FreeArtGS: Articulated Gaussian Splatting Under Free-moving Scenario

Supplementary Material

1. Method Details

This section provides additional implementation and methodological details of FreeArtGS:

- Sec. 1.1: the initialization procedure of the rigid transformations in part segmentation.
- Sec. 1.2: the trajectory filtering strategy to obtain a reliable subset of trajectories given the point-tracking priors.
- Sec. 1.3: the propagation algorithm from optimized part weights to segmentations.
- Sec. 1.4: additional description of part pose estimation.
- Sec. 1.5: the criterion of deciding the joint type.
- Sec. 1.6: the strategy of estimating the joint axis.

Figure 1 further illustrates the detailed data flow of the Part Solver module introduced in Sec. 3.2 of the main paper.

1.1. Transformation Initialization

As described in Sec. 3.2 of the manuscript, our method processes the input video in a window-wise manner. Before each optimization procedure, the rigid transformations must be initialized for the corresponding frame pairs within the window.

For each frame pair $(t, t{+}x)$ with $x \in [1, K-1]$, where K is the window size, let $X_{t,p}$ and $X_{t+x,p}$, with $p \in \mathcal{P}$, denote the 3D positions of trajectory point p at times t and $t{+}x$, respectively. Given an initial partition of the trajectories into two parts, $\mathcal{P}_0$ and $\mathcal{P}_1$, we estimate the rigid transforms $T^0_{t\to t+x}, T^1_{t\to t+x} \in \mathrm{SE}(3)$ by minimizing a robust registration objective:

$$T^k_{t\to t+x} = \arg\min_{T\in \mathrm{SE}(3)} \sum_{p\in\mathcal{P}_k} \rho\big(\|X_{t+x,p} - T X_{t,p}\|_2^2\big), \quad (1)$$

where $\rho(\cdot)$ denotes the Tukey loss. In practice, we realize this step with a RANSAC-based estimator followed by refinement under the Tukey weighting scheme, which increases robustness to outliers.

To further improve the initialization, we perform an EM-style iterative refinement over trajectory assignments and transforms. Let $z_p \in \{0,1\}$ denote the part label of trajectory p. At iteration i, the E-step assigns each trajectory to the transform with the smaller reconstruction error:

$$z_p^{(i+1)} = \arg\min_{k\in\{0,1\}} \|X_{t+x,p} - T^{k,(i)}_{t\to t+x} X_{t,p}\|_2^2$$

In the M-step, the transforms are re-estimated given the updated assignments by solving

$$T^{k,(i+1)}_{t\to t+x} = \arg\min_{T} \sum_{p: z_p^{(i+1)}=k} \rho\big(\|X_{t+x,p} - T X_{t,p}\|_2^2\big)$$

The final estimates are used as the initial transformations $T^0_{t\to t+x}$ and $T^1_{t\to t+x}$ for frame pair $(t, t{+}x)$, providing a robust starting point that reduces the risk of convergence to poor local optima.

1.2. Trajectory Filtering

The off-the-shelf point tracking model [1] can produce noisy trajectories, and directly using all trajectories $\{X_{t,p}\}$ is suboptimal. We therefore apply a multi-stage filtering scheme to obtain a reliable subset of trajectories.

We first retain trajectories that remain visible and inside the foreground masks across frames. Let $c_{t,p}$ denote the visibility confidence of trajectory p at time t predicted by the point tracking model, and let $m_{t,p} \in \{0,1\}$ indicate whether the corresponding pixel lies inside the foreground mask at time t. We define the visibility- and mask-consistent set as

$$\mathcal{S}_{\mathrm{vis}} = \big\{(t,p) \,\big|\, c_{t,p} > \tau_c,\ m_{t,p} = 1,\ m_{t+1,p} = 1\big\}, \quad (2)$$

where $\tau_c = 0.5$ is a visibility threshold.

