FOR APPLICATION PURPOSES
ONLY – YET TO BE ARXIVED

SPLATART24: Articulated Gaussian Splatting with Estimated Object Structure

Stanley Lewis[1], Tom Gao[1], Vishal Chandra[1] and Odest Chadwicke Jenkins[1]

***Abstract*— It is a common problem for robots on how to best represent articulated objects. Articulated objects such as pliers, clamps, or cabinets require representations that are capable of capturing not only the observed geometry and color information, but also part connectivity, joint paramaterization, and overall kinematic structure. In this work, we present SPLATART24 - a method for determining a common representation for tree structured articulated objects conditioned on segmented image and camera pose inputs based on Gaussian Splats [1]. SPLATART24 takes a parts-based approach in which the observed articulated object is first decomposed into constituent rigid body objects, which are then registered against each other across different configuration observations. These pose estimates are then used to estimate the connectivity and joint parameters between each part. Finally, this inferred structure can be used to render the object in novel poses and configurations. In this work, we present data on the SPLATART24 pipeline as applied to the synthetic Paris dataset objects, as well as qualitative results on real world objects to demonstrate the capabilities of the parts-based approach.**

I. Introduction

Articulated objects pose significant challenges for robots due to their complex degrees of freedom compared to rigid-body objects, complicating tasks like pose estimation and grasp synthesis. The relative scarcity of suitable datasets exacerbates these challenges. Tools such as LabelFusion [2] and ProgressLabeller [3] dramatically improve the speed at which researchers can create segmentation or pose estimation datasets (including difficult categories such as transparent objects), but articulated objects remain difficult to scale in terms of data collection. Each additional degree of freedom complicates the representation, and increases the number of observations needed to achieve robust generalization for many tasks. Data driven robot learning approaches then, would benefit from the ability to generate large numbers of synthetic articulated object observations from a comparatively small number of real-world collected data. Towards that end, representations such as Neural Radiance Fields (NeRFs) [4] and Gaussian Splats [1] have seen success as implicit and explict representations respectively for articulated objects in robotic applications. For example, NARF-22 [5] showed success in applying articulated NeRFs to a real-world tool pose and configuration estimation task, and GraspSplats [6] demonstrated success in using gaussian splatting for grasp synthesis. This work contributes an initial approach to utilizing Gaussian Splatting to the articulated object structural estimation and novel view-configuration rendering task.

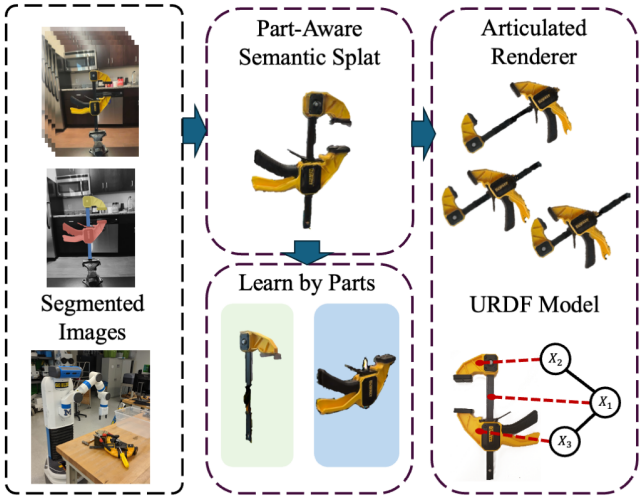

Fig. 1: SPLATART424 is a pipeline which takes in part-segmented images of an articulated object at a small number of configurations, then utilizes cross-scene gaussian splat observations to estimate part poses and joint parameters. These create a URDF-style model and a subsequent articulation enabled splat for configurable rendering.

In this work, we present SPLATART24, a parts based approach to gaussian splatting that learns a configuration conditioned splat-based renderer for tree-structured articulated objects from a small (on the order of 100) set of segmented images for very small - as few as two - distinct configuration examples. SPLATART24 utilizes parts-based segmentations to register each component part of the articulated object across each configuration example, and then utilizes a graph-search based approach to estimate the parameters for single-degree-of-freedom revolute or prismatic joints. These joint estimations can then be used to generate an articulation structure in URDF (Unified Robotics Description Format) format, or can be used directly for rendering the gaussian splat at previously unobserved poses and configuration. We show rendering and structure estimations results on the SAPIEN [7] objects within the PARIS dataset [8]. Finally, we show qualitative rendering results on real world data with a minimal number of training samples to demonstrate the capabilities of the SPLATART24 approach.

