ArticulatedGS: Self-supervised Digital Twin Modeling of Articulated Objects using 3D Gaussian Splatting

Junfu Guo [1] Yu Xin[1] Gaoyi Liu[1] Kai Xu[2] Ligang Liu [1] Ruizhen Hu [3*]

[1]University of Science and Technology of China
[2]National University of Defense Technology
[3]Shenzhen University

Abstract

We tackle the challenge of concurrent reconstruction at the part level with the RGB appearance and estimation of motion parameters for building digital twins of articulated objects using the 3D Gaussian Splatting (3D-GS) method. With two distinct sets of multi-view imagery, each depicting an object in separate static articulation configurations, we reconstruct the articulated object in 3D Gaussian representations with both appearance and geometry information at the same time. Our approach decoupled multiple highly interdependent parameters through a multi-step optimization process, thereby achieving a stable optimization procedure and high-quality outcomes. We introduce ArticulatedGS, a self-supervised, comprehensive framework that autonomously learns to model shapes and appearances at the part level and synchronizes the optimization of motion parameters, all without reliance on 3D supervision, motion cues, or semantic labels. Our experimental results demonstrate that, among comparable methodologies, our approach has achieved optimal outcomes in terms of part segmentation accuracy, motion estimation accuracy, and visual quality. The code will be made publicly available at our website https://guojunfu-tech.github.io/articulatedGS-io/

1. Introduction

Articulated objects, such as laptops, ovens, and drawers, are a part of our daily life. Building the digital twin of an articulated object with accurate reconstruction of and fine-grained analysis often plays an important role in robotics [24, 27, 31, 47], animation [19, 44], and simulation [41, 45]. Moreover, it is also desirable to equip the digital twins with color appearance to facilitate downstream applications such as training sim-to-real transferable embodied AI [1, 34, 35]. Consequently, the capturing and reconstruction of *shape geometry*, *visual appearance*, and *articulation parameters* is indispensable in building accurate and functional digital twins of articulated objects.

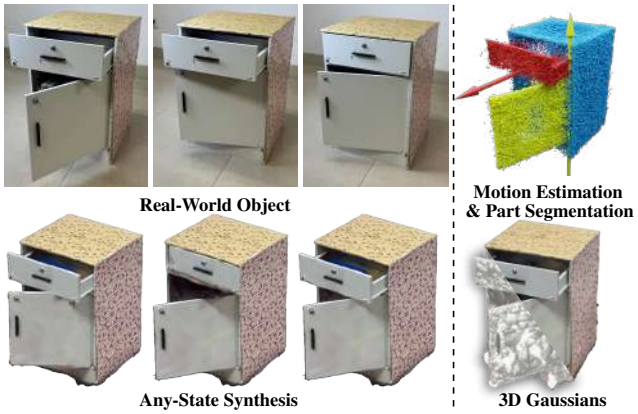

Figure 1. **ArticulatedGS** is a self-supervised modeling and reasoning pipeline to reconstruct the articulated objects using 3D Gaussian Splatting.

The problem of modeling high-fidelity digital twins of articulated objects is receiving increasing attention in recent years [10, 21, 22, 33]. Most of the previous works [10, 16, 21] are concentrated on the regeneration or construction of 3D geometrical representation of articulated objects, which may involve the estimation of articulation parameters or functionalities to animate the objects into different articulation states. The result is usually a mesh-based representation of articulated objects with relatively low geometric fidelity and visual realism. Moreover, the reliance on supervised learning, which typically consumes a large 3D dataset with articulation annotation, also limits the generalizability to unseen object categories. To our knowledge, the only *self-supervised* work that takes RGB observations as input is PARIS [21]. PARIS adopts the neural radiance field (NeRF) as an intermediate representation and outputs a

*Corresponding author

mesh with compromised accuracy of geometry and appearance. Additionally, it exhibits instability and suboptimality issues in the optimization process.

Recently, there has also been a massive success of 3D Gaussian Splatting (3D-GS) [12] in 3D scene learning and novel view synthesis as well as extensions to various applications such as 3D reconstruction [2, 17, 32] and dynamic scene rendering [3, 6]. 3D-GS requires only RGB images as input and enables rapid scene reconstruction with real-time and realistic rendering. The reconstructed scene is composed of numerous Gaussians densely distributed around the object's surface, thereby allowing for modeling rich geometric details. Recent works [25, 42, 43] make it possible to interact with digital twins represented with 3D-GS for physically meaningful simulation and manipulation. The fast and realistic rendering of 3D-GS also benefits learning in simulation. For those purposes, what has been missing is to reconstruct digital twins of articulated objects with the representation of 3D Gaussians. In this work, we aim to take the RGB observations of two states of a given articulated object as an input and reconstruct a 3D-GS model of the object with *geometry*, *appearance* and *mobility*.

Optimizing for object geometry (masks of movable parts), appearance, and motion parameters simultaneously from only two-view inputs is a highly ill-posed problem. This is because the three terms are deeply intertwined. The object's movable parts, undergoing rigid body transformations based on motion parameters, can greatly impact on the learning of part geometry and visual appearance. The same goes for the opposite way; the optimization of the movable mask and the motion parameters can be significantly affected by rendering quality. To that end, we propose to decouple the optimization process into multiple subprocesses. We first reconstruct the 3D-GS model of the start state. We then train a network to estimate a deformation flow for moving the movable Gaussians and initialize the geometry structure (movable mask) for the end state. Based on the initial movable mask, we leverage the geometry structure to supervise the learning of the motion parameters. At last, we jointly update the appearance, motion parameters, and movable mask to obtain the final reconstruction of the articulated object.

Evaluation on an extended dataset from the previous works [7, 21] demonstrates that our method achieves the state-of-the-art performance of visual quality, motion parameter estimation, and part segmentation. Our method also shows enhanced efficiency in model training and more stability of the optimization process compared to the previous self-supervision methods. Our work makes the following contributions:

- A 3D Gaussian model for building digital twins of articulated objects in a self-supervised fashion;
- A progressive optimization scheme to the challenging optimization task with efficiency and stability;
- New state-of-the-art results of reconstruction quality, motion estimation, and visual quality for articulated objects.