To further suppress motion outliers, we filter trajectories by displacement magnitude. Let $\Delta X_{t,p} = X_{t+1,p} - X_{t,p}$ and denote the mean and standard deviation of $\|\Delta X_{t,p}\|_2$ over $(t,p) \in \mathcal{S}_{\mathrm{vis}}$ as μ_v and σ_v, respectively. We keep

$$\mathcal{S}_{\mathrm{final}} = \big\{(t,p) \in \mathcal{S}_{\mathrm{vis}} \,\big|\, \|\Delta X_{t,p}\|_2 \le \mu_v + \tau_v \sigma_v\big\},$$

where $\tau_v = 2$ is a displacement threshold.

1.3. Part Segmentation

To propagate the part weights from sparse trajectories to the full-pixel map, we leverage DINO features for semantic-aware interpolation. Let $\phi_t(u)$ denote the DINO feature at pixel u in frame t, and let $\mathcal{U}_t$ be the set of foreground pixels that are not directly covered by the retained trajectories. For each $u \in \mathcal{U}_t$, we first compute the cosine similarity between u and each trajectory point:

$$s_t(u,p) = \cos\big(\phi_t(u), \phi_t(p)\big) = \frac{\phi_t(u)^\top \phi_t(p)}{\|\phi_t(u)\|_2\, \|\phi_t(p)\|_2},$$

The interpolated part weight at pixel u is then defined as the similarity-weighted average

$$\tilde{w}_t(u) = \frac{\sum_p s_t(u,p)\, w_{t,p}}{\sum_p s_t(u,p)}.$$

The resulting dense part weight map $\tilde{w}_t$ is passed to the next window.

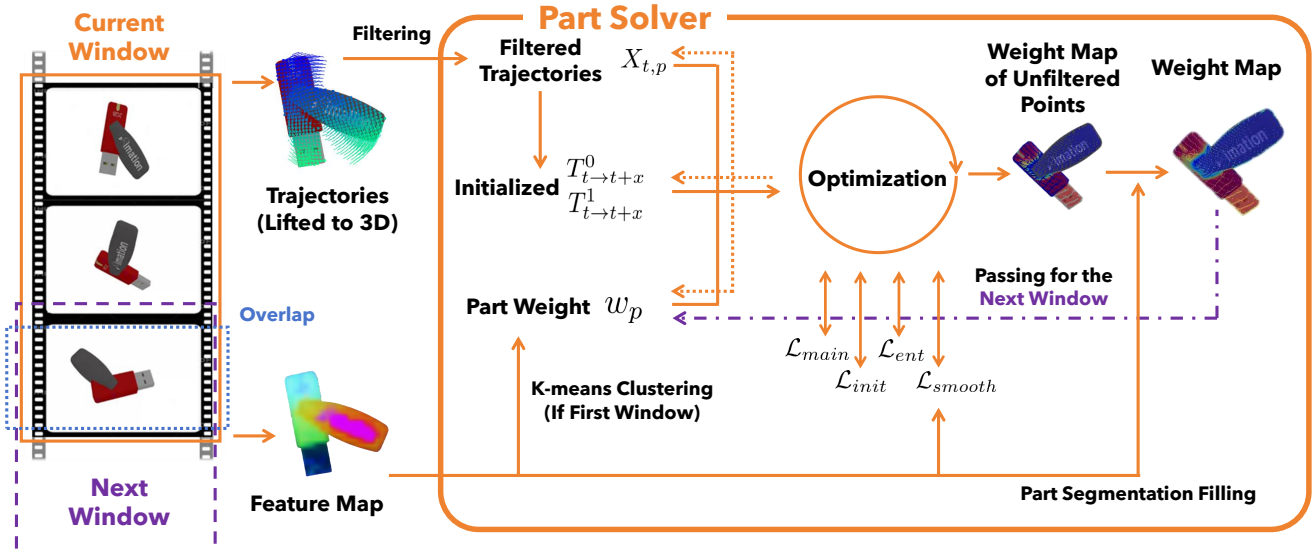

Figure 1. Illustration of Part Solver Module.

