Neural Implicit Representation for Building Digital Twins of Unknown Articulated Objects

Supplementary Material

A. Method Details

In this section, we describe the details of our method and adapted PARIS*/PARIS*-m. We will also release our code and data to facilitate future research.

A.1. Neural Object Field

We use Neural Object Field [35] as the object representation for our stage one reconstruction. We follow the practice of [35] and we describe the specifics below.

Given multi-view posed RGB-D images of the object $\mathcal{O}^t$ at state $t, t \in \{0, 1\}$, we reconstruct the object in the form of a Neural Object Field [35] (Ω^t, Φ^t) (we omit t for simplicity in the following). The geometry network $\Omega : \mathbf{x} \mapsto s$ maps spatial point $\mathbf{x} \in \mathbb{R}^3$ to its truncated signed distance $s = \text{clip}(d/\tau_d, -1, 1)$, where $\tau_d = 0.03$ is the truncation distance, d is the signed distance to the object surface. The appearance network $\Phi : (\mathbf{x}, \mathbf{d}) \mapsto \mathbf{c}$ maps point $\mathbf{x} \in \mathbb{R}^3$ and view direction $\mathbf{d} \in \mathbb{S}^2$ to RGB color $C \in \mathbb{R}^3$.

We supervise the Neural Object Field at 3D points $\{\mathbf{x}_i = \mathbf{o} + t_i\mathbf{d}\}$ sampled along camera rays $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, where $\mathbf{o} \in \mathbb{R}^3$ denotes ray origin, and $\mathbf{d} \in \mathbb{S}^2$ denotes ray direction.

The expected color $C(\mathbf{r})$ is approximated by a weighted average of point colors around the object surface:

$$C(\mathbf{r}) = \mathbb{E}_{\mathbf{x}_i \in \mathcal{X}_{\text{surf}}} \left[w(\mathbf{x}_i) \Phi(\mathbf{x}_i, \mathbf{d}) \right], \tag{17}$$

$$\mathcal{X}_{\text{surf}} = \{\mathbf{x} || \Omega(\mathbf{x})| < 1\}, \tag{18}$$

$$w(\mathbf{x}_i) = \frac{1}{(1 + e^{-\alpha\Omega(\mathbf{x}_i)})(1 + e^{\alpha\Omega(\mathbf{x}_i)})}, \tag{19}$$

where $\mathcal{X}_{\text{surf}}$ denotes the set of points within truncation distance τ_d to the object surface, $w(\mathbf{x}_i)$ is a bell-shaped function that peaks at object surface, $\alpha = 5$ is a hyper-parameter that controls its sharpness.

Let $z(\mathbf{r})$, $\hat{C}(\mathbf{r})$ be the groundtruth depth and color at training ray $\mathbf{r} \in \mathcal{R}$, $d(\mathbf{x})$ be $\mathbf{x}$'s distance to ray origin $\mathbf{o}$, $\hat{\Omega}(\mathbf{x})$ be the groundtruth untruncated SDF. We supervise (Ω, Φ) with color rendering loss $\mathcal{L}_{\text{render}}$ (denoted $\mathcal{L}_c$ in the main paper, changed to avoid confusion with l_c in consistency loss):

$$\mathcal{L}_{\text{render}} = \mathbb{E}_{\mathbf{r} \in \mathcal{R}} \left[||C(\mathbf{r}) - \hat{C}(\mathbf{r})||_2^2 \right], \tag{20}$$

And SDF loss $\mathcal{L}_{\text{SDF}}$:

$$\mathcal{L}_{\text{SDF}} = \lambda_e \mathcal{L}_e + \lambda_{\text{surf}} \mathcal{L}_{\text{surf}}, \tag{21}$$

$$\mathcal{L}_e = \mathbb{E}_{\mathbf{x} \in \mathcal{X}_e} \left[|\Omega(\mathbf{x}) - 1| \right], \tag{22}$$

$$\mathcal{L}_{\text{surf}} = \mathbb{E}_{\mathbf{x} \in \mathcal{X}_{\text{surf}}} \left[(\Omega(\mathbf{x}) \cdot \tau_d - (z(\mathbf{r}) - d(\mathbf{x})))^2 \right], \tag{23}$$

where $\mathcal{X}_e = \{\mathbf{x} | \hat{\Omega}(\mathbf{x}) > \tau_d\}$ denotes the empty space in front of the object surface, $\Omega(\mathbf{x}) \cdot \tau_d$ is the predicted SDF, $(z(\mathbf{r}) - d(\mathbf{x}))$ approximates groundtruth SDF for points in the near-surface region $\mathcal{X}_{\text{surf}}$. For more stable training, we substitute predicted signed distance $\Omega(\mathbf{x}_i)$ in Eq. (19) with approximated groundtruth signed distance $(z(\mathbf{r}) - d(\mathbf{x}))$.

