mathbb{R}^s$, and camera extrinsics $\tau \in \mathbb{R}^k$. The geometry is represented as

$$\mathcal{S}_\Theta = \{x \in \mathbb{R}^3 | f(x; \Theta) = 0\}, \quad (1)$$

where f models the signed distance function (SDF) to its zero level set \mathcal{S}_Θ (*i.e.* the object’s surface) [72]. Given a pixel indexed by p , the rendered color of the pixel is

$$L_p(\Theta, \Phi, \tau) = M(\hat{x}_p, \hat{n}_p, v_p; \Phi), \quad (2)$$

where L_p is the surface light field radiance and $\hat{x}_p = \hat{x}_p(\Theta, \tau)$ denotes the first intersection of the ray R_p and the surface \mathcal{S}_Θ with the corresponding surface normal $\hat{n}_p = \hat{n}_p(\Theta)$ and the viewing direction v_p . Both f and M are approximated by MLPs.

In the original IDR [72], each trained model only works for a single scene or object instance. We extend it to work across an entire class of objects by introducing additional embedding space. For each object instance i from a class, we learn a geometry code $\theta_i \in \mathbb{R}^m$ and an appearance code $\phi_i \in \mathbb{R}^n$. During the learning process, all objects from the same category share the same geometry (Θ) and appearance (Φ) parameters. The new geometry and light field functions for object i become:

$$\mathcal{S}_\Theta = \{x \in \mathbb{R}^3 | f(x, \theta_i; \Theta) = 0\}, \quad (3)$$

$$L_p(\Theta, \Phi, \tau, \theta_i, \phi_i) = M(\hat{x}_p, \hat{n}_p, v_p, \phi_i; \Phi). \quad (4)$$

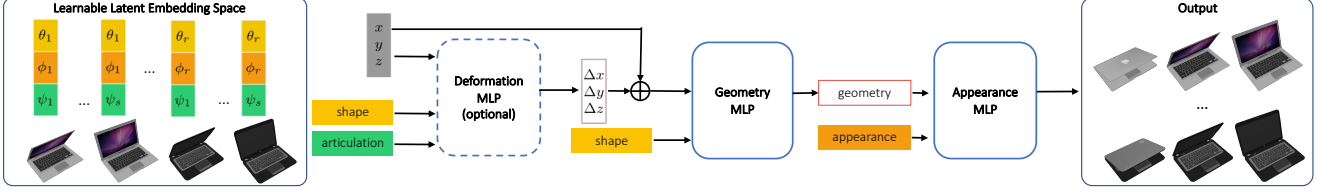


Figure 2. **Framework overview.** Each object i with articulation state j is represented as $(\theta_i, \phi_i, \psi_j)$, where each parameter encodes geometry, appearance, and articulation, respectively. A deformation network takes in the shape and articulation code in addition to query 3D locations (computed given camera parameters) to predict the displacement of the location. The displaced locations and the shape code pass through a geometry network to predict the geometry of the canonical pose shape. In the case of no deformation MLP, the geometry network directly takes in the 3D location, shape code, and articulation code. For the predicted geometry and given appearance code, an appearance network outputs an image according to input camera parameters.

Leveraging the category-level prior embedded in Θ and Φ allows to reconstruct unseen objects from the same category by recovering only their geometry and appearance codes.

3.2. Code Sharing and Deformation Field

To encode the articulation, we further introduce an articulation code. For a category with M training objects and N sampled articulation states for each object, let shape \mathcal{X}_{ij} denote an articulated object i from a specific category with articulation state j . We jointly learn the representation $(\theta_{ij}, \phi_{ij}, \psi_{ij})$. Note that when articulations across different objects are aligned, we can further enforce disentanglement by having all objects from the same category share the same set of articulation codes and all articulation states of the same object share the same object code. Therefore, the representation for shape \mathcal{X}_{ij} becomes $(\theta_i, \phi_i, \psi_j)$, as illustrated in Fig. 2.

One observation is that geometry change happens both when articulating the same object and between different object identities. Learning to generate shapes by simultaneously processing the object identity and articulation state information may cause unwanted interference. Therefore, we split the shape prediction module into two parts: a geometry network \mathcal{S} with parameters $\Theta \in \mathbb{R}^r$ and an (optional) deformation network \mathcal{D} with parameters $\Psi \in \mathbb{R}^t$. The former predicts the geometry of a canonical shape for each object given its shape code and is articulation-invariant. Conditional on a shape and articulation code that describes an articulated shape, the deformation network predicts a displacement of the query point to transform the query into the canonical space [44, 45, 58]. So the new geometry prediction flow is

$$x' = x + \mathcal{D}_{\Psi}(x, \theta_i, \psi_j; \Psi), \quad (5)$$

$$\mathcal{S}_{\Theta} = \{x' \in \mathbb{R}^3 | f(x', \theta_i; \Theta) = 0\}. \quad (6)$$

This separation of articulation prediction from the canonical shape helps further disentangle object identity and articulation state.

3.3. Training and Inference

During training, given camera intrinsic and extrinsic parameters and masked multi-view images, the model is trained to optimize both the latent embeddings $(\theta_i, \phi_i, \psi_j)$ and network weights (Θ, Φ) . As shown in Fig. 2, each object i with articulation state j is represented as $(\theta_i, \phi_i, \psi_j)$, where each parameter encodes geometry, appearance, and articulation, respectively. A deformation network takes in the shape and articulation code in addition to query 3D locations (computed given camera parameters) to predict the displacement of the location. The displaced locations and the shape code pass through a geometry network to predict the geometry of the canonical pose shape. For the predicted geometry and given appearance code, an appearance network outputs an image according to input camera parameters. Let $I_p \in [0, 1]^3, O_p \in \{0, 1\}$ be the RGB and mask values respectively for a pixel p in an image taken with camera $c_p(\tau)$ and direction $v_p(\tau)$ ($p \in P$ indexes all pixels in the input collection of images), and τ represents the parameters of all the cameras in scene. The overall loss function has the form:

$$\mathcal{L}(\Theta, \Phi, \tau, \{\theta_i\}, \{\phi_i\}, \{\psi_j\}) = L_{RGB} + \rho L_{mask} + \lambda L_E + \beta L_{code}. \quad (7)$$