1.4. Part Pose Estimation

As previously discussed, we adopt an off-the-shelf approach to estimate part-to-camera poses, exemplified by BundleSDF [3]. For improved accuracy on synthetic datasets, we replace the feature-matching stage in the Coarse Pose Initialization module of the BundleSDF pipeline with an ICP-based procedure. This adjustment is motivated by the fact that objects in simulation typically exhibit extremely limited texture—many are nearly monochromatic—rendering feature matching unreliable. For real-world objects, however, we retain the original BundleSDF pipeline unchanged, as it demonstrates robust performance under such conditions.

1.5. Joint Type Estimation

We propose a simple criterion to decide whether a motion sequence should be modeled as a revolute joint or a prismatic joint, based only on the observed part poses. The decision is made using two scalar features: **Rotation Amplitude** and **Translation Linearity Ratio**. We describe the decision criterion as follows.

Input. We are given a sequence of relative part poses

$$T_i \in SE(3), \quad i = 1, \dots, N,$$

where $T_i = \begin{bmatrix} R_i & t_i \\ 0 & 1 \end{bmatrix}$, $R_i \in SO(3)$, $t_i \in \mathbb{R}^3$. Each R_i is projected to $SO(3)$ to remove numerical noise.

Rotation Amplitude. We first measure how much the object rotates over the whole sequence.

We compute the mean rotation by projecting the sum of rotations to $SO(3)$:

$$S = \sum_{i=1}^{N} R_i, \quad S = U\Sigma V^\top, \quad R_{\text{mean}} = UV^\top \in SO(3).$$

For each frame, we form the relative rotation to the mean,

$$R_i^{\text{rel}} = \text{Proj}_{SO(3)}\big(R_i R_{\text{mean}}^\top\big),$$

and convert it to an angle

$$\theta_i = \arccos\Big(\tfrac{1}{2}(\text{tr}(R_i^{\text{rel}}) - 1)\Big).$$

To obtain a robust rotation span, we apply an IQR-based outlier rejection to the set $\{\theta_i\}$ and then define

$$\Delta\theta = \max_{\theta \text{ inliers}} \theta \; - \; \min_{\theta \text{ inliers}} \theta,$$

and convert it to degrees

$$\Delta\theta_{\text{deg}} = \frac{180}{\pi}\,\Delta\theta.$$

This scalar $\Delta\theta_{\text{deg}}$ measures the overall *rotation amplitude* of the sequence.

Translation Linearity Ratio. Independently, we analyze how linear the translation trajectory is. We first center the translations,

$$\bar{t} = \frac{1}{N}\sum_{i=1}^{N} t_i, \quad x_i = t_i - \bar{t},$$

and build the (unnormalized) covariance matrix

$$C = \sum_{i=1}^{N} x_i x_i^\top \in \mathbb{R}^{3\times 3}.$$

Let its eigenvalues in descending order be

$$\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0.$$

We define a *translation linearity ratio*

$$\rho = \frac{\lambda_2 + \lambda_3}{\lambda_1 + \varepsilon},$$

with a small $\varepsilon > 0$ to avoid division by zero. When the translations lie close to a single straight line, λ_1 dominates and ρ becomes small.

Decision Criterion. Using these two scalar features, we classify the joint type according to

$$\text{model} = \begin{cases} \text{prismatic}, & \text{if } \Delta\theta_{\text{deg}} < \theta_{\text{th}} \text{ or } \rho < \rho_{\text{th}}, \\ \text{revolute}, & \text{otherwise.} \end{cases}$$

In our implementation, we set $\theta_{\text{th}} = 10^\circ$ and $\rho_{\text{th}} = 0.05$.

Intuitively, if the object barely rotates (small $\Delta\theta_{\text{deg}}$), or if its translation is very close to a straight line in 3D (small ρ), the motion is better explained by a prismatic joint; otherwise we adopt a revolute-joint model.

1.6. Joint Axis Estimation

Given a sequence of relative part poses

$$T_i = \begin{bmatrix} R_i & t_i \\ 0 & 1 \end{bmatrix} \in SE(3), \quad i = 1, \ldots, N,$$

our goal is to estimate the underlying joint axis. Depending on the joint type inferred in the previous stage, we adopt two different strategies, described below.