2. Related work

Articulated Modeling. The primary objective of articulated object reconstruction is to accurately recover the three-dimensional (3D) geometry of articulated entities. Pekelny and Gotsman [29] introduced one of the pioneering works in this area, which focuses on reconstructing the surface from a sequence of point clouds. A-SDF [28] represents an innovative approach, the inaugural study to reconstruct articulated objects from signed distance fields (SDFs) derived from a solitary snapshot. This technique permits the deformation of objects into any conceivable articulation state. While echoing the goals of A-SDF, the work of [38] diverges in their approach by utilizing multi-view RGB images for reconstruction purposes. CARTO [8] and [15] advances this concept further by incorporating a single stereo RGB image as input. A common limitation across these three studies is their tendency to reconstruct the articulated object in its entirety as a unified surface. To counteract this limitation, CLANeRF [36], Ditto [10], and PARIS [21] have each developed methodologies that reconstruct articulated objects at the part level. Both CLANeRF and PARIS utilize multi-view RGB images as their input, whereas Ditto processes point cloud data. To overcome the instability of the optimization process of PARIS, the DTA [39] and ArtGS[23] take depth as extra information to achieve the same goal but higher performance. This work utilizes multiple modules and incorporates prior knowledge such as image feature extraction, but the output of this method does not inlcude appearance. In contrast, our approach requires only RGB data and employs a fully self-supervised method, with an optimization time that is approximately one-third of theirs, yet still achieves similar quality results. In our work, we adopt the same RGB series input as PARIS and solve the stabilization problem while yielding a more accurate result with higher visualization quality. Moreover, the Gaussian representation we utilize can, compared to PARIS, quite naturally encompass both appearance and geometry characteristics. For more complete coverage of works in this field, please see the recent surveys [22].

Deformable 3D Gaussians. Gaussian Splatting [12] offers improved rendering quality and speed for radiance fields. Several concurrent works have extended 3D-GS from representing static scenes to deformation areas. [4, 11, 14, 20, 26, 40, 46] use the deformation field to represent the GS deformation with time varies. Given the RGB observations at dense time frames, the deformation field takes the position of each GS and the time frame and predicts the deformation of each GS in position, rotation, and scaling. The

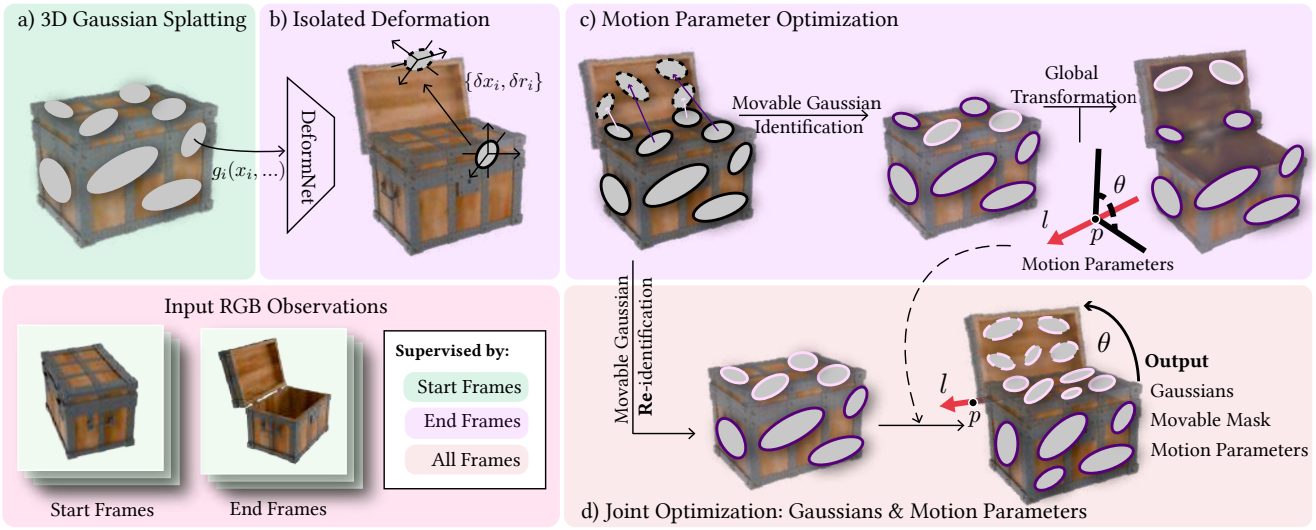

Figure 2. A pipeline overview of our self-supervised 3D Articulated Objects modeling method. (a) Given the sets of RGB observation sequence of two states I_0 and I_1 of an arbitrary articulated object, we first reconstruct the start state using 3D Gaussian Splatting method into the initial Gaussians $\mathcal{G} = \{g_i\}_{i=1}^{N}$. (b) Each Gaussian g_i is then fed into DeformNet, where its Gaussian kernel's 3D coordinates $\boldsymbol{x}_i$ is input to predict its individual rigid motion $\{\delta \boldsymbol{x}_i, \delta \boldsymbol{r}_i\}$ within a local coordinate system, with the end state images I_1 serving as visual supervision. (c) Based on the transformation predicted by DeformNet, we classify the displacements after regularization to obtain an initial moving Gaussian, and then apply the motion parameters to this moving Gaussian to perform a global rigid transformation, which is supervised by I_1, ultimately resulting in a motion category and converged motion parameters. (d) Subsequently, we optimize the preprocessed Gaussians, motion parameters, and the movable mask in tandem to achieve the final results.

work of [5] introduced a mesh-based approach to combine the mesh with 3D Gaussians, leveraging the mesh to guide the splitting process and enhancing the overall quality of the learned GS. However, these methods are highly dependent on the density of frames; when frames are sparse, the Gaussian motion process between frames cannot be accurately predicted. In our work, the movable Gaussians deform together with the motion parameters, remaining visually and geometrically consistent between the two states.

Another line of works focuses on using 3D-GS to model the organic articulated objects, such as human [11, 13, 17], body parts [2, 11, 30], and animals [17]. The motion allowed for these objects is usually structurally complex, with a relatively large number of joints varied in the topology of connections and degrees of freedom. However, the kinematic structure is relatively fixed and need to be predefined for a specific species.

3. Method

3.1. Problem Statement

In this work, we take two sets of multi-view RGB images $\mathcal{I}_0 = \{I_0^i\}$ and $\mathcal{I}_1 = \{I_1^i\}$, together with the corresponding camera poses, as input to represent the start and end states of the target articulated object, respectively. Our goal is to use a set of 3D Gaussians associated with part segmentation and articulated motion parameters to reconstruct such an articulated object.

To formally define the problem, we first explain how the part-aware 3D Gaussians and the motion parameters are represented, and then define the optimization objective.