[1]S. Lewis, Tom Gao, Vishal Chandra, and O.C. Jenkins are with the Robotics Department, University of Michigan, Ann Arbor, MI 48109 {stanlew, zimingg, chandrav, ocj}@umich.edu
This work is supported in part by Ford Motor Company, in part by J.P. Morgan AI Research, and in part by Amazon

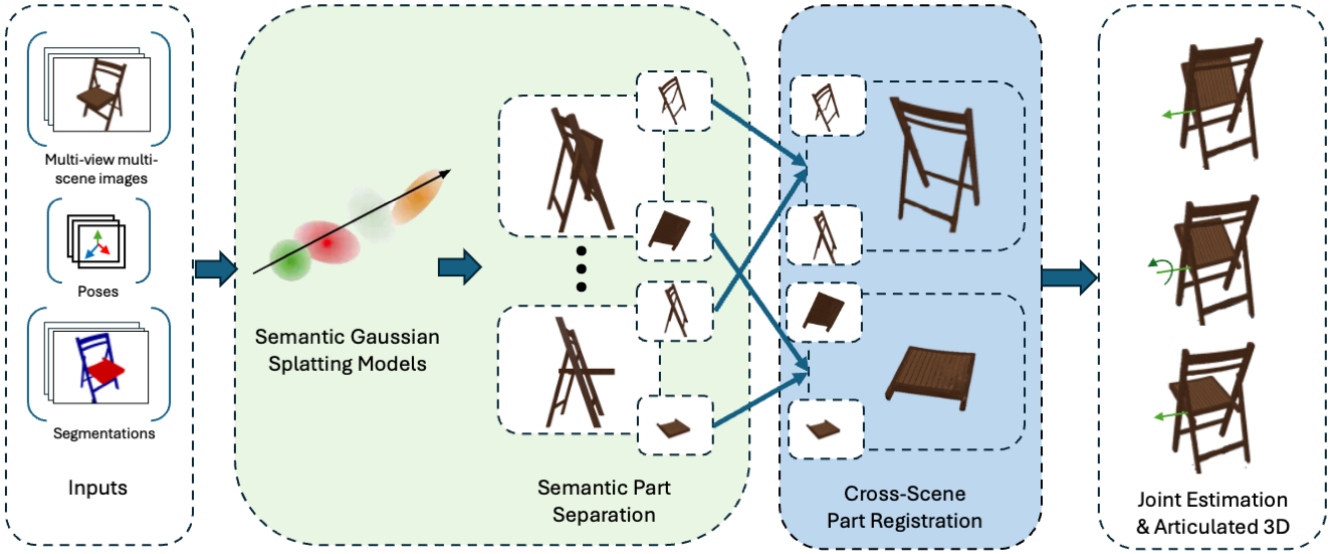

Fig. 2: An Overview of the SPLATART24 Pipeline

II. Related Work

A. Parts-Based Approaches

Articulated objects remain a difficult class of objects for robots to work with. Explicit methods using a parts-based approach combining 3d mesh models and URDF (Unified Robotics Description Format) files have seen success such as Pavlasek et al. which leveraged a belief propagation method combined with a a learned-likelihood particle filter to perform pose and configuration estimation on cluttered tool scenes [9]. However, creating these models is laborious and difficult. Neural Radiance Fields (NeRF) [4] have also shown success in breaking the reliance on mesh models such as in NARF22 [5] by learing per-part renderers, which are them subsequently composed to produce articulated object renderings. While works such as NARF22 have continued the parts-based approach to produce NeRF representations of articulated objects, they still require a-priori knowledge of the articulated object's structure . SPLATART24 in contrast leverages a parts-based separation of an object-level gaussian splat. We utilize a shared-scene splat representation for part extraction and localization from segmentation masks. This enables joint parameter estimation methods to produce a URDF-style model of the object's structure via gradient descent on the joint parameters. If more than two configuration examples are given, more traditional geometric joint inference methods such as in Sturm et al. [10] can also be leveraged, as demonstrated in NARF24 [11].