Revolute Model. For a fixed-axis (revolute) joint, all rotations share a common axis direction. Let

$$R_{ij} = \text{Proj}_{SO(3)}(R_i R_j^\top)$$

denote the relative rotation between frames i and j, where $\text{Proj}_{SO(3)}$ removes numerical drift. In an ideal revolute motion, the unknown axis direction u satisfies

$$R_{ij} u = u \quad \Longleftrightarrow \quad (R_{ij} - I)\, u = 0 \quad \forall\, i < j.$$

Thus u lies in the approximate null space shared by all matrices $(R_{ij} - I)$. We therefore minimize the quadratic error

$$u^\star = \arg\min_{\|u\|=1} \sum_{i<j} \|(R_{ij} - I)u\|^2.$$

Expanding the objective yields the symmetric matrix

$$A = \sum_{i<j} (R_{ij} - I)^\top (R_{ij} - I),$$

whose eigenvector associated with the smallest eigenvalue gives the optimal axis direction,

$$Au^\star = \lambda_{\min} u^\star, \qquad \|u^\star\| = 1.$$

Once the axis direction is known, the axis point p is estimated from the translational consistency equation for a rigid rotation without screw pitch:

$$t_i \approx c + (I - R_i)p.$$

Subtracting two frames removes the global offset c:

$$t_i - t_j \approx (R_j - R_i)p.$$

Since the component of p along u is unobservable, we solve p in the plane orthogonal to u. Let $P = I - uu^\top$ be the projection onto $u^\perp$, then

$$P(t_i - t_j) \approx P(R_j - R_i)p.$$

Stacking all such constraints yields a least-squares system, whose solution is projected back onto $u^\perp$ to enforce $u^\top p = 0$, giving a unique axis point closest to the origin.

Prismatic Model. For a prismatic joint, the rigid body exhibits negligible rotation while its translations $\{t_i\}$ lie approximately on a straight 3D line. We estimate the sliding direction and axis point from the translational trajectory.

Translations are first centered:

$$\bar{t} = \frac{1}{N} \sum_{i=1}^{N} t_i, \qquad x_i = t_i - \bar{t}.$$

The covariance matrix

$$C = \sum_{i=1}^{N} x_i x_i^\top$$

is decomposed as

$$Cv_k = \lambda_k v_k, \qquad \lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0.$$

If the motion is prismatic, λ_1 dominates, and the first principal component encodes the translation line direction. Thus the prismatic axis is

$$w^\star = \frac{v_1}{\|v_1\|}.$$

The axis point p_0 is the closest point on this line to the origin, obtained by projecting the mean translation onto $w^\star$:

$$p_0 = \bar{t} - (\bar{t}^\top w^\star)\, w^\star.$$

Table 1. Results of Robustness Analysis.

Joint Type	Method	Axis(deg)↓	Position(cm)↓	State(deg/cm)↓	CD-w(cm)↓	CD-m(cm)↓	CD-s(cm)↓	PSNR(dB)↑
Revolute	2% Depth Noise	1.41±1.49	0.51±0.58	1.22±1.27	0.23±0.30	0.30±0.27	1.05±2.47	23.52±4.65
	AllTracker → CoTracker3	**0.88±0.84**	0.46±0.86	1.68±1.30	0.79±2.19	0.80±1.08	1.47±2.43	23.87±4.94
	DINOv3 → DINOv2	1.64±1.22	0.32±0.47	**1.21±0.77**	**0.11±0.11**	**0.28±0.27**	1.04±2.52	23.74±4.41
	Ours	1.04±1.03	**0.29±0.36**	1.43±1.20	0.14±0.13	0.28±0.36	**0.97±2.69**	**24.02±3.15**
Prismatic	2% Depth Noise	1.86±2.51	-	0.92±0.66	**0.34±0.13**	1.07±1.18	1.78±3.03	21.75±2.99
	AllTracker → CoTracker3	1.98±3.09	-	1.12±0.68	0.79±1.08	0.98±1.22	0.94±1.18	21.85±3.64
	DINOv3 → DINOv2	**1.46±1.63**	-	0.96±1.59	0.56±0.40	0.99±0.93	0.46±0.51	22.39±3.92
	Ours	1.85±2.75	**-**	**0.90±0.53**	0.41±0.22	**0.67±0.34**	**0.30±0.13**	**22.92±2.65**

Each frame's joint displacement is then the signed projection

$$d_i = (t_i - p_0)^\top w^\star,$$

which, together with a constant rotation offset, fully characterizes the prismatic motion.