The total loss for training Neural Object Field in the first stage is

$$\mathcal{L} = \lambda_{\text{render}} \mathcal{L}_{\text{render}} + \lambda_{\text{SDF}} \mathcal{L}_{\text{SDF}} \tag{24}$$

We set $\lambda_{\text{render}} = 10,\ \lambda_{\text{SDF}} = 1,\ \lambda_e = 1\ , \lambda_{\text{surf}} = 6000\ ,$ following [35]. We also build an Octree from depth inputs to speed up ray sampling following their practice.

A.2. Architecture Details

Neural Object Field (Ω, Φ) is implemented with multi-resolution hash encoding [24] of the input position $\mathbf{x}$, spherical embedding of the input view direction $\mathbf{d}$, followed by a 2-layer MLP for TSDF and a 3-layer MLP for color.

The segmentation field $\mathcal{P}(\mathbf{x}, i)$ is implemented with a dense voxel feature grid followed by a 3-layer MLP. The raw P-dim output is activated with softmax to get a probability distribution over P parts. The dense voxel feature grid has size $50 \times 50 \times 50$ and feature dimension $C = 20$.

The hidden dimension of all MLPs is set to 64.

A.3. Training Details

Generating Mesh, SDF, and Occupancy Field We extract meshes from the TSDF field Ω using marching cubes [18] with resolution 0.003. We compute the smoothed occupancy value from SDF with $s = 0.01$ in Eq. (1). For computation efficiency, we pre-computed SDF values at a grid of resolution 0.01, and use trilinear interpolation of the pre-computed values for arbitrary query $\mathbf{x}$.

Handling Visibility We set $\epsilon = 0.03$ in Eq. (15) for visibility computation, and discount invisible corresponding points in the consistency loss with $w_{vis} = 0.5$.

Training Parameters We set $\alpha = 5$ in Eq. (8) while computing point weights for near-surface points in consistency loss, consistent with the setting of α Eq. (19) in the first stage rendering loss. We set $\lambda_s = 10,\ \lambda_c = 0.1,\ \lambda_o = 5$ in consistency loss Eq. (7), $\lambda_{\text{cns}} = 1,\ \lambda_{\text{match}} = 500,\ \lambda_{\text{coll}} = 50$ in total training loss Eq. (16).

We implement our method with PyTorch and use Adam optimizer with an initial learning rate of 0.01 and exponential decay with factor 0.1. Each stage of the reconstruction optimizes for 2000 steps. We also enable occupancy consistency loss and collision loss (both aim at better part seg-

mentation and make more sense when joint parameters are roughly optimized) after 500 optimization steps in the second stage.

Computation Time The optimization part of our method runs for 40 minutes on an NVIDIA Tesla V100 GPU. Precomputing SDF takes another 20 minutes, which we plan to optimize with parallel computation.

A.4. Inference Details

To extract mesh for part i, we run marching cubes on the reconstructed SDF field $\hat{\Omega}(\mathbf{x})$, during which we additionally query the part index of the grid points and set SDF values of points not belonging to part i to 1 (out of the part).

Given the raw optimized 6-DoF rigid transformation $(R, \mathbf{t})$, we classify the joint as prismatic when $|\operatorname{angle}(R_i^0)| < \tau_r$, where the threshold $\tau_r = 10^\circ$. For prismatic joints, we take the translation component $\mathbf{t}$ for axis and part motion computation. For revolute joints, we compute axis direction $\mathbf{u}$ and rotation angle θ from the rotation component R, and compute axis position $\mathbf{p}$ as the solution of $\arg\min_{\mathbf{p}} ||(I - R)\mathbf{p} - \mathbf{t}||_2^2$.