B. Data Driven Approaches

Other works have utilized data-driven, deeply learned approaches to infer object articulations. URDFormer [12] utilized simulation assets combined with known URDF models to learn an image-to-URDF transformer model. Weng el al. additionally inferred joint locations and angles from posed RGB input images via part segmentations [13]. Several works have also perceived articulations from video such as GART [14] which utilized template models to infer articulations even for deformable articulated objects without clear part separations (e.g. a dog or human body). PARIS [8] utilized inference against a learned deformation field to represent articulated objects, but is generally applied to objects with low degrees of freedom.

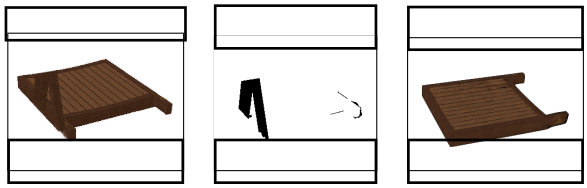

Fig. 3: An input image, training mask, and final part splat generated from the training process. Of note, the mask removes the area of self-occlusion caused by the chair frame that is observable in the input training image. This approach allows for better object-centric training across scenes, which is especially useful for prismatic join objects which may have large amounts of self-occlusions

III. Method

A. Overview

The SPLATART24 pipeline is generally comprised of five primary steps:

1) a per-scene semantic gaussian splat training step
2) a cross-scene part registration step
3) a joint parameter estimation step
4) a canonical part training step
5) a novel pose/configuration rendering step

. A overview of this pipeline is shown in figure **??**, and each component step is discussed more in detail below.

B. Semantic Gaussian Splats and Parts Separation

Given an arbitrary camera pose, X, Gaussian Splats render an RGB image of a known scene by rasterizing collection of gaussian distributions $(\mu_i, \Sigma_i, \alpha_i, sh_i)$ where μ, Σ, α, and sh represent the means, covariances, opacities and spherical harmonic coefficients of each distribution, and i is an index with $i \in [0, I]$. In practice, splats typically do not store and optimize Σ_i, but instead store a colletion of scale variables s_i and quaternions q_i which are used to compute the covariance $\Sigma_i = R_i s_i s_i^T R_i^T$ where R_i is the rotation matrix corresponding to quaternion q_i. SPLATART24 extends this representation to render semantic information by including $\mathcal{P}$, where each $\mathcal{P}_i \in [0, I]$ is a vector of length N where N is the number of part classes. Each index $n \in [0, N]$ of $\mathcal{P}_i$ then corresponds to the likelihood score of a splat belonging of part n.

SPLATART24 uses an extension of the Splatfacto model as provided in NerfStudio [15] to train K splats $(\mu_{i,i}, _i, \alpha_i, sh_i)_k$, where k is an indexing variable into each scene. The Splatfacto model is convenient as it represents a typical collection of Gaussian Splat training improvements such as scale regularization as proposed in PhysGaussians [16]. A scene corresponds to a single observed configuration and background environment, and contains a collection of images $\mathcal{I}_j$, poses $\mathcal{X}_j$, and segmentation images $\mathcal{S}_j$, where $j \in [0, J]$ is the indexing variable into the collection of each scene's training data. The Splat's parameters are trained as normally specified within Nerfstudio (a combination ℓ_1 and Structural Similarity loss), but with the addition of a cross entropy loss between each $\mathcal{S}_j$ and the rasterized $\hat{\mathcal{S}}_j$.

Subsequent to training, the each splat collection is then further subdivided into parts. Each splat member is assigned to part $n = argmax(\hat{\mathcal{S}})$ based on the segmentation prediction of the gaussian. Each part is also augmented with an additional q_n and $trans_n$ quaterion and translation variable, which are used to transform the μ_i and Σ_i's during rendering, thus enabling each part to be transformed independently of the rest of the splat.