2. Benchmark Details of FreeArt-21

We provide further details about the FreeArt-21 benchmark in this section. All data are collected through tele-operation within the Sapien simulator [4]. A human operator simultaneously controls the 6-DoF poses and joint parameters of articulated objects using a PICO 4 Ultra VR headset and controllers, enabling real-time manipulation and interaction with the objects. The camera is positioned approximately 3 meters away from the object in simulation, providing a realistic perspective for the data collection. Each case in the benchmark contains multi-modal data, including RGB images, depth maps, ground-truth object and part masks, as well as the 6-DoF object poses and joint parameters. These elements provide rich supervision for training and evaluating object pose estimation and articulated manipulation tasks. The data collection involves between 200 and 400 frames per sequence, capturing diverse interaction patterns.

Compared to other datasets, FreeArt-21 offers more human-like data collection, as the sequences are captured through human teleoperation, rather than automated generation or pre-set trajectories, making it more representative of real-world scenarios where objects are scanned by a person. The objects included in FreeArt-21 are shown in Figure 3.

3. Additional Experiment Results

3.1. Robustness Analysis

We further evaluate the robustness of our framework by replacing AllTracker with CoTracker3 and DINOv3 with DINOv2, while also injecting 2% Gaussian noise into the depth of the FreeArt-21 dataset. As shown in Table 1, the results indicate that FreeArtGS remains robust to noisy depth inputs and variations in predictions from upstream models.

3.2. Camera Pose Estimation Results

We evaluate our camera pose estimation results on the Video2Articulation-S dataset and compare them with the baseline method, Video2Articulation [2]. Camera pose is a crucial variable that is jointly optimized in our end-to-end pipeline. As shown in Table 2, our method achieves significantly lower rotation and translation errors than the baseline, resulting in a more accurate reconstruction of the articulated objects. We also provide trajectory visualizations in Figure 2 to qualitatively show the alignment between estimated and ground-truth trajectories.

Table 2. Results of Camera Pose Estimation on the Video2Articulation-S Dataset. * indicates that the results are taken from the original paper.

Method	Rot. Error (deg)↓	Pos. Error (cm)↓
Video2Articulation* [2]	4.621	9.246
Ours	**2.483**	**1.526**

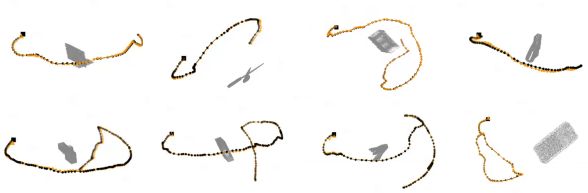

Figure 2. Visualization of camera pose trajectories in FreeArt-21. The black line shows the ground truth camera trajectory, and the yellow line shows our estimated camera trajectory.

3.3. Part Segmentation Results

We additionally report the evaluation of our part segmentation. The metrics are calculated in terms of mIoU of the two parts in Table 3. This result confirms that our model produces consistent part segmentations that align closely with the annotated masks.

Table 3. Results of Part Segmentation. * indicates that the results are taken from the original paper.

Method / mIoU (%)↑	FreeArt-21	Video2Articulation-S
Video2Articulation [2]	34.61	59.65*
Ours	**90.88**	**84.92**

3.4. More Visualization on FreeArt-21

We visualize more reconstruction results on FreeArt-21 in Figure 4.

4. Failure Case Analysis

Part Segmentation Failures. Our pipeline's efficacy is closely tied to the accuracy of part segmentation. In cases involving narrow or elongated structures or objects with extremely low texture, the tracked 3D trajectories often exhibit significant drift. These deviations potentially bias the motion-based clustering toward incorrect part assignments. Such mis-segmentation directly undermines the subsequent joint estimation and geometry reconstruction.