A.5. PARIS*/PARIS*-m Implementation Details

We augment original PARIS with depth supervision following the practice of [2], where the following depth rendering loss is added to the optimization.

$$\mathcal{L}_d = ||\hat{D}(\mathbf{r}) - D_{gt}||_2^2, \tag{25}$$

$$\hat{D}(\mathbf{r}) = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma(t_i)\delta_i))t_i, \tag{26}$$

$$T_i = \exp(-\sum_{j=1}^{i-1} \sigma(t_j)\delta_j), \tag{27}$$

$$\delta_j = t_{j+1} - t_j \tag{28}$$

We use a loss weight of $\lambda_d = 0.01$ since larger weights tend to sabotage the optimization and smaller weights have very limited impact.

The original PARIS composites static field $\mathcal{F}^S$ and mobile field $\mathcal{F}^M$, each representing one object part. It also optimizes the axis and state change of the joint connecting the parts. For objects with P parts, where $P > 2$, we extend PARIS to optimize P fields, $\mathcal{F}^S, \mathcal{F}^{M_0}, \mathcal{F}^{M_1}, \ldots, \mathcal{F}^{M_{P-2}}$, as well as $P - 1$ rigid transformations for each movable part. The per-point color composition in Eq. 1 of PARIS is directly extended to include P terms, namely

$$\hat{C}(\mathbf{r}) = \int_{h_n}^{h_f} (w^S(h) \cdot \mathbf{c}^S(h) + \sum_{i=0}^{P-2} w^{M_i}(h)\mathbf{c}^{M_i}(h))dh. \tag{29}$$

We also closely follow PARIS' fine-tuning procedure and use its proposed regularization loss $\mathcal{L}_{\text{prob}}$ that encourages each point to accumulate information only from one field. Formally,

$$\mathcal{L}_{\text{prob}} = H(P_M(\mathbf{r})), \tag{30}$$

$$P_M(\mathbf{r}) = \frac{O^M(\mathbf{r})}{O^M(\mathbf{r}) + O^S(\mathbf{r})}, \tag{31}$$

$$H(x) = -(x \cdot \log(x) + (1 - x) \cdot \log(1 - x)), \tag{32}$$

where $P_M(\mathbf{r})$ denotes the ratio of the contribution of the mobile field $\mathcal{F}^M$ to ray $\mathbf{r}$, H is the binary entropy function.

For PARIS*-m, we extend $\mathcal{L}_{\text{prob}}$ to be the entropy over the P-ary probability distribution $(P_S, P_{M_0}, \ldots, P_{M_{P-2}})$. Formally,

$$\mathcal{L}_{\text{prob}} = H(P(\mathbf{r})) \tag{33}$$

$$= -\left(P_S(\mathbf{r}) \log P_S(\mathbf{r}) + \sum_{i=0}^{P-2} P_{M_i}(\mathbf{r}) \log P_{M_i}(\mathbf{r})\right). \tag{34}$$

B. Additional Results

B.1. Results on PARIS Two-Part Object Dataset

Being optimization-based methods, PARIS, PARIS*, PARIS*-m, and our approach all have varying performances across trials depending on different initialization of the model parameters. For a comprehensive evaluation, we run each method 10 times with different random seeds (also set randomly) and report their mean and standard deviation in Table 1 of the main paper. While our approach produces quite stable results across trials, PARIS and its variations have large performance variances. For completeness, in Table 4 we summarize the results from the *best* trials of PARIS and PARIS*, alongside the numbers reported in the original PARIS paper, denoted PARIS† for reference. To select the best trial, we compute the minimum value for each metric across 10 trials, then choose the trial with the most number of minimum metric values. When there are ties, we prioritize the metrics with larger variances.

For most objects, the best results from our re-run trials of PARIS are comparable to the reported numbers from PARIS†. However, achieving such results takes many trials and drastic failure cases are not uncommon, as reflected by the overall large average errors. For challenging instances such as stapler, 10 trials are still insufficient to get one successful reconstruction.

The best trials from depth-augmented PARIS* have comparable performance to PARIS on joint-related metrics, and overall better performance on part- and object-level chamfer distances, showing the usefulness of depth supervision. Nevertheless, as suggested in Table 1 of the main paper, the introduction of one more loss term to PARIS' optimization process further destabilizes it and leads to larger variances and average errors.