Part-aware 3D Gaussian Representation. In the default setting, we assume that there is only one *movable* part during the movement between two states, and the rest of the object is *unmovable*, and use a movable mask to distinguish those two types of Gaussians. In more details, the classical 3D Gaussian can be represented as $\mathcal{G} = \{g_i = (\boldsymbol{x}_i, \boldsymbol{r}_i, \boldsymbol{s}_i, \sigma_i, sh_i) | i = 1...N\}$, where each Gaussian g has a center position $\boldsymbol{x}$, the rotation matrix represented by a quaternion $\boldsymbol{r} \in \mathbb{SO}(3)$, S as a scaling matrix represented by a 3D vector $\boldsymbol{s}$, and view-dependent appearance represented by spherical harmonics (SH) parameters sh. In our method, for each g_i, we add an extra element m_i to present the movable mask such that the g_i is a movable part if $m_i \neq 0$. Therefore, our part-ware 3D Gaussian representation $\mathcal{G}$ can be denoted as $\mathcal{G} = \{\mathcal{G}_m, \mathcal{G}_u\}$, where $\mathcal{G}_m = \{g_i | m_i \neq 0\}$ and $\mathcal{G}_u$ represents the complement.

Articulated Motion Representation. Following previous works[10, 18, 21], we consider two types of joints: *prismatic* and *revolute*. The prismatic joint is parameterized by a 3D translation axis $\boldsymbol{a} \in \mathbb{R}^3, ||\boldsymbol{a}|| = 1$ and a move distance

d along this axis; And the revolute joint is parameterized by a 3D rotation orientation axis $\boldsymbol{l} \in \mathbb{R}^3, ||\boldsymbol{l}|| = 1$, a pivot $\boldsymbol{p} \in \mathbb{R}^3$ that locates the axis in the world coordinates, and the rotate angle θ.

With the motion parameters above, we can construct rigid-body transformation functions for revolute joint on each Gaussian by $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{r}}) = f_{\boldsymbol{l},\boldsymbol{p},\theta}(\boldsymbol{x}, \boldsymbol{r})$, and prismatic joint, $\hat{\boldsymbol{x}} = f_{\boldsymbol{a},d}(\boldsymbol{x})$, allowing each Gaussian $g(\boldsymbol{x}, \boldsymbol{r})$ to calculate its post-motion $\hat{g}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{r}})$. In this way, the transformed set of Gaussians can be represented as

$$\hat{\mathcal{G}} = \{\mathcal{G}_u, f(\mathcal{G}_m)\}, \tag{1}$$

where $f \in \{f_{\boldsymbol{l},\boldsymbol{p},\theta}, f_{\boldsymbol{a},d}\}$.

Optimization formulation. With the part-aware Gaussians and motion parameters, our goal is to optimize them all together to achieve high-fidelity rendering of images similar to the input RGB images across different angles and states. Thus, our optimization problem can be formulated as

$$\underset{\mathcal{G},f}{\operatorname{argmin}} \sum_i \mathcal{L}_{app}\big(I_0^i(\mathcal{G}) - I_0^i\big) + \sum_j \mathcal{L}_{app}\big(I_1^j(\hat{\mathcal{G}}) - I_1^j\big), \tag{2}$$

where $I(\mathcal{G})$ represents the image rasterized by Gaussian $\mathcal{G}$ with the camera extrinsic and intrinsic of image I, and $\mathcal{L}_{app}$ is the same appearance loss function used in the 3D Gaussian Splatting method [12]:

$$\mathcal{L}_{app} = (1-\lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{D\_SSIM}. \tag{3}$$

3.2. Overview

Given the complex problem formulated in Eq. (2), optimizing the motion parameters, the movable mask, and the Gaussian representations concurrently can result in a tendency to settle for a local optimum or to face computational failure. To address this, we introduce a novel process that methodically tackles the overarching objective in a stepwise fashion, each step aimed at securing initial parameter estimates that are proximal to the ground truth, thus enhancing the likelihood of reaching the optimal solution.

The pipeline of our method is illustrated in Fig. 2. We first train a set of static Gaussians $\mathcal{G}$ with only viewpoints I_0 from the start frames. Then, we use a deformation network to predict each Gaussian's translation and rotation, supervised by I_1. With the deformed Gaussians, we classify the movements and obtain the initial movable Gaussians $\mathcal{G}_m$ and unmovable Gaussians $\mathcal{G}_u$. After that, We optimize the motion parameters in $\mathbb{SE}(3)$ by casting the rigid movement to $\mathcal{G}_m$. During this step, we classify the motion type according to the motion parameters. With the pre-processed motion parameters and movable mask, we jointly optimize the Gaussians' representation, the movable mask, and the motion parameters to obtain the final result. Below we will explain the details of each step.

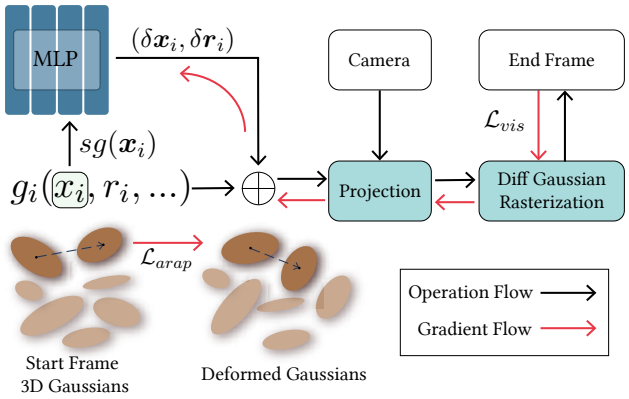

Figure 3. The process diagram of DeformNet illustrates our approach where a neural network is utilized to predict the rigid transformation of a Gaussian. The input consists of the kernel's positions $\boldsymbol{x}_i$ for each Gaussian g_i, and the output is the displacement and relative rotation to the end state. In addition to visual supervision by I_1, we also impose geometrical supervision $\mathcal{L}_{arap}$ that maintains local consistency among the local Gaussians.

3.3. Isolated Deformation Prediction

Fig. 3 gives an illustration of our isolated deformation Process. With the trained Gaussians of start frames, we do not optimize the motion parameters directly, but use a network $f_{\boldsymbol{\theta}}$ to predict the deformation field of the Gaussians. The deformation field takes the position of each 3D Gaussian $\boldsymbol{x}_i$ as input, and output $\delta\boldsymbol{x}_i$ and $\delta\boldsymbol{r}_i$, which represent the translation and rotation of Gaussian $\boldsymbol{g}_i$:

$$(\delta\boldsymbol{x}_i, \delta\boldsymbol{r}_i) = f_{\boldsymbol{\theta}}(sg(\boldsymbol{x}_i)). \tag{4}$$

Note that as we do not want to change the origin location of each Gaussian but only predict the deformation field, we add a stop-gradient operation $sg(\cdot)$ to $\boldsymbol{x}_i$. The network architecture is relatively simple, which consists of a four-layer one-dimensional Convolutional Network (1D CNN) connected to a four-layer Multi-Layer Perceptron (MLP).