We train on mini-batches of $P \subset \bar{P}$ pixels sampled from one view of shape \mathcal{X}_{ij} following IDR [72]. The RGB loss is computed over regions where intersection has been found between the surface \mathcal{S}_x and Ray R_p (i.e., $c_p + t_p \cdot v_p$ for $O_p = 1$):

$$L_{RGB} = \frac{1}{|P|} \sum_{O_p=1} |I_p - L_p(\Theta, \Phi, \tau, \theta_i, \phi_i, \psi_j)|, \quad (8)$$

where $|\cdot|$ is L_1 norm and L_p is defined in Eq. 4. The mask loss is

$$L_{mask} = \frac{1}{\alpha |P|} \sum_{O_p=0} CE(O_p, S_{p,\alpha}(\Theta, \tau, \theta_i, \psi_j)), \quad (9)$$

where CE is the cross-entropy loss and $S_{p,\alpha} = \text{sigmoid}(-\alpha \min_{t \leq 0} f(c + tv, \theta_i; \Theta))$ is an almost-everywhere-differentiable approximation to an indicator function for an object-occupied pixel p [72]. We enforce f to be approximately a signed distance function with the Eikonal regularization following Implicit Geometric Regularization (IGR) [15]:

$$L_E(\Theta) = \mathbb{E}_x(\|\nabla_x f(x, \theta_i, \psi_j; \Theta)\| - 1)^2, \quad (10)$$

where x is distributed uniformly in a bounding box of the scene. Lastly, following [43], we include a zero-mean multivariate Gaussian prior per latent code to facilitate learning a continuous shape manifold:

$$L_{code}(\theta_i, \phi_i, \psi_j) = \|\theta_i\|_2^2 + \|\phi\|_2^2 + \|\psi_j\|_2^2. \quad (11)$$

The goal during inference is to recover the representation $(\theta_i, \phi_i, \psi_j)$ given the RGB images of an unseen object. The three codes are randomly initialized, and then optimized through backpropagation with the following objective:

$$\hat{\theta}_i, \hat{\phi}_i, \hat{\psi}_j = \arg \min_{\theta_i, \phi_i, \psi_j} \mathcal{L}(\Theta, \Phi, \tau, \theta_i, \phi_i, \psi_j). \quad (12)$$

During optimization we can either fix the network weights (for in-distribution testing) or jointly optimize the network weights (for out-of-distribution testing).

4. Experiments

4.1. Experiment Setup

Dataset All experiments use SAPIEN [66], a large-scale, public domain data set containing 2346 articulated objects across 46 categories. We select six categories with representative articulation types and a sufficient number of instances: laptop, stapler, dishwasher, two-door fridge (LR for left and right, UD for up and down), eyeglasses, and storage furniture with drawer(s) (and door). We use the SAPIEN simulation environment [66] to render RGB images and corresponding masks. During training and testing, we sample every 10° for rotational joints and 10 states in total for sliding joints. For multiple joints, we take all combinations of each single joint sampling. For each articulation, 60 views are sampled for training and 6 views for inference. Please refer to supplementary materials for further details.

Evaluation Metrics We evaluate both the geometry and appearance of the predicted shape. For geometry, we sample 30,000 points per shape [37, 43] and evaluate the Chamfer-L1 distance, which is the mean of the accuracy and the completeness score [34]. For evaluating the rendered appearance, we report the Peak Signal-to-noise Ratio (PSNR). All visualizations in this paper are rendered from unseen views.

Training and Inference For training, latent codes are randomly initialized with $\mathcal{N}(0, \frac{1}{l})$, where l is the code length.

Table 1. **Methods used in experiments.** We specify whether they: handle static or articulated objects; share articulation code; use deformation field; output appearance or only geometry. We list the necessary input at train and test time. SDFs means SDF samples.

Method name	art./static	share art.	deform. field	train input	test input	appearance
A-SDF	art.	✓	×	SDFs	SDFs	×
IDR 6 views	static	×	×	6 RGBs	6 RGBs	✓
IDR 60 views	static	×	×	60 RGBs	6 RGBs	✓
Ours-base	art.	×	×	60 RGBs	6 RGBs	✓
Ours-Art	art.	✓	×	60 RGBs	6 RGBs	✓
Ours-Def	art.	×	✓	60 RGBs	6 RGBs	✓
Ours-ArtDef	art.	✓	✓	60 RGBs	6 RGBs	✓

We set $\rho = 100, \lambda = 0.1, \beta = 0.0001$ for the loss in Eq. 7. We start with $\alpha = 50$ and multiply it by a factor of 2 every 50,000 iterations (up to a total of 5 multiplications). During inference, articulation codes are initialized to the mean of all learned articulation codes, while other codes are initialized as in training. We run 600 iterations to recover the latent codes; if we do test-time adaptation [37], we fine-tune both, model weights and codes, for another 600 iterations.

Baseline Methods The method variants that we compare are listed in Tab. 1. For A-SDF [37], we sample SDF values from SAPIEN data as described in [37] and run the author-provided code. For IDR [72], we use our own implementation in PyTorch which follows the original work, but without the global lighting feature: the SAPIEN dataset does not provide such effects and we empirically found this does not influence the result. Each method may additionally use test-time adaptation (TTA), as described in [37], which in addition to optimizing the latent codes, optimizes the network weights during inference.

4.2. Reconstruction

To reconstruct unseen testing objects, we first optimize the codes (optionally with network weights) through backpropagation. Then we run another forward pass to extract the mesh and render images. In Tab. 2 and Tab. 3, we compare both geometry and appearance with previous methods for 9 object categories, respectively. We report the scores of our full model with articulation code sharing and using deformation field, with and without TTA. Please refer to the supplementary materials for a full list of results of each variant of our method. For reference, we also compare with two IDR models. Note that IDR does not have an embedding space to encode categorical priors and can only reconstruct a single static object or scene that it has been trained on, *i.e.* for IDR we train and test on the same object instances, one model per articulation state. Since this is too computationally expensive, we randomly choose two objects with two articulations in each category and report the average PSNR as the PSNR on each category for IDR. Our model is trained on 60 views per articulation and tested on 6 views per articulation.

Table 2. **Comparison for reconstructing unseen synthetic shapes (Chamfer-L1).** We compare our method with A-SDF [37] and IDR [72]. As training IDR for each object is too computationally expensive, we present the average across 2 articulation states for 2 objects from each category. IDR is tested on the same objects it is trained on. Please note that our method uses 60 views for training and 6 views for inference. A-SDF* indicates A-SDF results compared to the geometry it trains on, while all other results use a sampling of the original geometry as ground truth. Lower score is better.