		Simulation										Real	
		FoldChair	Fridge	Laptop	Oven	Scissor	Stapler	USB	Washer	Blade	Storage	Fridge	Storage
Axis Ang	PARIS† [16]	0.02	0.00	0.03	0.03	0.02	0.07	0.07	0.08	0.00	0.37	1.91	3.88
	PARIS [16]	0.03	0.00	0.00	0.04	0.03	42.99	0.00	0.04	0.11	0.02	1.90	14.61
	PARIS* [16]	0.02	0.00	0.03	0.04	0.00	1.05	0.02	0.08	1.66	0.03	1.91	15.64
Axis Pos	PARIS† [16]	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.02	-	-	0.53	-
	PARIS [16]	0.00	0.00	0.00	0.00	0.00	0.16	0.01	0.01	-	-	0.54	-
	PARIS* [16]	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.02	-	-	0.51	-
Part Motion	PARIS† [16]	0.00	0.00	0.03	0.00	0.00	0.00	0.03	0.08	0.06	0.00	0.77	0.31
	PARIS [16]	0.04	0.00	0.05	0.04	0.03	34.81	0.03	0.18	0.27	0.30	1.48	1.16
	PARIS* [16]	0.04	0.00	0.00	0.03	0.00	0.72	0.00	0.10	0.28	0.30	1.57	0.58
CD-s	PARIS† [16]	0.20	2.88	0.15	6.19	0.28	0.94	2.60	19.45	0.58	11.76	10.22	20.92
	PARIS [16]	0.21	1.92	0.15	10.78	0.39	1.84	2.68	28.15	0.53	8.96	9.32	81.11
	PARIS* [16]	0.21	1.73	0.15	2.73	0.23	1.39	2.47	5.24	0.62	7.68	8.87	16.86
CD-m	PARIS† [16]	0.53	1.13	0.14	0.43	0.23	0.85	0.89	0.27	5.13	20.67	67.54	101.20
	PARIS [16]	0.55	1.43	0.14	7.09	0.25	2.49	0.83	0.24	7.16	83.54	77.48	143.17
	PARIS* [16]	0.41	1.21	0.15	0.76	0.20	0.97	0.62	0.26	6.63	49.14	92.26	12.89
CD-w	PARIS† [16]	0.42	2.68	0.25	6.07	0.30	0.96	1.80	18.31	0.46	8.12	8.20	18.98
	PARIS [16]	0.43	1.95	0.25	9.24	0.33	1.76	1.97	30.30	0.43	7.63	7.84	68.77
	PARIS* [16]	0.37	1.81	0.26	2.68	0.26	1.10	1.64	4.97	0.44	7.40	6.43	15.75

Table 4. Results of the best-performing optimization trials from PARIS and PARIS* on PARIS Two-Part Dataset, where PARIS* [16] is augmented with depth, PARIS† are numbers reported in the original PARIS paper [16] for reference. Note that Blade, Storage, and Real Storage have prismatic joints whose Axis Position Error is undefined.

			Axis Ang 0	Axis Ang 1	Axis Pos 0	Axis Pos 1	Part Motion 0	Part Motion 1	CD-s	CD-m 0	CD-m 1	CD-w
Fridge-m	PARIS*-m [16]	mean$_{\pm std}$	34.52$_{\pm 19.1}$	15.91$_{\pm 7.0}$	3.60$_{\pm 1.6}$	1.63$_{\pm 1.3}$	86.21$_{\pm 55.2}$	105.86$_{\pm 43.6}$	8.52$_{\pm 2.0}$	526.20$_{\pm 141.6}$	160.86$_{\pm 102.2}$	15.00$_{\pm 3.5}$
		best trial	27.30	13.89	3.22	2.27	22.62	43.18	6.74	265.46	150.20	10.45
	Ours	mean$_{\pm std}$	0.16$_{\pm 0.0}$	0.10$_{\pm 0.0}$	0.01$_{\pm 0.0}$	0.00$_{\pm 0.0}$	0.11$_{\pm 0.0}$	0.13$_{\pm 0.0}$	0.61$_{\pm 0.0}$	0.40$_{\pm 0.0}$	0.52$_{\pm 0.0}$	0.89$_{\pm 0.0}$
		best trial	0.11	0.10	0.01	0.00	0.08	0.13	0.61	0.41	0.52	0.89
Storage-m	PARIS*-m [16]	mean$_{\pm std}$	43.26$_{\pm 25.1}$	26.18$_{\pm 7.2}$	10.42$_{\pm 19.1}$	-	79.84$_{\pm 45.4}$	0.64$_{\pm 0.2}$	8.56$_{\pm 1.1}$	128.62*	266.71$_{\pm 102.7}$	8.66$_{\pm 5.4}$
		best trial	0.38	27.54	6.50	-	47.06	0.91	9.15	128.62	216.96	21.95
	Ours	mean$_{\pm std}$	0.21$_{\pm 0.0}$	0.88$_{\pm 0.2}$	0.05$_{\pm 0.0}$	-	0.13$_{\pm 0.0}$	0.00$_{\pm 0.0}$	0.85$_{\pm 0.0}$	0.21$_{\pm 0.0}$	3.46$_{\pm 2.8}$	0.99$_{\pm 0.0}$
		best trial	0.20	0.76	0.05	-	0.13	0.00	0.85	0.21	0.23	0.99