The deformed 3D Gaussians $\hat{\mathcal{G}} = \{\hat{g}_i = (\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{r}}_i, ...) | \hat{\boldsymbol{x}}_i = \boldsymbol{x}_i + \delta\boldsymbol{x}_i, \hat{\boldsymbol{r}}_i = \boldsymbol{r}_i + \delta\boldsymbol{r}_i\}$ is then put into the differential Gaussian rasterization pipeline to get the rendered image $\hat{I} = I(\hat{\mathcal{G}})$. The appearance loss $\mathcal{L}_{app}$ (defined in Eq. (3)) is then computed between the rendered image $\hat{I}$ and the target image I_1.

Moreover, with the prior that the desired deformation should be part-wise rigid, we add an extra as-rigid-as-possible (ARAP) loss $\mathcal{L}_{arap}$ as in previous works [9, 26] to encourage the motion to be locally rigid:

$$\mathcal{L}_{arap} = \frac{1}{k|\mathcal{S}|} \sum_{i \in \mathcal{S}} \sum_{j \in knn_{i,k}} \mathcal{L}_{arap}^{i,j}, \tag{5}$$

$$\mathcal{L}_{arap}^{i,j} = \omega_{i,j} ||(\boldsymbol{x}_j - \boldsymbol{x}_i) - R_i \hat{R}_i^{-1} (\hat{\boldsymbol{x}}_j - \hat{\boldsymbol{x}}_i)||_2. \tag{6}$$

Here, we restrict the set of Gaussians j to be the k-nearest-neighbor of i ($k = 20$), and the weight factor for the Gaussian pair is defined as:

$$\omega_{i,j} = exp\big(-\lambda_\omega ||\boldsymbol{x}_j - \boldsymbol{x}_i||_2^2\big) \tag{7}$$

where $\lambda_\omega = 20$ in our experiments.

Therefore, the final loss function is the combination of the appearance loss and the ARAP loss:

$$\mathcal{L}_{deform} = \mathcal{L}_{app} + \lambda_{arap}\mathcal{L}_{arap}, \tag{8}$$

where $\lambda_{arap} = 1$ in our experiments.

3.4. Motion Parameter Optimization

Once we obtain the deformation field of all the Gaussians, we can first roughly identify the movable part and estimate the motion parameters based on their deformations. Note that the movable part doesn't need to be very accurate at this step, as the estimated motion parameters could be further refined in the following joint optimization, and our goal here is to get an good approximation as initialization for joint optimization.

To identify the movable part, we first normalize the displacement $||\delta\boldsymbol{x}||_2$ within 0 and 1, and take the Gaussians with $\delta\boldsymbol{x} \geq \tau$ as the movable Gaussians $\mathcal{G}_m$. Based on our observations, selecting a lower threshold τ often results in the inclusion of stationary Gaussians, which can lead to sub-optimal outcomes when optimizing motion parameters later on. Thus, we tend to set a higher threshold to get a partial set of movable parts with higher confidence, which is usually already enough for motion parameter prediction for the entire part. We set $\tau = 0.3$ in our experiments to classify the movable Gaussians.

With the movable Gaussians $\mathcal{G}_m$, we proceed to optimize the motion parameters by applying a global rigid transformation to $\mathcal{G}_m$. We observed that when fitting a prismatic motion with revolute motion parameters, the pivot of the revolute parameters typically moves a considerable distance to ensure that the movable Gaussians undergo minimal rotation during the motion. Therefore, we initiate with optimizing the revolute parameters. The motion is considered to be prismatic if the pivot $||\boldsymbol{p}|| > R$, where $R = 1$ in our experiments, and we will reinitialize the optimization parameters and the motion function to align with the prismatic motion.

We utilize both appearance and geometric supervision to refine the motion parameters:

$$\mathcal{L}_{param} = \mathcal{L}_{app} + \mathcal{L}_{geo}. \tag{9}$$

For the appearance, we maintain the use of the end frame for supervision, and thus $\mathcal{L}_{app}$ is the same as the one used for the Deformation Network. For the geometric supervision, we preserve the Gaussian points that have undergone transformation by the Deformation Network and calculate the Chamfer distance between the coordinates of these two sets of points to provide geometric guidance. Therefore, the geometric loss is defined as:

$$\mathcal{L}_{geo} = d_c\big(f(\boldsymbol{x}), \boldsymbol{x} + \delta\boldsymbol{x}\big), f \in \{f_{\boldsymbol{l},\boldsymbol{p},\theta}, f_{\boldsymbol{a},d}\}, \tag{10}$$

where $d_c(\cdot,\cdot)$ denotes the chamfer distance.

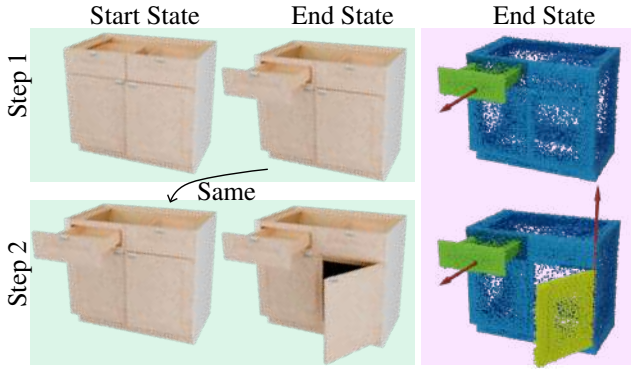

Figure 4. An example of modeling a multi-part articulated object. Taking the target object in multiple states, we update the Gaussians sequentially to reconstruct the final object with multiple sets of movable parts.

3.5. Final Joint Optimization

With the good initialization of 3D Gaussians, movable mask, and the motion parameters, now we are able to perform a joint optimization to find optimal solution.

Given that we already obtained quite accurate motion parameters in the previous step, we employ a lower threshold $\tau = 0.1$ to reclassify the Gaussians to get a more strict movable mask for the joint optimization. With this movable mask, we divide the Gaussians into movable part $\mathcal{G}_m$ and unmovable part $\mathcal{G}_u$. We apply the motion parameters obtained earlier to the movable part to reach its corresponding end state, while for the unmovable part, the start and end states are the same. Note that during the densification and pruning processes of the Gaussians, inaccuracies in the classification within two masks can be treated as noise within the Gaussians. This noise will be automatically updated alongside the Gaussians' densification and pruning processes, thereby achieving an accurate movable mask. We employ multi-perspective images from both states to supervise the rendering results before and after the movable part's movement, with the expectation that the loss values for each state will eventually converge to the same magnitude.

3.6. Multi-Part Object Modeling

To extend our method to reconstruct objects with multiple moving parts, we dissect the entire procedure into multiple sub-processes of single-object modeling and address each in turn, as shown in Fig. 4.