Part Pose Ambiguities. The precision of articulation axis estimation and surface reconstruction is also highly sensitive to part pose accuracy. While we initialize poses using off-the-shelf methods and refine them during optimization, thin or planar objects (e.g., monitors or scissors) present significant challenges. The scarcity of reliable feature correspondences on these geometries often leads to poorly constrained pose optimization. Consequently, the resulting articulation axes may deviate from the physical ground truth, and the reconstructed geometry may suffer from noticeable structural artifacts.

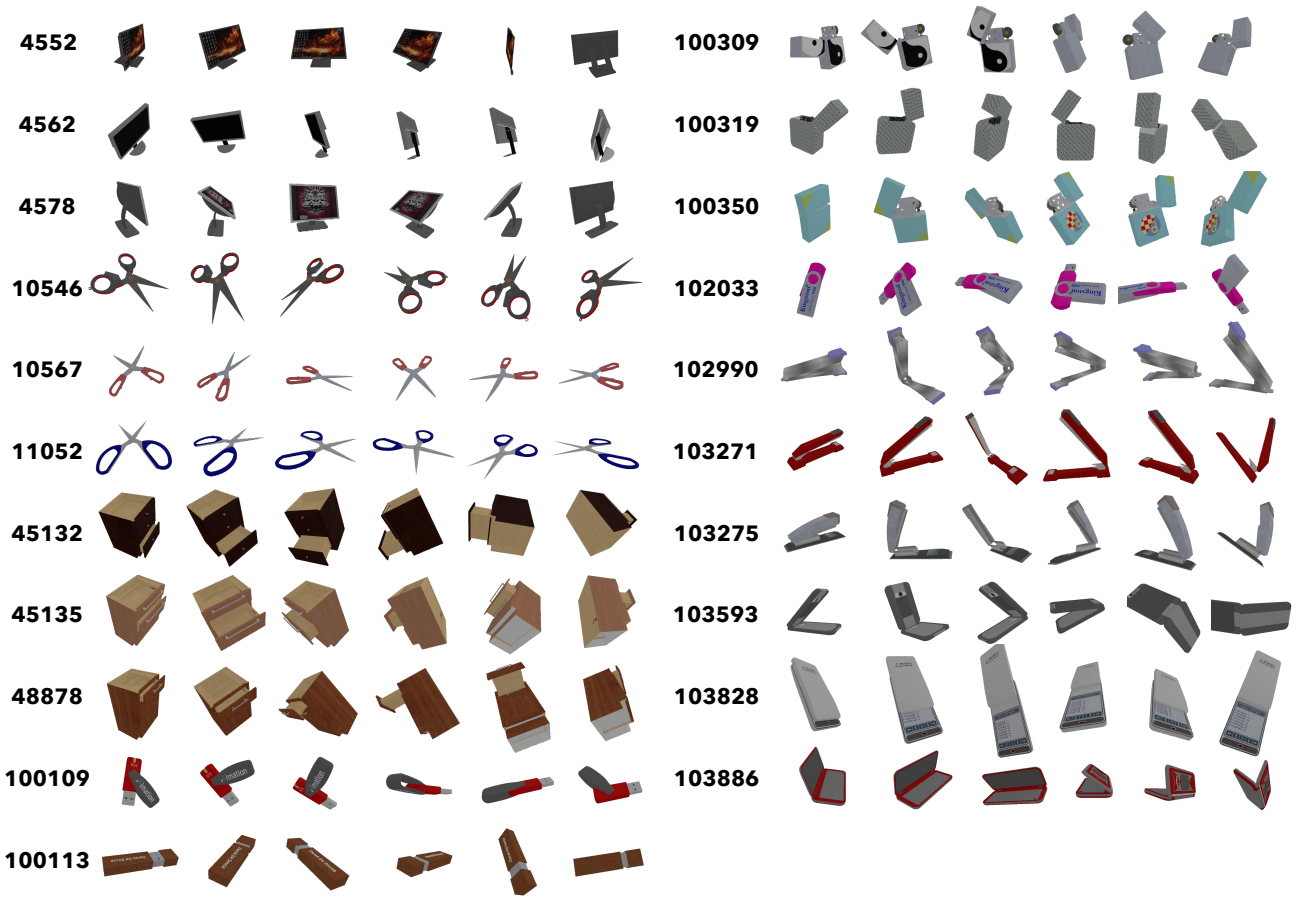

Figure 3. Visualization of all objects in FreeArt-21.