After each scene's splat collection is divided into parts, SPLATART24 seeks to find values for each q_n and $trans_n$ that result in high levels of overlap with the canonical part at index $n = 0$. This is accomplished via an iNeRF style of pose estimation, in which the canonical part is rendered with q_0, $trans_n$ at identity, and the other q_n and $trans_n$ are optimized via stochastic gradient descent(SGD). Both parts are rendered using the set of camera poses $\mathcal{X}_j, 0$ from the canonical part's training process, and then a traditional image space ℓ_2 sum-of-squares loss is used in combination with an Adam optimizer over 100 epochs to determine the best transform between each part in each scene. An example of the intitial and final results of this process can be seen in figure 4. Of note, we do speculate that the simple image space loss metric used here will not be sufficient for many objects. Additional losses such as a chamfer loss or similar will likely improve this step, but we leave those improvements for future research.

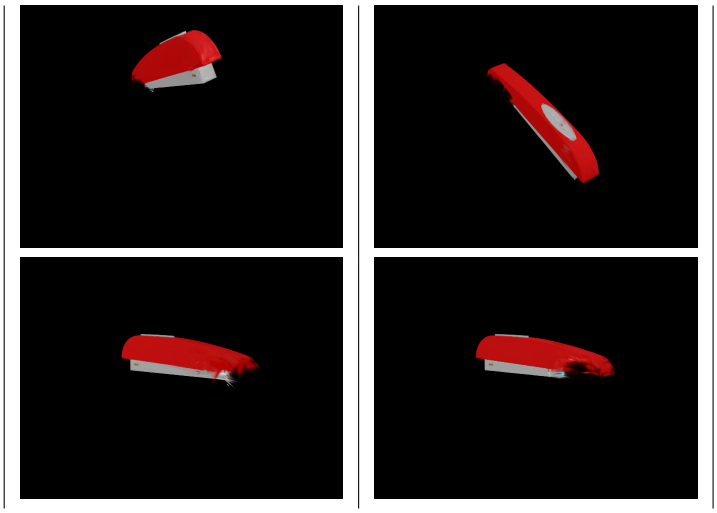

Fig. 4: Pre- and post-registration results for the PARIS dataset stapler head. Top row: The initial states of the parts, where the left image is from the initial scene, and the right image is from the target scene. Bottom row: Subsequent to the part registration process, the part in the target scene is able to be rendered at the same pose as the base scene

C. Joint Estimation

Subsequent to Part Registration, SPLATART24 performs a joint parameter estimation step. For each joint combination of parts $n_x, n_y \in [0, N]$, the collection of transforms between the parts is computed from the q_n and $trans_n$'s determined during part registration. The list of registration based transforms between the parts is denoted as $\mathcal{X}_{base}$. Next, both a single-axis revolute and single-axis translation joint are estimated via SGD. Each joint is characterized by a 3 degree of freedom axis $\mathcal{A}$, a 6 degree of freedom pre-transformation $\mathcal{X}_{joint}$ and a scalar joint parameter γ. These parameters are trained via an average distance (ADD) loss as originally proposed by Hinterstoisser et al. [17]. Typically, the ADD loss is computed using a point cloud, mesh model, or other similar geometry representation. For SPLATART24, we use the μ_i values as a stand in for such a representation. We compute the loss as the average euclidean distance between corresponding μ's after being transformed by the registered $\mathcal{X}_{base}$ transforms, and after having been transformed by the current estimated joint's $\mathcal{X}_{joint}$ transform. For this optimization, the first entry in the γ vector is not optimized, and is forced to remain as a zero value. This eliminates a parameter from the optimization process, and enforces better solutions for the other joint parameter values. Subsequent to optimization, a joint is kept if its average ADD score is below some human provided threshold τ_{ADD}. For this work, we set τ_{ADD} to $1.5mm$. If both the translation and rotation join clear the threshold, the one with the lower value is retained. If neither joint estimate clear the threshold, then it is assumed that the two parts are not connected within the ground truth URDF.

Fig. 5: 8 novel scenes synthesized with our method, overlaid on the original training image. The clamp progressively opens from top left to bottom right.

D. Canonical Part Training

After part registration, there is an optional step to train so-called 'canonical' versions of each part in the articulated object. Because of self-occlusion and other difficulties, it is unlikely that any of the extracted part splats generated during the initial training stages are exceptional representations of the entire part. Thus, an additional step can be optionally taken to train individual splats for each part, using the poses determined during the Part Registration stage, and utilizing a training mask which zeros out all pixels outside of the observed geometry of the desired part. An example of an input image, training mask, and final part splat can be seen in figure 3.