			Simulation										Real	
			Foldchair	Fridge	Laptop*	Oven*	Scissor	Stapler	USB	Washer	Blade	Storage*	Fridge	Storage*
Motion	Ang Err	Ditto	89.35	89.30	3.12	0.96	4.50	89.86	89.77	89.51	79.54	6.32	1.71	**5.88**
		PARIS	7.90	9.19	**0.02**	**0.04**	3.92	0.73	**0.13**	25.18	15.18	**0.03**	**1.64**	43.13
		Vanilla	87.47	14.08	11.65	44.95	89.80	84.85	75.50	84.69	88.76	83.17	82.99	78.35
		Ours	**0.02**	**0.04**	0.07	**0.04**	**0.20**	**0.03**	0.30	**0.06**	**1.99**	0.05	5.41	41.52
	Pos Err	Ditto	3.770	1.020	0.010	0.130	5.700	0.200	5.410	0.660	-	-	1.840	-
		PARIS	0.374	0.298	0.010	**0.003**	2.154	2.258	2.367	1.502	-	-	0.340	-
		Vanilla	0.036	0.503	0.265	0.055	0.027	0.127	0.013	0.030	-	-	0.084	-
		Ours	**0.003**	**0.001**	**0.003**	0.007	**0.002**	**0.000**	**0.000**	**0.004**	-	-	**0.057**	-
	Geo Dist	Ditto	99.36	F	5.18	2.09	19.28	56.61	80.60	55.72	F	**0.09**	8.43	**0.38**
		PARIS	131.82	24.64	**0.03**	**0.04**	120.61	110.71	64.91	60.62	0.54	0.14	**2.16**	0.56
		Vanilla	80.00	59.49	37.92	29.97	40.00	40.05	44.77	30.00	48.41	32.55	35.45	48.25
		Ours	**0.05**	**0.18**	0.55	0.61	**0.15**	**0.06**	**0.33**	**0.34**	**0.09**	0.30	9.38	0.41
Geometry	CD-s	Ditto	33.79	3.05	**0.25**	2.52	39.07	41.64	2.64	10.32	46.90	9.18	47.01	**16.09**
		PARIS	9.12	3.73	0.45	12.85	1.83	1.96	2.58	25.19	1.33	12.80	42.57	54.54
		Vanilla	308.39	8.81	389.67	F	F	100.23	88.12	312.80	F	218.01	F	F
		Ours	**0.33**	**0.90**	2.96	**2.06**	**0.34**	**1.68**	**1.02**	**6.17**	**0.69**	**2.96**	**37.01**	50.12
	CD-d	Ditto	141.11	0.99	**0.19**	0.94	20.68	31.21	15.88	12.89	195.93	2.20	50.60	**20.35**
		PARIS	8.79	7.76	0.49	28.51	46.69	19.36	5.53	178.39	25.29	76.75	45.66	864.82
		Vanilla	189.26	254.50	159.95	717.63	90.31	116.08	88.12	512.02	102.91	854.10	923.22	965.15
		Ours	**0.32**	**0.59**	6.23	**0.80**	**0.41**	**1.14**	**0.90**	**0.17**	**4.75**	**0.71**	**43.00**	730.45
	CD-w	Ditto	6.80	2.16	**0.31**	2.51	1.70	2.38	2.09	7.29	42.04	3.91	**6.50**	**14.08**
		PARIS	1.90	2.53	0.50	1.94	10.20	3.60	2.31	24.71	0.44	8.34	22.98	63.35
		Vanilla	5.91	9.99	4.73	2.80	7.63	33.34	8.74	8.37	5.52	3.21	30.44	78.65
		Ours	**0.36**	**0.85**	0.87	**1.91**	**0.33**	**1.42**	**0.93**	**5.64**	**0.34**	**2.09**	13.25	55.57

Table 1. The results on the PARIS dataset, encompassing both synthetic and real data, are presented. Objects marked with a '*' are the categories that Ditto [10] has been trained to recognize. Occasionally, Ditto may provide incorrect motion type predictions, which are indicated with an 'F' for joint state and a '*' for joint axis or position. There is a chance that Vanilla may fail to successfully segment the object, classifying the entire object as part of the movable category, which we also mark with an 'F'.

Throughout the reconstruction process, we consistently maintain the same set of Gaussians. We align the start state of the subsequent sub-process with one of the states from the preceding sub-process, ensuring that only an additional moving component is introduced in the new sub-process. Before entering the next sub-process, we configure the movable components within the existing Gaussian to the starting state of the subsequent sub-process using the motion parameters derived from the previous one. Within each sub-process, we employ a new identifier to label the movable mask while concurrently optimizing the unmovable Gaussians to update the new movable mask. Given that our movable mask is capable of being simultaneously densified and pruned with the Gaussian, the Gaussians marked by the movable mask from the previous step can also be optimized in the new sub-process. Furthermore, there is an advantage in that for the new sub-process, we can bypass the step of pre-training the initial Gaussian and proceed directly to the deformation network.

4. Results and Evaluation

4.1. Experimental Setup

Baselines. We compare our method to most related SOTA works Ditto [10] and PARIS [21]. Note that we follow PARIS's protocol and report the results derived from Ditto's released pre-trained model, which was trained on four object categories from the Shape2Motion dataset [37]. Moreover, we have also constructed a trivial reconstruction method based on our framework, denoted as *Vanilla*. Rather than employing our multi-step optimization process, we simultaneously optimize all parameters. We directly initialize two sets of Gaussians, one representing the static part and the other the mobile part. To simplify the problem, we also assume that the type of motion is known.

Evaluation metrics. To evaluate the efficacy of our methodology, we have designed a comparative experimental framework that spans three critical aspects: reasoning of motion, precision of geometry, and visual quality of novel views, as in previous works [21, 39]. The details of all the metrics are provided in the supplementary material. Moreover, we compare our approach's computational efficiency and memory utilization against PARIS, which is also a self-supervised method like ours.