Table 5. Additional result statistics on multi-part object dataset, including the average, standard deviation, and the best result across 10 trials with different random seeds. PARIS*-m [16] is augmented with depth and extended to handle objects with more than two parts. Joint 1 of "Storage-m" is prismatic and does not have Axis Position Error. PARIS*-m only reconstructed two non-empty parts in 9 out of 10 trials, and its CD-m 0 (chamfer distance of movable part 0) is reported from the only trial with three reconstructed parts.

B.2. Results on Multi-Part Objects

For multi-part objects, we follow the same protocol and run 10 trials with different random seeds for both PARIS*-m and our method. Their average, standard deviation, and best trials are summarized in Table 5. Our method continues to exhibit stable performance. On the other hand, PARIS*-m struggles to deal with the increased complexity of multiple movable parts. Even the best trials cannot successfully reconstruct both movable parts or their joint parameters.

B.3. Additional Qualitative Results

Figures 7, 8 show additional qualitative results from PARIS, PARIS*, and our method on PARIS two-part objects. We visualize the per-part reconstructions and reconstructed joint axes. Please refer to our supplementary video for 360° view of the reconstructions and motion interpolation results.

B.4. Demo Interaction with the Reconstructed Multi-Part Storage in Simulation

Our reconstructed digital twin can be readily imported to simulation environments and interacted with. Figure 9 shows screenshots of a demo interaction sequence we made. Please refer to our supplementary video for the full sequence. We imported our reconstruction of the multi-part storage furniture ("Storage-m") into Issac Gym, using both the reconstructed meshes and the joint parameters. We generated a control sequence for a Franka Emika robotic arm to interact with both movable parts of the reconstruction. As illustrated by the demo, the ability to reconstruct digital twins of multi-part articulated objects enables exciting real2sim applications.

