Supplementary Material: Self-supervised Neural Articulated Shape and Appearance Models

Fangyin Wei[1*] Rohan Chabra[2] Lingni Ma[2] Christoph Lassner[2] Michael Zollhoefer[2]
Szymon Rusinkiewicz[1] Chris Sweeney[2] Richard Newcombe[2] Mira Slavcheva[2]

[1]Princeton University [2]Reality Labs Research

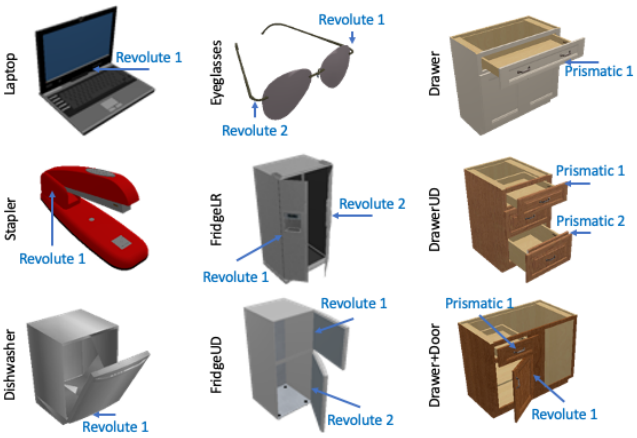

Figure 1. **Objects.** We show one sample object from each category in one articulation state. The joints and their types are annotated.

Table 1. **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

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

1. Dataset

All experiments use SAPIEN [5], 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. 1.

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 [5] 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 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. 1.

We set the angle range to train and test on revolute joints following A-SDF [2]. To evaluate interpolation, for every two neighboring testing articulations, we use the codes for

*Work done during internship at Reality Labs Research.

Figure 2. Articulations used in the experiments are not aligned.

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. 2, 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.

2. Implementation Details

2.1. Network Architecture

The architecture of the geometry and appearance networks in our method follows exactly the description in IDR [6]. Concretely, the geometry network takes a 256-dimensional geometry feature and 3-dimensional 3D query location (and optionally an 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 an 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 [6] 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.

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 [2], 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 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.

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 [3] 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

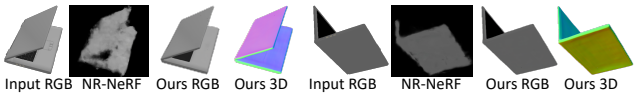

Figure 3. Comparison with NR-NeRF [4] on articulating a laptop.

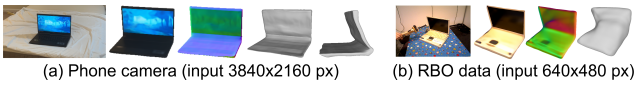

Figure 4. **Reconstruction from a single RGB image.** We show input RGB, output appearance and normals, other untextured views.

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.

3. Results

Full quantitative results. In Tab. 2, 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 appearance 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. 2 where ours-Def TTA performs the best on the stapler.

Comparison with NeRF-extension. In Fig. 3, we show a comparison with NR-NeRF [4], 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 views $\times$ 10 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 [1]. RBO dataset [1] 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. 4. 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 [2], even though we do not use 3D input.

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.

References

[1] Roberto Martín-Martín, Clemens Eppner, and Oliver Brock. The rbo dataset of articulated objects and interactions. *The International Journal of Robotics Research*, 38, 2018. 3

[2] 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. 1, 2, 3

[3] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[4] 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. 3

[5] 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. 1

[6] 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. 2

Table 2. **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