4.2. Comparison to Baselines

Table 1 and Table 2 show results on the PARIS object dataset, including synthetic and real instances, summarized over 10 trials. Ditto's performance is contingent upon its exposure to object forms and structural templates acquired from the training dataset, which includes categories such as laptops, ovens, and storage units. The algorithm demonstrates proficiency in reconstructing shapes for ob-

Metrics	Methods	Foldchair	Fridge	Laptop	Oven	Scissor	Stapler	USB	Washer	Blade	Storage
PSNR-s (↑)	PARIS	37.95	36.45	36.59	35.59	37.64	33.84	37.58	37.22	38.28	35.02
	Vanilla	31.49	23.21	22.45	24.16	21.08	22.26	21.99	26.65	23.87	29.65
	Ours	**41.68**	**36.89**	**37.08**	**36.60**	**40.99**	**43.36**	**41.09**	**40.63**	**41.31**	**38.65**
PSNR-e (↑)	PARIS	38.45	33.68	34.24	34.33	38.15	40.19	36.13	37.75	37.13	36.41
	Vanilla	31.42	25.44	22.56	23.38	21.07	22.75	23.11	26.09	27.76	29.65
	Ours	**42.83**	**37.63**	**35.37**	**36.72**	**41.13**	**42.84**	**42.27**	**39.98**	**41.36**	**38.08**
SSIM-s (↑)	PARIS	0.956	0.968	0.987	0.966	0.953	0.957	0.979	0.970	0.988	0.961
	Vanilla	0.949	0.938	0.936	0.954	0.961	0.964	0.926	0.979	0.940	0.952
	Ours	**0.980**	**0.987**	**0.988**	**0.984**	**0.995**	**0.996**	**0.993**	**0.993**	**0.996**	**0.976**
SSIM-e (↑)	PARIS	0.955	0.965	0.984	0.970	0.954	0.956	0.980	0.972	0.987	0.965
	Vanilla	0.943	0.948	0.924	0.945	0.933	0.967	0.931	0.980	0.947	0.950
	Ours	**0.983**	**0.989**	**0.986**	**0.984**	**0.995**	**0.995**	**0.994**	**0.992**	**0.996**	**0.975**
LPIPS-s (↓)	PARIS	0.062	0.038	0.047	0.070	0.067	0.064	0.070	0.020	0.008	0.084
	Vanilla	0.084	0.083	0.083	0.081	0.083	0.081	0.087	0.084	0.081	0.114
	Ours	**0.056**	**0.039**	**0.036**	**0.040**	**0.012**	**0.008**	**0.019**	**0.021**	**0.008**	**0.081**
LPIPS-e (↓)	PARIS	0.061	0.039	0.047	0.044	0.016	0.021	0.027	0.020	0.009	0.083
	Vanilla	0.094	0.073	0.090	0.073	0.067	0.051	0.068	0.037	0.085	0.124
	Ours	**0.059**	**0.033**	**0.031**	**0.039**	**0.012**	**0.009**	**0.017**	**0.023**	**0.007**	**0.087**

Table 2. Comparison of visual quality on the PARIS dataset, where (-s) and (-e) denote the *start* and *end* states.

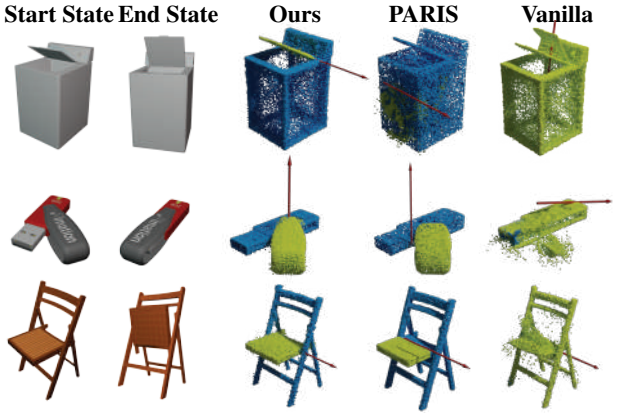

Figure 5. Qualitative results of part segmentation and motion estimation for some of the synthetic objects derived from the PARIS dataset.

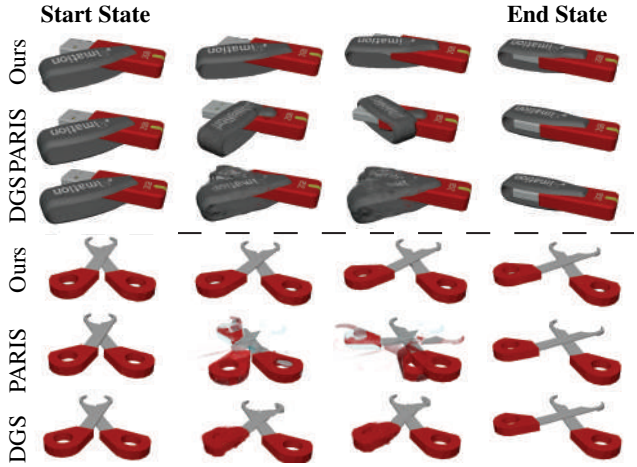

Figure 6. Illustration of unseen states inference.

jects that fall within its training purview. PARIS, on the other hand, experiences significant fluctuations in its performance across various trials. Although it occasionally achieves commendable shape reconstruction, it also suffers from occasional catastrophic failures. These inconsistencies result in a markedly inferior overall performance in both the reconstruction of shape and the delineation of object articulation. Our approach demonstrates resilience to varying initial conditions and consistently delivers precise reconstructions of both shape and articulation throughout multiple iterations.

Figure 5 shows some representative results of various methods in part-segmentation and motion estimation. We have visualized each of our Gaussian movable masks in the form of point clouds, which allows for a more intuitive observation of the accuracy of our method in segmentation and estimation. Figure 6 illustrates the predicted articulation sequence based on the reconstructions obtained by different methods. PARIS struggles to accurately estimate motion parameters for certain objects. Even when the segmentation results are close to correct, the motion process still deviates from the actual outcome. We also added the Deformable GS (DGS) [46], a recent and pioneering 4D Gaussian work, for visual comparison. Due to the sparsity of the input temporal states, DGS can only reconstruct the appearance of the two input states with relative accuracy. It lacks prior knowledge of articulated motion and thus cannot infer the actual transformation process between the two states. We also created our own datasets for comparative analysis. As shown in Figure 7, our method demonstrates superior results across a greater variety of objects compared to previous works, and more results can be found in the supplementary material.

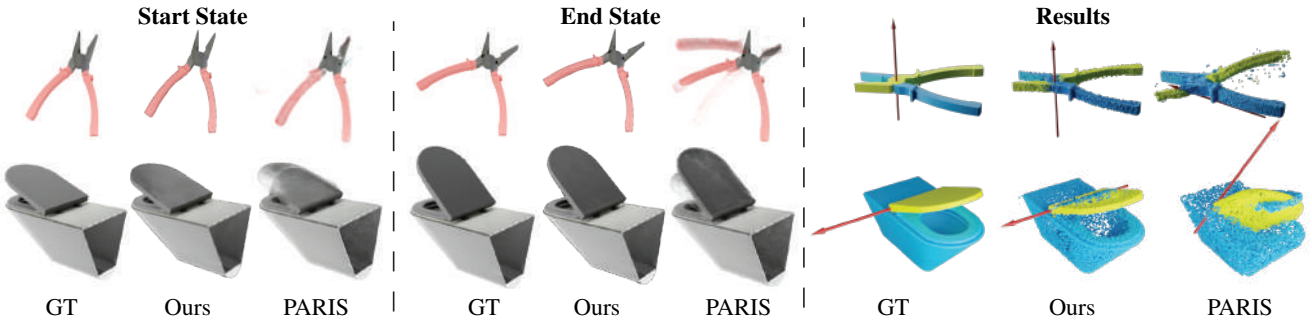

Figure 7. Illustration of the effects of applying our method and the PARIS algorithm to our dataset.