Method	Laptop	Stapler	Dishwasher	Eyeglasses	FridgeLR	FridgeUD	Drawer	DrawerUD	Drawer+Door
A-SDF*	0.126	1.510	0.543	15.972	0.599	0.837	1.362	4.791	2.282
A-SDF TTA*	0.103	0.978	0.209	7.792	1.682	4.623	1.142	2.832	0.476
A-SDF	0.580	6.058	4.180	17.298	1.527	1.427	1.971	6.048	2.945
A-SDF TTA	0.542	5.358	3.756	9.052	1.351	0.842	1.689	4.139	1.082
IDR 6 views	1.656	1.113	4.139	1.386	7.915	1.826	4.202	fails	12.672
IDR 60 views	0.259	0.994	3.106	1.171	12.368	2.119	3.047	7.871	13.491
Ours-ArtDef	0.382	1.125	3.945	9.790	2.738	3.648	2.627	5.979	3.264
Ours-ArtDef TTA	0.355	0.936	3.936	7.894	2.063	3.649	2.745	5.912	3.243

Table 3. **Comparison with IDR [72] for reconstructing unseen synthetic shapes (PSNR).** The training and inference procedures of IDR and our methods are the same as in Tab. 2. Note that IDR trains a separate model for each articulation state of each object instance, it is trained on 60 views and then tested on 6 novel views of the same object, *i.e.* it trains and tests on the same instances, while our method is trained per category and is tested on unseen objects. IDR 60 views is trained on 60 views and tested on 6 novel views, offering an upper bound for the quality we can expect from our method. Higher score is better.

Method	Laptop	Stapler	Dishwasher	Eyeglasses	FridgeLR	FridgeUD	Drawer	DrawerUD	Drawer+Door
IDR 6 views	13.32	9.75	17.64	11.49	10.81	13.47	15.26	fails	15.80
IDR 60 views	20.70	22.40	24.10	26.59	20.01	23.49	24.32	21.19	22.71
Ours-ArtDef	18.33	17.18	20.79	20.69	18.79	23.72	24.65	22.25	23.34
Ours-ArtDef TTA	17.84	18.15	20.87	20.89	18.94	23.52	24.27	22.20	23.96

ulation for unseen objects, while IDR is trained on 60 or 6 views for one static object and tested on novel views of the same object with the same articulation. Therefore, IDR trained on 60 views gives us a sense of the upper bound performance of our differentiable rendering system.

Note that meshes in the dataset are not watertight, so A-SDF processes them with the Manifold [18] software in order to be able to sample SDF values, which does a re-sampling that leads to a thickening of the original meshes. In A-SDF [37] output geometry is evaluated against those thickened meshes that were used for training, rather than the original geometry, so we follow the same evaluation protocol and report it as A-SDF* in Tab. 2, achieving comparable numbers to the original paper. All other method variants are evaluated with respect to a sampling of the original geometry, which may sometimes contain points sampled from hidden, internal structures, such as the bottom of a laptop’s trackpad, but the fraction of these points is small.

We can see that despite being used on a harder task, our model performs on par with 60-view IDR on many categories such as furniture. On other categories, our PSNR is significantly lower. This is, because these categories tend to have higher frequency texture. To show the benefit of the learned categorical prior introduced in our model, we further compare with an IDR model trained on 6 views. Eventually, we would like to have a model that works nicely on unseen objects with only very limited observations. This brings up the question: should we model the object by over-

fitting an IDR model with very few views or leveraging categorical priors—which is better? The comparison between our method and 6-view IDR in Tab. 3 and Tab. 2 clearly shows that by introducing a shape, articulation, and appearance prior, our model during inference significantly improves over an overfitted IDR with same number of views. We observe through visualization that our variants with deformation field performs better than without it when articulations involve topology change (*e.g.*, a closed laptop being opened). And sometimes TTA may result in overfitting to appearance and makes the geometry prediction worse.

We visualize the testing results from unseen views in Fig. 3 and compare with A-SDF (with TTA). All models are trained with shared articulations and after testing time optimization. The first two rows are single rotational joints followed by three rows of multi-joint categories with various joint type combinations. We can see that for classes with small intra-class geometry variation such as laptop and dishwasher, A-SDF works well, which is also consistent with observations from Tab. 2. However, for classes with larger intra-class shape variations such as staplers and eyeglasses, A-SDF fails to capture the geometry details. For example, for A-SDF, the geometry of the bottom part of both staplers are not correct, despite being directly optimized with 3D geometry during inference. While A-SDF originally only shows results on rotational joints, we further test both methods on other joint types. We can see that the performance of A-SDF on sliding joints is much worse