E. PARIS Dataset

The PARIS Dataset is comprised of ten (10) different articulated objects curated from the SAPIEN dataset, in addition to a handful of real-world collected examples [8] [7]. For the purposes of this experiment, we focus only on the simulated SAPIEN objects, of which examples can be seen in figure 6. A total of 220 images are rendered for each of the objects, 100 train examples and 10 test examples for two different configuration values, as well as their corresponding segmentations. We report metrics for our part registration step as compared to the ground truth, as well as joint estimation metrics as compared to PARIS and DITTO benchmarks.

F. Final Configurable Rendering

In the final step, the final structure and rendering is accomplished. Starting from a designated root node part (for this work, it is always the part at semantic index zero), a breadth-first search is performed over the estimated joint's connectivity graph. This is to ensure that no kinematic loops are created in our joint representations - although such a structure is in theory detectable and solvable, we leave loopy objects for future research efforts. With the obtained set of joints, a configuration vector can then either be instantiated to match an example from the training set, or it can be provided by the user for some downstream robot task. During rendering, the joint parameter tree is once again traversed to generate out the list of $\mathcal{X}_{joint}$'s, which are subsequently turned into q_n and $trans_n$ values for use during each part's rasterization operation.

IV. Results

1) Part Registration: While visually the part registration appears to produce good results, we present metrics and loss values to illustrate the registration effectiveness in table III. In general, the image loss values are very low on a per-pixel basis. When comparing the average μ values between the ground truth and registered part however, there sometimes are larger deltas than expected. This appears to typically be caused by self-obscuration when looking at objects like the Blade or Storage, when a translational joint results in chunks of geometry not being represented in the part's splat parameters.

2) Joint Estimation: Table I shows the error of the predicted joint states for each of the objects in the PARIS dataset as compared to the ground truth. In general, SPLATART24 performs comparably to the baselines (DITTO in particular), but can suffer if the part registration step produces noisy or inaccurate results. This is best seen in the washer example, which has a slightly perturbed localization of the lid, resulting in the worst performance of all the examples of SPLATART24.

3) Renderings: Renderings of the inferred articulated objects are shown on figure 6. Each figure has a sweep for one of the parts from the minimum detected joint parameter to the maximum, in 9 steps.

A. Real World Data

Though we have shown the proposed method to be effective on synthetic datasets of substantial size, the computational and data efficiency of gaussian splatting inspires real world applications. We show the effectiveness of the method with very few human-collected segmentation images of a real object. Training data for this experiment was collected with a single iPhone 14 Pro's built-in LiDAR and the Record 3D app [18]. The segmentation was subsequently performed manually using Roboflow [19]. In total, 30 images were captured in each of 3 scenes of a trigger clamp (total 90 images) as shown in figure 7. From this, several novel scenes (i.e. novel clamp positions) were rendered. Qualitative results for novel rendering are shown in figure 5.

V. Conclusion

In conclusion, SPLATART24 demonstrates the capability to generate articulated object renderers via gaussian splatting without a prior knowledge of the object structure. It achieves results comprable to previous radiance field based efforts such as PARIS, and additionally is able to generate renderers from real world collected data with a

TABLE I: Quantitative results for joint state estimation

Metrics	Methods	Stapler	USB	Scissor	Fridge	Foldchair	Washer	Oven	Laptop	Blade	Storage
Joint Err (°,m)	Ditto	56.61	80.60	19.28	F	99.36	55.72	2.094	5.181	F	Storage
	PARIS	0.000	0.028	0.000	0.001	0.000	0.079	0.000	0.028	0.064	0.000
	SPLATART24	0.093	0.060	12.101	0.002	0.019	9.4	0.16	0.074	0.002	0.041

TABLE II: Quantitative results for the relative joint states. Revolute joint errors are in degrees, and prismatic joints (Blade, Storage) are in meters.