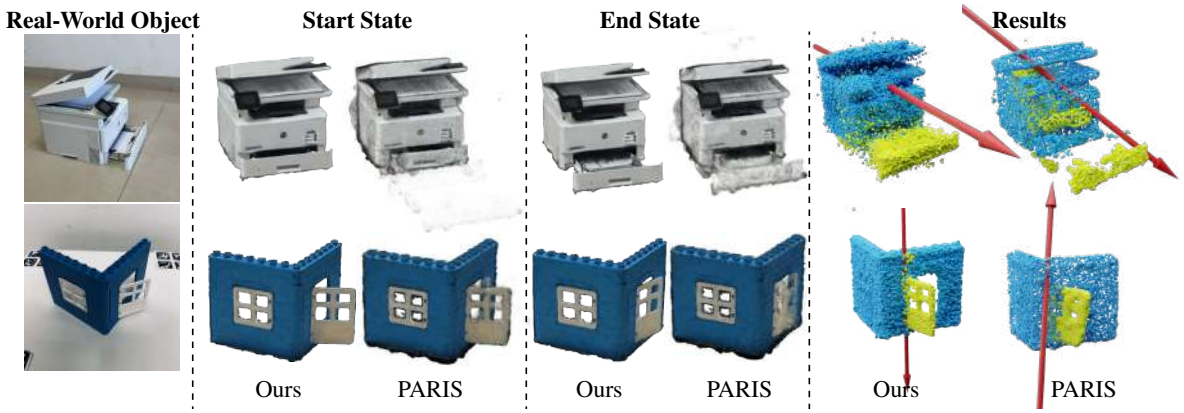

Figure 8. Illustrations of our method applied to the reconstruction results of real-world objects.

Note that our method on real objects has not achieved consistently the best performance for all metrics in Table 1. That is because, in the PARIS dataset, the real-world object masks have various imperfections at the edges, such as missing parts of the object or mistakenly including portions of the background as part of the object. Additionally, obtaining the ground truth for motion and segmentation of real-world objects is also quite peculiar, as manual annotation of motion and segmentation can easily introduce errors in the ground truth. Consequently, using it is not entirely effective for quantitative evaluation. To demonstrate the effectiveness of our method in reconstructing real objects, we scanned multiple objects and applied our method for digital-twin modeling, as shown in Figure 1 and Figure 8.

For additional visual results, please refer to the supplementary materials.

4.3. Ablation Studies

The "Vanilla" baseline in Table 1, Table 2 and Figure 5 refers to the framework of our method without the progressive optimization process. The results demonstrate that the visual similarity between the initial and final states is preserved even after the removal of our progressive optimization. Nonetheless, part segmentation tends to optimize directly to a scenario where all Gaussians are consolidated into a single category. Consequently, the motion parameters that are learned result in a motion that is nearly static.

5. Conclusion

In this work, we introduce ArticulatedGS, a self-supervised framework designed to build digital twins of articulated objects using 3D Gaussian Splatting. Utilizing two distinct sets of multi-view images, each depicting an object in separate static articulation configurations, we reconstruct the articulated object in 3D Gaussian representations, integrating both appearance and mobility information simultaneously. Our method adeptly disentangles a multitude of highly interrelated parameters through a sophisticated multi-step optimization process. This approach not only ensures a stable and reliable optimization procedure but also delivers high-fidelity results. These outcomes are crucial for the precise depiction of articulated objects within digital twins, ensuring that they are accurately and realistically represented. Our approach also has certain limitations, such as the requirement for world coordinate alignment between two states, and corresponding areas under two different states should have similar lighting effects. More detailed analysis can be found in the supplementary material.

Acknowledgement

This work was supported in part by NSFC (62322207, 62025207, 62325211, 62132021), Shenzhen Science and Technology Program (RCYX20210609103121030), and the Major Program of Xiangjiang Laboratory (23XJ01009). We would like to thank Youcheng Cai and Changhao Li from University of Science and Technology of China, for their advice on the methodology. We also thank the anonymous reviewers for their valuable comments and suggestions.

References

[1] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems*, 35:5982–5994, 2022. 1

[2] Helisa Dhamo, Yinyu Nie, Arthur Moreau, Jifei Song, Richard Shaw, Yiren Zhou, and Eduardo Pérez-Pellitero. Headgas: Real-time animatable head avatars via 3d gaussian splatting. *arXiv preprint arXiv:2312.02902*, 2023. 2, 3

[3] Yuanxing Duan, Fangyin Wei, Qiyu Dai, Yuhang He, Wenzheng Chen, and Baoquan Chen. 4d gaussian splatting: Towards efficient novel view synthesis for dynamic scenes. *arXiv preprint arXiv:2402.03307*, 2024. 2

[4] Bardienus P Duisterhof, Zhao Mandi, Yunchao Yao, Jia-Wei Liu, Mike Zheng Shou, Shuran Song, and Jeffrey Ichnowski. Md-splatting: Learning metric deformation from 4d gaussians in highly deformable scenes. *arXiv preprint arXiv:2312.00583*, 2023. 2

[5] Lin Gao, Jie Yang, Bo-Tao Zhang, Jia-Mu Sun, Yu-Jie Yuan, Hongbo Fu, and Yu-Kun Lai. Mesh-based gaussian splatting for real-time large-scale deformation. *arXiv preprint arXiv:2402.04796*, 2024. 3

[6] Quankai Gao, Qiangeng Xu, Zhe Cao, Ben Mildenhall, Wenchao Ma, Le Chen, Danhang Tang, and Ulrich Neumann. Gaussianflow: Splatting gaussian dynamics for 4d content creation. *arXiv preprint arXiv:2403.12365*, 2024. 2

[7] Haoran Geng, Helin Xu, Chengyang Zhao, Chao Xu, Li Yi, Siyuan Huang, and He Wang. Gapartnet: Cross-category domain-generalizable object perception and manipulation via generalizable and actionable parts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7081–7091, 2023. 2

[8] Nick Heppert, Muhammad Zubair Irshad, Sergey Zakharov, Katherine Liu, Rares Andrei Ambrus, Jeannette Bohg, Abhinav Valada, and Thomas Kollar. Carto: Category and joint agnostic reconstruction of articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21201–21210, 2023. 2

[9] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. *arXiv preprint arXiv:2312.14937*, 2023. 4

[10] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5616–5626, 2022. 1, 2, 3, 6