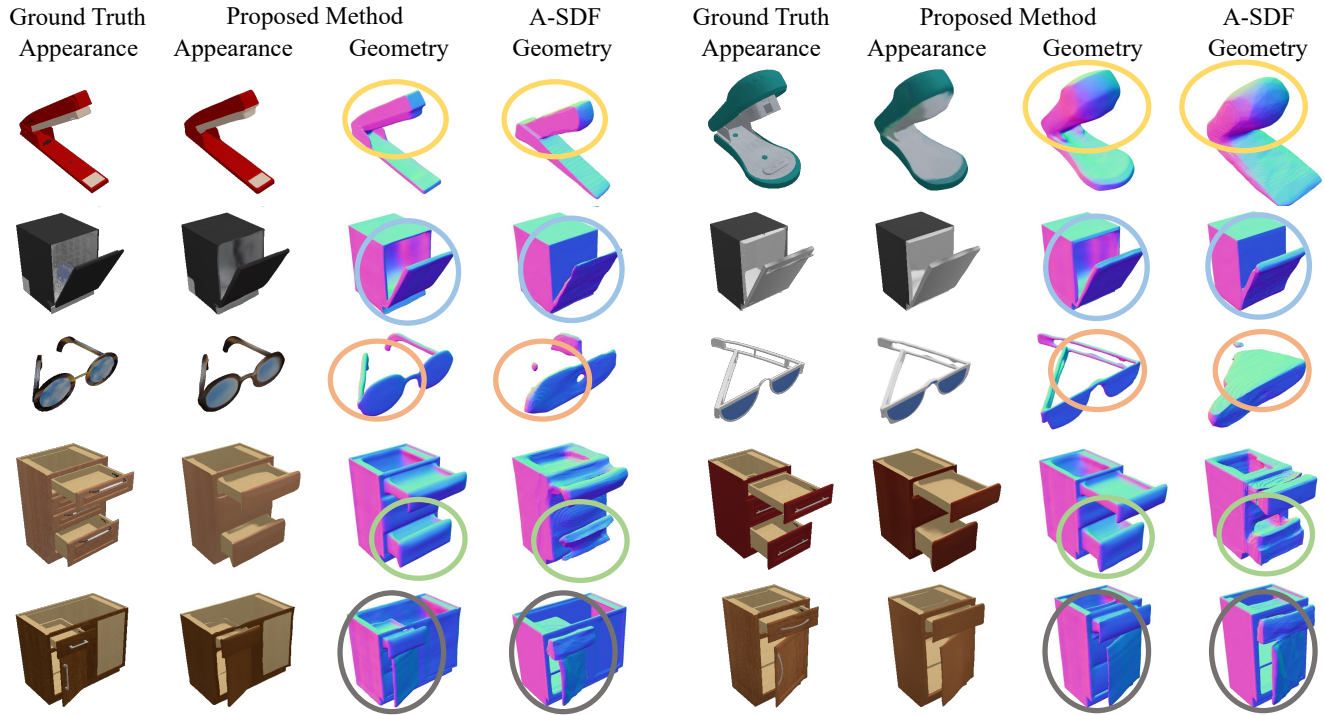


Figure 3. **Reconstruction results from unseen data.** We compare to A-SDF on various joint types and combinations, including revolute, prismatic and multiple ones. While A-SDF trains using both geometry and articulation ground truth and only predicts geometry, our method also faithfully recovers appearance in addition to producing better geometry and more accurate articulations from only RGB supervision.

than on rotational joints. We reckon one reason is that A-SDF requires articulation ground truth input during training. However, for drawers with different lengths, it is hard to define a single value shared across all objects. In contrast, our method does not require articulation annotation and learns the articulation code through training. Please refer to the supplementary material for more results.

4.3. Analysis

Interpolation and Extrapolation One application of the proposed method is to generate new articulations of an object through interpolation and extrapolation, given only a few training articulations. To do so, we render a few images for two articulations, and optimize their shape, articulation, and appearance codes jointly with the images. After the two sets of codes are estimated, we interpolate/extrapolate a set of shape, articulation and appearance codes. This procedure follows what is done in A-SDF [37], to which we compare in Fig. 4 for different joint types and combinations. We run the two fastest variants of our method for this, *i.e.* those without deformation field. Note that there is no TTA for either method as the two estimated code sets which are being interpolated need to share the same network weights. We see that A-SDF may fail on extrapolations more than 6° out of the training range for laptop and extrapolations for other classes, while our method is able to recover plau-

sible shapes. The quantitative results in Tab. 4 demonstrate that our interpolated models are geometrically more accurate than those of A-SDF for the majority of object categories.

Disentangling Geometry and Appearance. The separate geometry and appearance networks strongly enforce disentanglement, as inherited from IDR [72]. One application of our method is to switch the geometry and appearance codes between different objects. As shown in Fig. 5, by replacing the appearance code with one from another pair of eyeglasses, we can faithfully create a new pair of eyeglasses that has a new appearance but the same geometry. Note how the colors from the thin frame correctly map onto the frame of the new eyeglasses, despite the large geometry difference between the shapes. Our learned embedding space that encodes categorical prior also helps with the correct mapping.

4.4. Testing on Real-world RGB Images

Since our method only uses images for supervision and does not require any 3D annotations that may be expensive, it can be easily set up to test on real data. In this section, we directly test our proposed method (trained from synthetic data) on images captured in the real world. We use a personal cell phone to record a static opened laptop or drawer with fixed focal length and exposure. We then run Structure-from-Motion (SfM) algorithm [52] on

Table 4. **Interpolation Chamfer-L1 error.** Evaluation details are the same as in Tab. 2. Lower score is better.

Method	Laptop	Stapler	Dishwasher	Eyeglasses	FridgeLR	FridgeUD	Drawer	DrawerUD	Drawer+Door
A-SDF*	0.359	4.989	2.101	45.326	1.445	1.677	1.634	6.067	4.454
A-SDF	0.610	5.918	4.744	43.708	1.708	1.881	1.958	7.105	5.032
Ours-base	0.347	1.486	3.029	2.632	3.068	4.723	2.952	5.195	3.226
Ours-Art	0.309	1.716	2.807	2.588	3.860	3.357	3.086	4.151	3.845

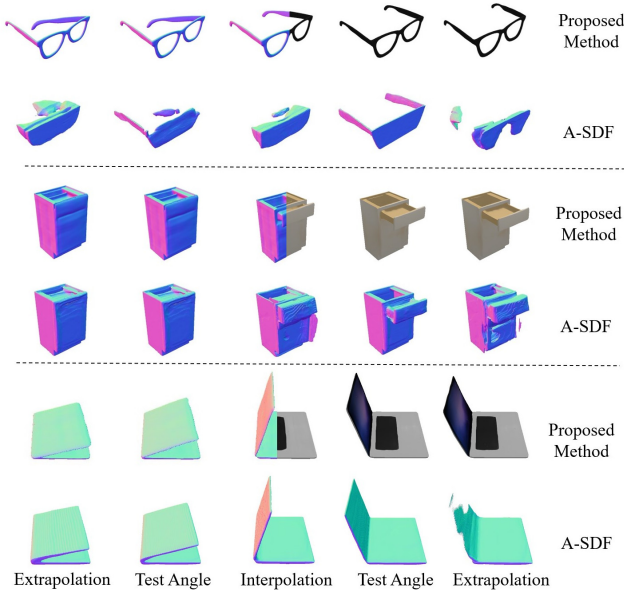


Figure 4. **Interpolation and extrapolation comparison.** For these unseen objects, we first infer the geometry, appearance and articulation codes for multiple testing states, then we interpolate / extrapolate the inferred codes to generate new articulations. The proposed method successfully generates shapes with articulations beyond the range seen during training for various joint types.