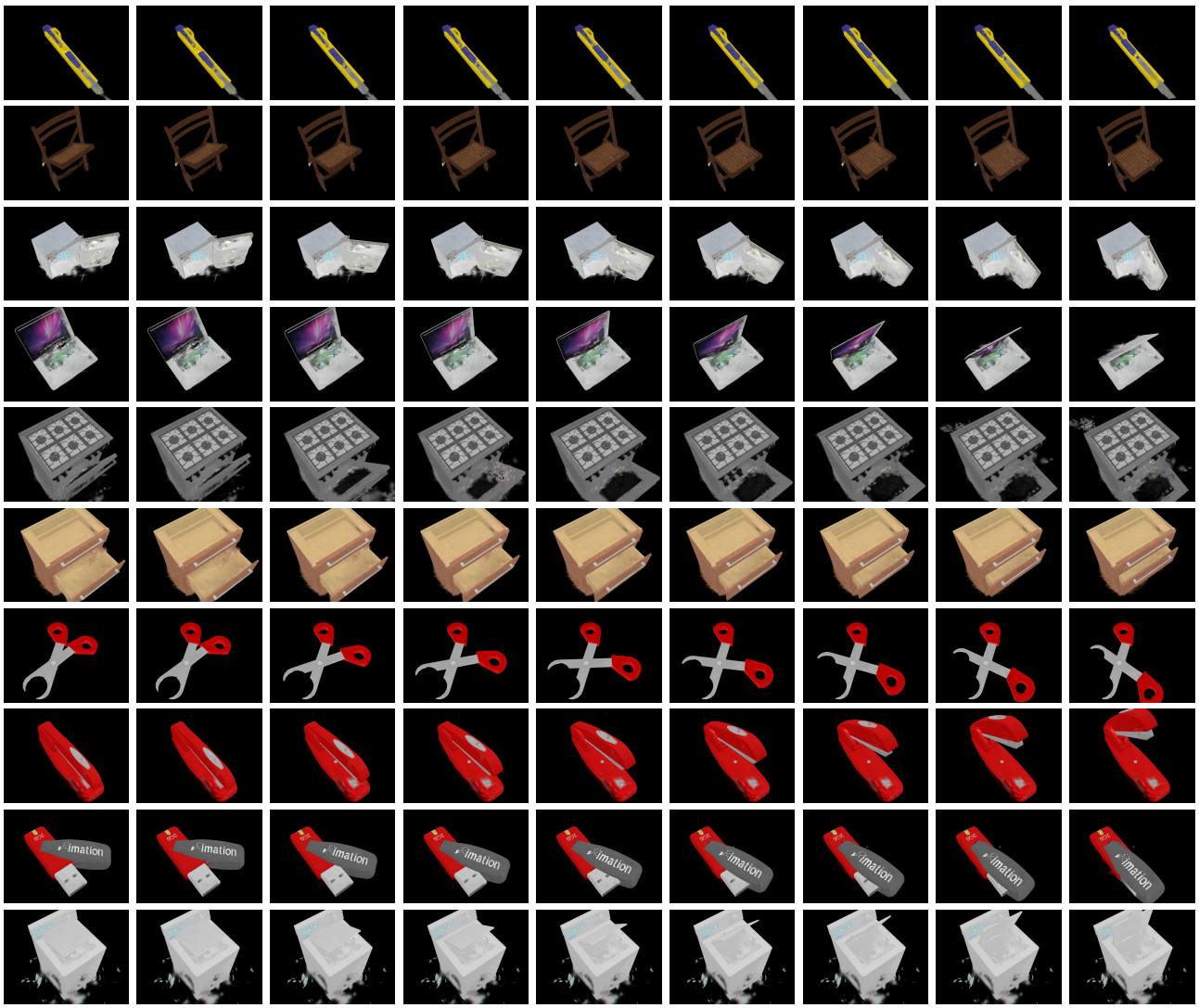

Fig. 6: Renderings of the articulated objects within the PARIS dataset. These renderings do not implement the canonical part training, but instead use the first scene as the canonical collection of parts. This is done to demonstrate the effects of self-obscuration on the parts, which is clearly visible on object areas such as the inside of the oven door.

minimal amount of human labelling required. The addition of segmentations is a drawback of this approach, however that is ameliorated by the rise of Segment Anything based labelling tools. We additionally do speculate that a run-time inference of part seperation is possible in the gaussial splat context, but we leave this effort for future study. Further information on this work can be found on its website at https://progress.eecs.umich.edu/projects/splatart/.

REFERENCES

[1] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering.," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.