[11] Zheheng Jiang, Hossein Rahmani, Sue Black, and Bryan M Williams. 3d points splatting for real-time dynamic hand reconstruction. *arXiv preprint arXiv:2312.13770*, 2023. 2, 3

[12] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 2, 4

[13] Muhammed Kocabas, Jen-Hao Rick Chang, James Gabriel, Oncel Tuzel, and Anurag Ranjan. HUGS: Human gaussian splatting. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[14] Agelos Kratimenos, Jiahui Lei, and Kostas Daniilidis. Dynmf: Neural motion factorization for real-time dynamic view synthesis with 3d gaussian splatting. *arXiv preprint arXiv:2312.00112*, 2023. 2

[15] Long Le, Jason Xie, William Liang, Hung-Ju Wang, Yue Yang, Yecheng Jason Ma, Kyle Vedder, Arjun Krishna, Dinesh Jayaraman, and Eric Eaton. Articulate-anything: Automatic modeling of articulated objects via a vision-language foundation model. *arXiv preprint arXiv:2410.13882*, 2024. 2

[16] Jiahui Lei, Congyue Deng, Bokui Shen, Leonidas Guibas, and Kostas Daniilidis. Nap: Neural 3d articulation prior. *arXiv preprint arXiv:2305.16315*, 2023. 1

[17] Jiahui Lei, Yufu Wang, Georgios Pavlakos, Lingjie Liu, and Kostas Daniilidis. Gart: Gaussian articulated template models. *arXiv preprint arXiv:2311.16099*, 2023. 2, 3

[18] Xiaolong Li, He Wang, Li Yi, Leonidas J Guibas, A Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3706–3715, 2020. 3

[19] Zhe Li, Zerong Zheng, Lizhen Wang, and Yebin Liu. Animatable gaussians: Learning pose-dependent gaussian maps for high-fidelity human avatar modeling. *arXiv preprint arXiv:2311.16096*, 2023. 1

[20] Youtian Lin, Zuozhuo Dai, Siyu Zhu, and Yao Yao. Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle. *arXiv preprint arXiv:2312.03431*, 2023. 2

[21] Jiayi Liu, Ali Mahdavi-Amiri, and Manolis Savva. Paris: Part-level reconstruction and motion analysis for articulated objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 352–363, 2023. 1, 2, 3, 6

[22] Jiayi Liu, Manolis Savva, and Ali Mahdavi-Amiri. Survey on modeling of articulated objects. *arXiv preprint arXiv:2403.14937*, 2024. 1, 2

[23] Yu Liu, Baoxiong Jia, Ruijie Lu, Junfeng Ni, Song-Chun Zhu, and Siyuan Huang. Building interactable replicas of complex articulated objects via gaussian splatting. In *The Thirteenth International Conference on Learning Representations*, 2025. 2

[24] Haozhe Lou, Yurong Liu, Yike Pan, Yiran Geng, Jianteng Chen, Wenlong Ma, Chenglong Li, Lin Wang, Hengzhen Feng, Lu Shi, et al. Robo-gs: A physics consistent spatial-temporal model for robotic arm with hybrid representation. *arXiv preprint arXiv:2408.14873*, 2024. 1

[25] Guanxing Lu, Shiyi Zhang, Ziwei Wang, Changliu Liu, Jiwen Lu, and Yansong Tang. Manigaussian: Dynamic gaussian splatting for multi-task robotic manipulation. *arXiv preprint arXiv:2403.08321*, 2024. 2

[26] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023. 2, 4

[27] Kaichun Mo, Leonidas J Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6813–6823, 2021. 1

[28] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13001–13011, 2021. 2

[29] Yuri Pekelny and Craig Gotsman. Articulated object reconstruction and markerless motion capture from depth video. In *Computer Graphics Forum*, pages 399–408. Wiley Online Library, 2008. 2

[30] Chandradeep Pokhariya and N Ishaan. Manus: Markerless hand-object grasp capture using articulated 3d gaussians. In *IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR)*. CVPR 2024, 2024. 3

[31] Shengyi Qian and David F Fouhey. Understanding 3d object interaction from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21753–21763, 2023. 1

[32] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. *arXiv preprint arXiv:2312.02069*, 2023. 2

[33] Andrei Sharf, Hui Huang, Cheng Liang, Jiapei Zhang, Baoquan Chen, and Minglun Gong. Mobility-trees for indoor scenes manipulation. In *Computer Graphics Forum*, pages 2–14. Wiley Online Library, 2014. 1

[34] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749, 2020. 1

[35] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in neural information processing systems*, 34:251–266, 2021. 1

[36] Wei-Cheng Tseng, Hung-Ju Liao, Lin Yen-Chen, and Min Sun. Cla-nerf: Category-level articulated neural radiance field. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8454–8460. IEEE, 2022. 2

[37] Xiaogang Wang, Bin Zhou, Yahao Shi, Xiaowu Chen, Qinping Zhao, and Kai Xu. Shape2motion: Joint analysis of motion parts and attributes from 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8876–8884, 2019. 6

[38] Fangyin Wei, Rohan Chabra, Lingni Ma, Christoph Lassner, Michael Zollhöfer, Szymon Rusinkiewicz, Chris Sweeney, Richard Newcombe, and Mira Slavcheva. Self-supervised neural articulated shape and appearance models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15816–15826, 2022. 2

[39] Yijia Weng, Bowen Wen, Jonathan Tremblay, Valts Blukis, Dieter Fox, Leonidas Guibas, and Stan Birchfield. Neural implicit representation for building digital twins of unknown articulated objects. *arXiv preprint arXiv:2404.01440*, 2024. 2, 6

[40] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023. 2

[41] 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

[42] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. *arXiv preprint arXiv:2311.12198*, 2023. 2

[43] Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. Street gaussians for modeling dynamic urban scenes. *arXiv preprint arXiv:2401.01339*, 2024. 2

[44] Gengshan Yang, Minh Vo, Natalia Neverova, Deva Ramanan, Andrea Vedaldi, and Hanbyul Joo. Banmo: Building animatable 3d neural models from many casual videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2863–2873, 2022. 1

[45] Ji Yang, Xinxin Zuo, Sen Wang, Zhenbo Yu, Xingyu Li, Bingbing Ni, Minglun Gong, and Li Cheng. Object wake-up: 3d object rigging from a single image. In *European Conference on Computer Vision*, pages 311–327. Springer, 2022. 1

[46] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023. 2, 7

[47] Jiajun Zhang, Yuxiang Zhang, Liang An, Mengcheng Li, Hongwen Zhang, Zonghai Hu, and Yebin Liu. Manidext: Hand-object manipulation synthesis via continuous correspondence embeddings and residual-guided diffusion. *arXiv preprint arXiv:2409.09300*, 2024. 1