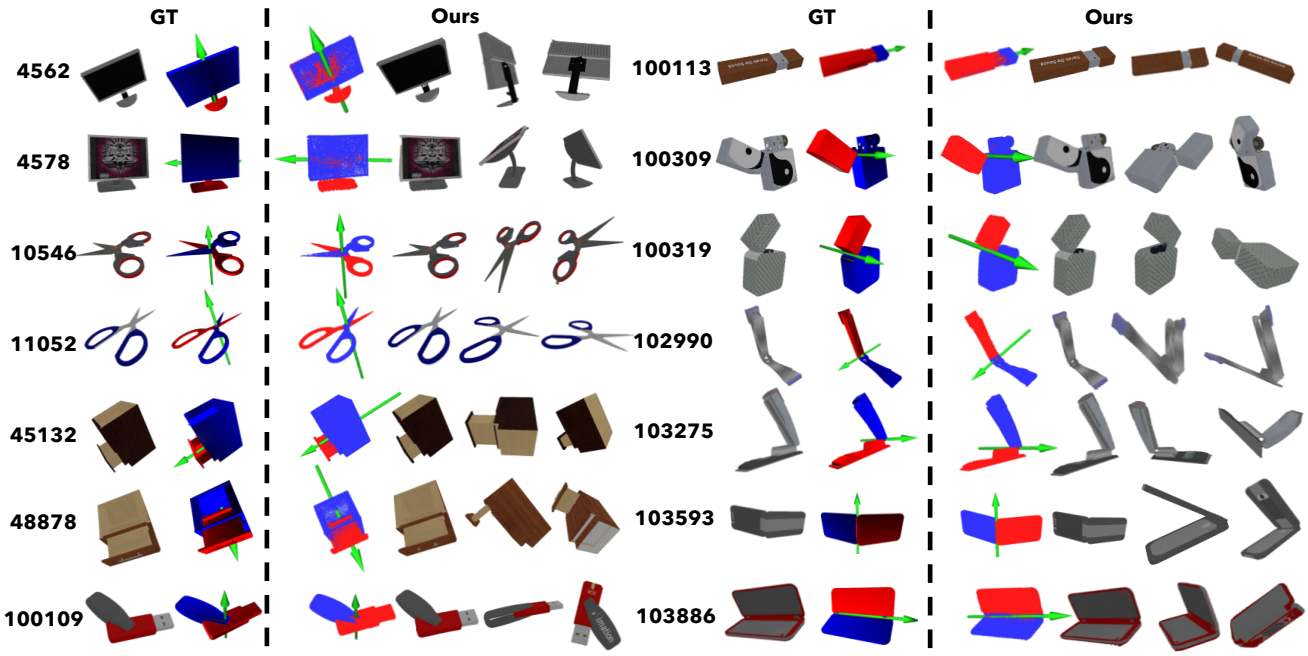

Figure 4. Visualization of our reconstruction results on FreeArt-21.

References

[1] Adam W Harley, Yang You, Xinglong Sun, Yang Zheng, Nikhil Raghuraman, Yunqi Gu, Sheldon Liang, Wen-Hsuan Chu, Achal Dave, Suya You, et al. Alltracker: Efficient dense point tracking at high resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5253–5262, 2025. 1

[2] Weikun Peng, Jun Lv, Cewu Lu, and Manolis Savva. itaco: Interactable digital twins of articulated objects from casually captured rgbd videos, 2025. 4, 5

[3] Bowen Wen, Jonathan Tremblay, Valts Blukis, Stephen Tyree, Thomas Müller, Alex Evans, Dieter Fox, Jan Kautz, and Stan Birchfield. Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 606–617, 2023. 2

[4] 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. 4