[2] P. Marion, P. R. Florence, L. Manuelli, and R. Tedrake, "Label fusion: A pipeline for generating ground truth labels for real rgbd data of cluttered scenes," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3235–3242, IEEE, 2018.

[3] X. Chen, H. Zhang, Z. Yu, S. Lewis, and O. C. Jenkins, "Progress-labeller: Visual data stream annotation for training object-centric 3d

TABLE III: Performance metrics for the part registration stage

Object	Mean Img. L2 Loss (pixel intensity)	Delta Means (mm)
Blade	5.60e-4	16.46
Foldchair	3.54e-5	1.16
Fridge	9.46e-4	11.71
Laptop	2.97e-3	4.85
Oven	3.62e-3	9.15
Scissor	1.45e-4	5.22
Stapler	1.53e-4	13.87
Storage	3.93e-3	13.16
USB	5.84e-4	8.61
Washer	8.48e-4	7.51

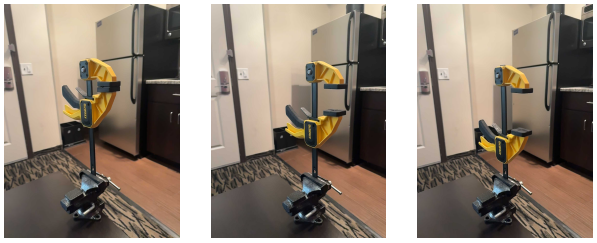

Fig. 7: The 3 real-world scenes captured, each with a different clamp position.

perception," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 13066–13073, IEEE, 2022.

[4] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[5] S. Lewis, J. Pavlasek, and O. C. Jenkins, "Narf22: Neural articulated radiance fields for configuration-aware rendering," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 770–777, IEEE, 2022.

[6] M. Ji, R.-Z. Qiu, X. Zou, and X. Wang, "Graspsplats: Efficient manipulation with 3d feature splatting," in *8th Annual Conference on Robot Learning*.

[7] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, *et al.*, "Sapien: A simulated part-based interactive environment," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11097–11107, 2020.

[8] J. Liu, A. Mahdavi-Amiri, and M. Savva, "Paris: Part-level reconstruction and motion analysis for articulated objects," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 352–363, 2023.

[9] J. Pavlasek, S. Lewis, K. Desingh, and O. C. Jenkins, "Parts-based articulated object localization in clutter using belief propagation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10595–10602, IEEE, 2020.

[10] J. Sturm, C. Stachniss, and W. Burgard, "A probabilistic framework for learning kinematic models of articulated objects," *Journal of Artificial Intelligence Research*, vol. 41, pp. 477–526, 2011.

[11] S. Lewis, T. Gao, and O. C. Jenkins, "Narf24: Estimating articulated object structure for implicit rendering," *arXiv preprint arXiv:2302.04264 arXiv:submit/5856538*, 2024.

[12] Q. Chen, M. Memmel, A. Fang, A. Walsman, D. Fox, and A. Gupta, "Urdformer: Constructing interactive realistic scenes from real images via simulation and generative modeling," in *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition@ CoRL2023*, 2023.

[13] Y. Weng, B. Wen, J. Tremblay, V. Blukis, D. Fox, L. Guibas, and S. Birchfield, "Neural implicit representation for building digital twins of unknown articulated objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3141–3150, 2024.

[14] J. Lei, Y. Wang, G. Pavlakos, L. Liu, and K. Daniilidis, "Gart: Gaussian articulated template models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19876–19887, June 2024.

[15] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, J. Kerr, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, *et al.*, "Nerfstudio: A modular framework for neural radiance field development," *arXiv preprint arXiv:2302.04264*, 2023.

[16] T. Xie, Z. Zong, Y. Qiu, X. Li, Y. Feng, Y. Yang, and C. Jiang, "Physgaussian: Physics-integrated 3d gaussians for generative dynamics," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4389–4398, 2024.

[17] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *Computer Vision–ACCV 2012: 11th Asian Conference on Computer Vision, Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part I 11*, pp. 548–562, Springer, 2013.

[18] M. Simonik, "Record3d." https://record3d.app/, 2024. Version 1.10.2, Mobile app for iOS.

[19] B. Dwyer, J. Nelson, and T. Hansen, "Roboflow (version 1.0) [software]." https://roboflow.com, 2024. computer vision.