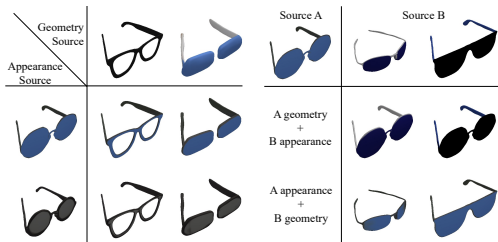


Figure 5. Disentangling shape and appearance for shape synthesis.

the captured frames to estimate the camera calibration parameters and their poses. For each view, we then run <https://remove.bg> to estimate a segmentation mask for the foreground object. The views shown in Fig. 1 are the input images to our model. We test our model trained on synthetic data from SAPIEN with deformation field and shared articulation code on these real-world images. The shape, articulation, and appearance codes are initialized as described earlier, we then jointly fine-tune both network weights and

codes on these images for 2000 iterations. At this point, we are able to reconstruct the static real objects. Then by replacing the inferred articulation code with the articulation codes learned during training, we are able to articulate the static reconstruction realistically.

4.5. Limitations

While we push the boundaries for articulated shape reconstruction by removing limitations on required data and supervision compared to previous methods, remaining limitations exist. Even though we are able to fine-tune our object models on real data, the domain gap from synthetic to real remains large, and the appearance prior we learn from the limited synthetic data is not powerful enough to explain general object appearance. As a consequence, we make use of foreground masks to alleviate this problem. A more elegant direction for future work could be to follow the example of VoISDF [71] and extend the method towards unsupervised disentanglement of shape and appearance. Another limitation is the current scaling behavior w.r.t. the number of joints: for modeling objects with n joints with m states, we need m^n combinations. This is only feasible for objects with a low number of joints. Better joint priors and decoupling are interesting directions for future research.

5. Conclusion

In this paper, we set out to answer the research question whether it is possible to jointly learn a prior over the 3D geometry, articulation, and appearance of an entire class of objects, solely from photometric 2D observations, with no articulation annotations. Our results show that this is not only possible but, given enough computational power, can be achieved with high fidelity and with remarkably little data. In our experiments, we successfully fine-tune our model to real-world data using only 6 views and create animatable objects that closely resemble the real-world objects under articulations, interpolated, and extrapolated. At the same time, our method is the first that not only handles revolute, but also prismatic joints and combinations thereof. We hope that this encouraging result inspires further research into general-purpose object reconstruction.

Acknowledgement We would like to thank Michael Goesele, Eddy Ilg, Zhaoyang Lv, Jisan Mahmud, Tanner Schmidt, and Anh Thai for helpful discussions.

References

- [1] Ben Abbatematteo, Stefanie Tellex, and George Konidaris. Learning to generalize kinematic models to novel objects. *Conference on Robot Learning*, 2019. 2
- [2] Kara Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural Point-Based Graphs. In *Lecture Notes in Computer Science (including sub-series Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 12367 LNCS, pages 696–712. Springer Science and Business Media Deutschland GmbH, jun 2020. 3
- [3] Benjamin Attal, Eliot Laidlaw, Aaron Gokaslan, Changil Kim, Christian Richardt, James Tompkin, and Matthew O’Toole. Törf: Time-of-flight radiance fields for dynamic scene view synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 3
- [4] Aljaz Bozic, Pablo Palafox, Michael Zollhofer, Justus Thies, Angela Dai, and Matthias Nießner. Neural deformation graphs for globally-consistent non-rigid reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1450–1459, 2021. 2
- [5] Aljaz Bozic, Michael Zollhofer, Christian Theobalt, and Matthias Nießner. Deepdeform: Learning non-rigid rgb-d reconstruction with semi-supervised data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7002–7012, 2020. 2
- [6] Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *European Conference on Computer Vision*, pages 608–625. Springer, 2020. 3
- [7] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 5932–5941, 2019. 2
- [8] Chiyu Max Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas Funkhouser. Local implicit grid representations for 3D scenes. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 6000–6009, 2020. 3
- [9] Boyang Deng, John P Lewis, Timothy Jeruzalski, Gerard Pons-Moll, Geoffrey Hinton, Mohammad Norouzi, and Andrea Tagliasacchi. NASA neural articulated shape approximation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 612–628. Springer, 2020. 2
- [10] Yu Deng, Jiaolong Yang, and Xin Tong. Deformed implicit field: Modeling 3d shapes with learned dense correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10286–10296, 2021. 2
- [11] Yilun Du, Yanan Zhang, Hong-Xing Yu, Joshua B. Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 3
- [12] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8649–8658, June 2021. 3
- [13] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 3
- [14] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local Deep Implicit Functions for 3D Shape. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 4857–4866, 2020. 2
- [15] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *International Conference on Machine Learning*, pages 3789–3799. PMLR, 2020. 5
- [16] Kaiwen Guo, Feng Xu, Tao Yu, Xiaoyang Liu, Qionghai Dai, and Yebin Liu. Real-time geometry, albedo, and motion reconstruction using a single rgb-d camera. *ACM Transactions on Graphics (ToG)*, 36(4):1, 2017. 2
- [17] Yudong Guo, Keyu Chen, Sen Liang, Yong-Jin Liu, Hujun Bao, and Juyong Zhang. AD-NeRF: Audio Driven Neural Radiance Fields for Talking Head Synthesis. *IEEE/CVF International Conference on Computer Vision (ICCV)*, mar 2021. 3
- [18] Jingwei Huang, Hao Su, and Leonidas Guibas. Robust Watertight Manifold Surface Generation Method for ShapeNet Models. 2018. 6
- [19] Zeng Huang, Yuanlu Xu, Christoph Lassner, Hao Li, and Tony Tung. ARCH: Animatable Reconstruction of Clothed Humans. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3090–3099, 2020. 3
- [20] Matthias Inmann, Michael Zollhöfer, Matthias Nießner, Christian Theobalt, and Marc Stamminger. Volumedeform: Real-time volumetric non-rigid reconstruction. In *European Conference on Computer Vision*, pages 362–379. Springer, 2016. 2
- [21] Ajinkya Jain, Rudolf Lioutikov, and Scott Niekum. Screwnet: Category-independent articulation model estimation from depth images using screw theory. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13670–13677, 2021. 2
- [22] Hanbyul Joo, Tomas Simon, and Yaser Sheikh. Total capture: A 3d deformation model for tracking faces, hands, and bodies. 2018. 2
- [23] Christoph Lassner and Michael Zollhöfer. Pulsar: Efficient Sphere-based Neural Rendering. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 3
- [24] Tianye Li, Mira Slavcheva, Michael Zollhofer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, and Zhaoyang Lv. Neural 3D Video Synthesis. 2021. 2, 3
- [25] Xiaolong Li, He Wang, Li Yi, Leonidas Guibas, A Lynn Abbott, and Shuran Song. Category-level articulated object