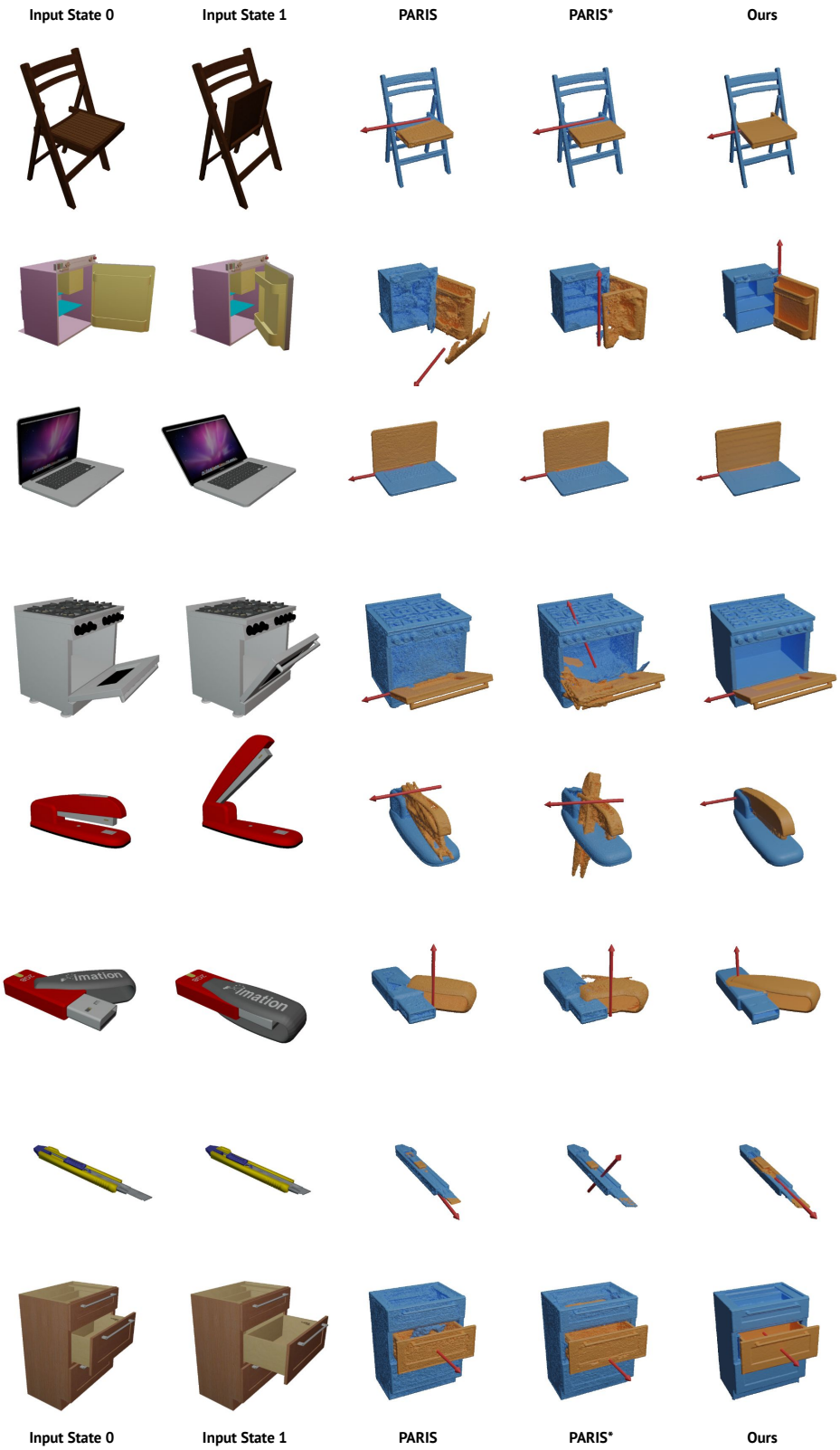

Figure 7. Additional visualizations of reconstruction results from PARIS, PARIS* (PARIS augmented with depth supervision), and our approach on synthetic objects from PARIS Two-Part Object Dataset. For each method, we selected typical trials with performance closest to the average performance. Please refer to our supplementary video for 360° views and motion interpolation results.

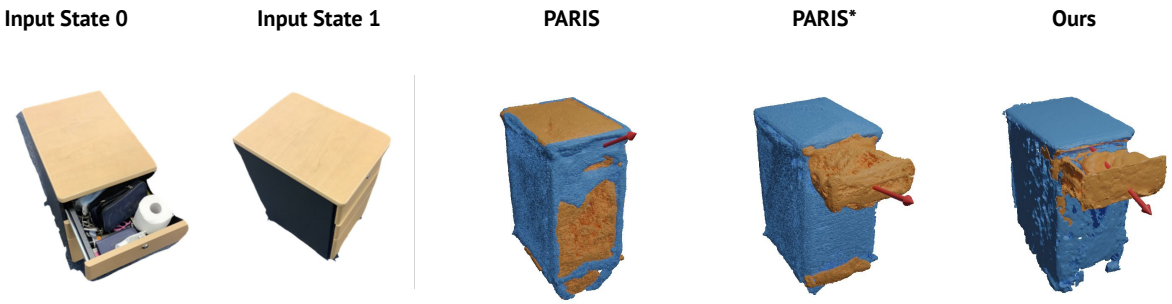

Figure 8. Additional visualizations of reconstruction results from PARIS, PARIS* (PARIS augmented with depth supervision), and our approach on the real storage furniture from PARIS Two-Part Object Dataset. For each method, we selected a typical trial with performance closest to the average performance. Please refer to our supplementary video for 360° views and motion interpolation results.

(a) (b) (c) (d)

Figure 9. Screenshots from demo interaction sequence with our reconstructed storage furniture in Issac Gym. We controlled a Franka Emika robot arm to interact with both movable parts. Please refer to our supplementary video for the full sequence.