- pose estimation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. [2](#)
- [26] Yang Li, Aljaz Bozic, Tianwei Zhang, Yanli Ji, Tatsuya Harada, and Matthias Nießner. Learning to optimize non-rigid tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4910–4918, 2020. [2](#)
- [27] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. [3](#)
- [28] 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. *ACM Trans. Graph.(ACM SIGGRAPH Asia)*, 2021. [3](#)
- [29] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, July 2019. [3](#)
- [30] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Trans. Graph.*, 40(4), July 2021. [3](#)
- [31] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. SMPL: A skinned multi-person linear model. *ACM Transactions on Graphics (TOG)*, 34(6):1–16, 2015. [2](#)
- [32] Roberto Martín-Martín, Clemens Eppner, and Oliver Brock. The rbo dataset of articulated objects and interactions. *The International Journal of Robotics Research*, 38, 2018. [15](#)
- [33] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 4460–4470, 2019. [1, 2](#)
- [34] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. [5](#)
- [35] Marko Mihajlovic, Yan Zhang, Michael J Black, and Siyu Tang. LEAP: Learning articulated occupancy of people. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10461–10471, 2021. [2](#)
- [36] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 12346 LNCS, pages 405–421. Springer Science and Business Media Deutschland GmbH, mar 2020. [3](#)
- [37] Jiteng Mu, Weichao Qiu, Adam Kortylewski, Alan Yuille, Nuno Vasconcelos, and Xiaolong Wang. A-SDF: Learning disentangled signed distance functions for articulated shape representation. In *ICCV*, 2021. [2, 5, 6, 7, 13, 14, 15](#)
- [38] Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 343–352, 2015. [2](#)
- [39] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable Volumetric Rendering: Learning Implicit 3D Representations without 3D Supervision. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3501–3512, 2020. [3](#)
- [40] Atsuhiko Noguchi, Xiao Sun, Stephen Lin, and Tatsuya Harada. Neural Articulated Radiance Field. *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. [2, 3](#)
- [41] Pablo Palafox, Aljaž Božič, Justus Thies, Matthias Nießner, and Angela Dai. Npms: Neural parametric models for 3d deformable shapes. In *ICCV*, 2021. [2](#)
- [42] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 165–174, 2019. [1, 2, 3](#)
- [43] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. [2, 5](#)
- [44] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable Neural Radiance Fields. *ICCV*, 2021. [3, 4](#)
- [45] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021. [3, 4](#)
- [46] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Animatable Neural Radiance Fields for Human Body Modeling. *IEEE/CVF International Conference on Computer Vision (ICCV)*, may 2021. [3](#)
- [47] 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. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. [3](#)
- [48] Sudeep Pillai, Matthew Walter, and Seth Teller. Learning articulated motions from visual demonstration. *page Robotics: Science and Systems*, 2014. [2](#)
- [49] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *CVPR*, 2021. [3](#)

- [50] Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics (TOG)*, 36(6):1–17, 2017. [2](#)
- [51] Darius Rückert, Linus Franke, and Marc Stamminger. ADOP: Approximate Differentiable One-Pixel Point Rendering. 2021. [3](#)
- [52] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [7](#), [14](#)
- [53] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, volume 32, 2019. [3](#)
- [54] Miroslava Slavcheva, Maximilian Baust, Daniel Cremers, and Slobodan Ilic. Killingfusion: Non-rigid 3d reconstruction without correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1395, 2017. [2](#)
- [55] Miroslava Slavcheva, Maximilian Baust, and Slobodan Ilic. Sobolevfusion: 3d reconstruction of scenes undergoing free non-rigid motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2646–2655, 2018. [2](#)
- [56] Shih-Yang Su, Frank Yu, Michael Zollhoefer, and Helge Rhodin. A-NeRF: Surface-free Human 3D Pose Refinement via Neural Rendering. *Advances in Neural Information Processing Systems (NeurIPS)*, feb 2021. [3](#)
- [57] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, Tomas Simon, Christian Theobalt, Matthias Niessner, Jonathan T. Barron, Gordon Wetzstein, Michael Zollhoefer, and Vladislav Golyanik. Advances in neural rendering, 2021. [3](#)
- [58] 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. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, dec 2021. [3](#), [4](#)
- [59] 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 *ICCV*, 2021. [14](#), [15](#)
- [60] Alex Trevithick and Bo Yang. GRF: Learning a General Radiance Field for 3D Representation and Rendering. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. [3](#)
- [61] Weiyue Wang, Qiangeng Xu, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. DISN: Deep implicit surface network for high-quality single-view 3D reconstruction. In *Advances in Neural Information Processing Systems*, volume 32, 2019. [1](#), [3](#)
- [62] Ziyang Wang, Timur Bagautdinov, Stephen Lombardi, Tomas Simon, Jason Saragih, Jessica Hodgins, and Michael Zollhöfer. Learning Compositional Radiance Fields of Dynamic Human Heads. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. [3](#)
- [63] Yijia Weng, He Wang, Qiang Zhou, Yuzhe Qin, Yueqi Duan, Qingnan Fan, Baoquan Chen, Hao Su, and Leonidas J. Guibas. Captra: Category-level pose tracking for rigid and articulated objects from point clouds. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 13209–13218, October 2021. [2](#)
- [64] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 7465–7475. IEEE Computer Society, dec 2020. [3](#)
- [65] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9421–9431, 2021. [3](#)
- [66] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107, 2020. [5](#), [13](#)
- [67] Christopher Xie, Keunhong Park, Ricardo Martin-Brualla, and Matthew Brown. Fig-nerf: Figure-ground neural radiance fields for 3d object category modelling. *arXiv preprint arXiv:2104.08418*, 2021. [2](#)
- [68] Hongyi Xu, Thiemo Alldieck, and Cristian Sminchisescu. H-nerf: Neural radiance fields for rendering and temporal reconstruction of humans in motion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. [3](#)
- [69] Hongyi Xu, Eduard Gabriel Bazavan, Andrei Zanfir, William Freeman, Rahul Sukthankar, and Cristian Sminchisescu. GHUM & GHUML: Generative 3d human shape and articulated pose models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. [2](#)
- [70] 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. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. [2](#), [3](#)
- [71] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *CoRR*, abs/2106.12052, 2021. [8](#)
- [72] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020. [1](#), [3](#), [4](#), [5](#), [6](#), [7](#), [14](#)
- [73] Ge Zhang, Or Litany, Srinath Sridhar, and Leonidas J. Guibas. Strobenet: Category-level multiview reconstruction of articulated objects. *CoRR*, abs/2105.08016, 2021. [2](#)

- [74] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5524–5532, Los Alamitos, CA, USA, Jul 2017. [2](#)

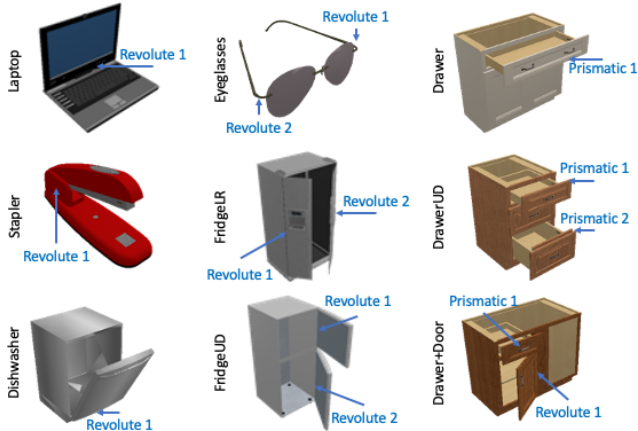


Figure 6. **Objects.** We show one sample object from each category in one articulation state. The joints and their types are annotated.

A. Supplementary Material

In this supplementary material, we describe details of dataset preparation in Sec. A.1 and implementation details for training, inference, and experiments on real data in Sec. A.2. In Sec. A.3, we provide more quantitative and qualitative results.

A.1. Dataset

All experiments use SAPIEN [66], a large-scale, public domain dataset containing 2346 articulated objects across 46 categories. We select six categories with representative articulation types and a sufficient number of instances: laptop, stapler, dishwasher, two-door fridge (LR for left and right, UD for up and down), eyeglasses, and storage furniture with drawer(s) (and door) (Drawer is for single-drawer furniture, DrawerUD for two-drawer furniture, and Drawer+Door for furniture that has one drawer and one door). Note that we follow the same classification practice of A-SDF. For example, FridgeLR and FridgeUD both belong to the category of two-door fridges, but we still trained two separate models because A-SDF treated these two as two categories. We didn't try training on a combined category, but we expect it to work. The different combinations of joint types and numbers in total make nine different categories. We display one example of each category in Fig. 6.

To render the shapes, we normalize them to fit in a unit sphere and make sure that the same object with different articulations are normalized in the same way (their non-motion parts are aligned). We use the SAPIEN simulation environment [66] to render RGB images and corresponding masks. During training and testing, we sample every 10° for rotational joints and 10 states in total for sliding joints. For multiple joints, we take all combinations of every single joint sampling. For each articulation, 60 views are sampled for training and 6 views for inference. Cameras

Table 5. **Dataset details.** We list the details of the SAPIEN data set for synthetic experiments. It covers a wide range of object classes and joint types. For each category, we show the number of joints of each type (revolute or prismatic), the number of object instances in the training and testing splits, the number of articulations sampled for training, and the number of views used for training and testing.

Category	#joint	train / test split	#art.	train/test #view
Laptop	1 revolute	35/11	10	60/6
Stapler	1 revolute	15/5	10	60/6
Dishwasher	1 revolute	18/6	10	60/6
Eyeglasses	2 revolute	48/14	36	60/6
FridgeLR	2 revolute	8/3	36	60/6
FridgeUD	2 revolute	12/4	36	60/6
Drawer	1 prismatic	21/7	10	60/6
DrawerUD	2 prismatic	27/9	36	60/6
Drawer+Door	1 revolute, 1 prismatic	9/4	100	60/6



Figure 7. Articulations used in the experiments are not aligned.

are placed on vertices of a randomly rotated rhombicosidodecahedron for 60 views (octahedron for 6 views) with the object in its center. The RGB images and masks are of resolution 640×480 . These details are summarized in Tab. 5.

We set the angle range to train and test on revolute joints following A-SDF [37]. To evaluate interpolation, for every two neighboring testing articulations, we use the codes for these two articulations to interpolate the middle point. Concretely, for the stapler and dishwasher with training and testing angles $\{0, 10, 20, 30, 40, 50, 60, 70, 80, 90\}$, the angles used for evaluating interpolation are $\{5, 15, 25, 35, 45, 55, 65, 75, 85\}$. For laptop, angles used for training and testing are $\{-72, -62, -52, -42, -32, -22, -12, -2, 8, 18\}$ and used for interpolation are $\{-67, -57, -47, -37, -27, -17, -7, 3, 13\}$. For the eyeglasses and fridge (fridgeLR and fridgeUD) with training and testing angles $\{0, 10, 20, 30, 40, 50\}$ for each joint, the angles used for evaluating interpolation are $\{5, 15, 25, 35, 45\}$. For the drawer in storage furniture (Drawer, DrawerUD, Drawer+Door), we sample 10 articulations with equal distance for training and testing and use the 9 midpoints of the 10 articulations for interpolation. For the door in storage furniture (Drawer+Door), we use $\{0, 10, 20, 30, 40, 50, 60, 70, 80, 90\}$ for training and testing, and $\{5, 15, 25, 35, 45, 55, 65, 75, 85\}$ to evaluate interpolation.

To clarify, aligned articulation is *not* required in training. In fact, SAPIEN objects are not aligned and we do not align them in our experiments. In Fig. 7, for example, eyeglasses at the articulation states 0 and 9 differ in the lens-leg angles. However, if articulations are roughly aligned, we can also leverage it by sharing articulations (main paper Sec. 3.2) in variants Ours-Art/ArtDef.

A.2. Implementation Details

A.2.1 Network Architecture

The architecture of the geometry and appearance networks in our method follows exactly the description in IDR [72]. Concretely, the geometry network takes a 256-dimensional geometry feature and 3-dimensional 3D query location (and optionally a 8-dimensional articulation feature if running without deformation field) as input and predicts a single SDF value. When included, the deformation module takes in a 256-dimensional geometry feature, 3-dimensional 3D query location and a 8-dimensional articulation feature as input and predicts a 3-dimensional displacement for the query point, which is added to the original query point and then passed to the geometry network. Both the geometry and the deformation network have eight fully connected hidden layers with a width 512 and a last fully connected layer with output dimension 1 or 3 for their corresponding predictions. There is a single skip connection from the input to the middle layer. The fully connected layers are interlaced with softplus activation in both networks. We follow the non-linear maps [72] on the input query points. We initialize the weights of the geometry network so that it produces an approximate SDF of a unit sphere.

In the appearance network, there are four fully connected layers with output dimension 512 and a last fully connected layer with output dimension 3 for color prediction. The input is a concatenation of the following: a 256-dimensional appearance feature for each object, a 3D surface point and its normal, and the viewing direction. We use the ReLU activation between hidden layers of the appearance network and tanh for the output to get valid color values.

A.2.2 Training and Inference

For training, latent codes are randomly initialized with $\mathcal{N}(0, \frac{1}{l})$, where l is the code length. We set $\rho = 100$, $\lambda = 0.1$, $\beta = 0.0001$ for the loss in Eq. 7 of the main paper. We start with $\alpha = 50$ and multiply it by a factor of 2 every 50,000 iterations (up to a total of 5 multiplications). The networks are trained using ADAM optimizer with a learning rate starting from 0.0001 and decreasing by a factor of 2 at the 50% and 75% point of the total number of iterations.

During inference, articulation codes are initialized to the mean of all learned articulation codes, while other codes are initialized as in training. To reconstruct unseen testing objects, we first optimize the geometry, articulation, and appearance codes through backpropagation for 600 iterations with learning rate starting from 0.009 and decreasing by a factor of 2 at 300 and 450 iterations. If we do test-time adaptation [37], we further optimize both the codes and the network weights for another 600 iterations with learning rate starting from 0.00005 and decreasing by a factor of

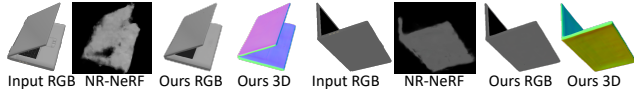


Figure 8. Comparison with NR-NeRF [59] on articulating a laptop.

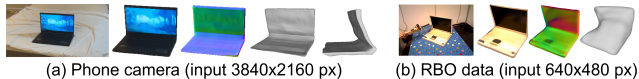


Figure 9. **Reconstruction from a single RGB image.** We show input RGB, output appearance and normals, other untextured views.

2 at 300 and 450 iterations. For both optimization stages, we start with $\alpha = 50$ and multiply it by a factor of 2 every 100 iterations (up to a total of 5 multiplications). Then we run another forward pass to predict SDF values and render images. The Marching Cubes algorithm is used to extract an approximate iso-surface given the predicted SDF values.

A.2.3 Real Experiment Setup

We test the model trained on synthetic laptops and drawers and directly apply the trained models to real-world phone-captured static objects. We use a personal cell phone to record a static opened laptop or drawer with fixed focal length and exposure. We then run Structure-from-Motion (SfM) algorithm [52] on the captured frames to estimate the camera calibration parameters and their poses. For each view, we then run <https://remove.bg> to estimate a segmentation mask for the foreground object. We use seven input images to reconstruct the laptop and 24 images to reconstruct the drawer in Fig. 1 of the main paper. We test our model trained on synthetic data from SAPIEN with deformation field and shared articulation code on these real-world images. The shape, articulation, and appearance codes are initialized as described earlier, we then jointly fine-tune both the network weights and the codes on these images for 2000 iterations. At this point, we are able to reconstruct the static real objects. Then by replacing the inferred articulation code with the articulation codes learned during training, we are able to articulate the static reconstruction realistically.

A.3. Results

Full quantitative results. In Tab. 6, we show the full list of results for each variant of our method. We observe that with deformation field it manages better with topology changes, but it takes longer to train. This is why the results of Ours-Def and Ours-ArtDef might be numerically worse as those models did not get to the same number of iterations in the same training time as without deformation field. We also observe that sometimes TTA may cause the model to optimize towards a local minimum, *e.g.* overfitting to ap-

Table 6. **Reconstruction results on unseen synthetic shapes (Chamfer-L1)**. We compare all variants of our proposed method. This table corresponds to Table 2 from the main paper.

Method	Laptop	Stapler	Dishwasher	Eyeglasses	FridgeLR	FridgeUD	Drawer	DrawerUD	Drawer+Door
Ours-base	0.383	1.453	3.269	1.771	2.969	4.683	2.924	5.326	2.786
Ours-Art	0.328	1.560	2.962	1.735	3.955	3.332	3.114	4.185	3.416
Ours-Def	0.363	1.026	4.046	2.558	1.976	5.007	3.005	5.726	3.394
Ours-ArtDef	0.382	1.125	3.945	9.790	2.738	3.648	2.627	5.979	3.264
Ours-base TTA	0.345	1.336	3.187	1.606	1.637	4.614	2.940	5.100	2.899
Ours-Art TTA	0.475	1.400	2.881	1.659	2.635	3.238	3.135	4.166	3.897
Ours-Def TTA	0.333	0.815	4.046	2.026	2.244	4.669	3.042	5.335	3.652
Ours-ArtDef TTA	0.355	0.936	3.936	7.894	2.063	3.649	2.745	5.912	3.243

pearance while making the geometry worse. The errors are larger on bulky objects like fridges, drawers, dishwashers, where the concave geometry is visible from very few views, so a method that only uses RGB information may not have enough coverage to carve the space out. While the numbers may not reflect all variants’ strengths, combined with visualizations, we observe variants with deformation handle large topology changes better. This is confirmed in Tab. 6 where ours-Def TTA performs the best on the stapler.

Comparison with NeRF-extension. In Fig. 8, we show a comparison with NR-NeRF [59], a representative NeRF extension to multi-view dynamic scenes. We ran its official code and our method on a SAPIEN laptop with the same $60 \text{ views} \times 10 \text{ angles}$ setting. We observe that despite only recovering a single scene, NR-NeRF performs poorly due to large inter-frame movements.

Results on RBO dataset [32]. RBO dataset [32] only has monocular videos of articulated objects with fixed camera-object pose, so it is improper to evaluate our multi-view method. We still tested our single-view reconstruction on our real phone camera data and RBO in Fig. 9. It succeeds on high-res phone images, but on noisy, low-res RBO data it recovers plausible appearance but poor geometry that looks correct only from the input view. This strongly indicates that a few more views will be sufficient to disambiguate even noisy input. The Chamfer-L1 distance of the RBO example is 5.75 after scale determination, which is close to the DeepSDF error reported in A-SDF [37], even though we do not use 3D input.

A.4. Video

Please refer to the video for more results on reconstruction, interpolation and extrapolation on testing synthetic data, as well as reconstruction and animation on real data.