Diorama: Unleashing Zero-shot Single-view 3D Indoor Scene Modeling Supplemental Materials

In this supplement, we provide additional details of our method (Appendix A) and experiments (Appendix B). In Appendix C, we present more qualitative results of Diorama on SSDB images (Fig. 5), as well as on real-world internet images (Fig. 6) and text-conditioned generated images (Fig. 7). Particularly, we demonstrate the potential application of Diorama in flexible scene editing in Fig. 2.

A. Method details

For all experiments involving LMM-based visual reasoning, we use the checkpoint "gpt-4o-2024-08-06" of GPT-4o. It costs approximately $0.12 on average per SSDB image for 344 examples.

A.1. Holistic scene parsing

Holistic scene parsing consists of several tasks to comprehensively understanding the semantics and geometry of a scene image that serve as inputs to downstream components, including object recognition and localization, and depth and normal estimation.

For open-world object recognition, given an image, we aim to identify the objects in the scene by prompting GPT-4o and run open-vocabulary detectors and segmentation models to obtain the bounding boxes and masks. Below, we provide prompts and additional details.

Prompt for identifying objects in an image:

To obtain object bounding boxes and masks for localization, we first run the detector OWLv2 [10] by providing text inputs of a template prompt *"a photo of CLASS"* and obtain detection results after non-maximal suppression. To ensure each object instance is only captured by one detection box to avoid repeated 3D shape retrieval for the same instance, we apply class-aware multiple-instance suppression where the bounding box of the same category with the largest intersection-over-self (IoS) is discarded. We then prompt SAM [8] with each detected bounding box to predict the segmentation mask for each object by assigning the one with the highest score.

A.2. LMM-powered scene graph generation

After obtaining the set of objects in the scene, we then use an LMM (e.g. GPT-4o) to obtain the scene-graph. The original input image is augmented by partitioning into semantically meaningful regions and overlaying a set of visual marks on it. Specifically, we use predicted object masks $\mathbf{M}$ to represent image regions of interest and each region is marked by its corresponding $\langle$*object id*$\rangle$ that can be recognized and referred by GPT-4o using its OCR and visual reasoning capabilities. The generated scene-graph provide information about relationships between objects, including support relations.

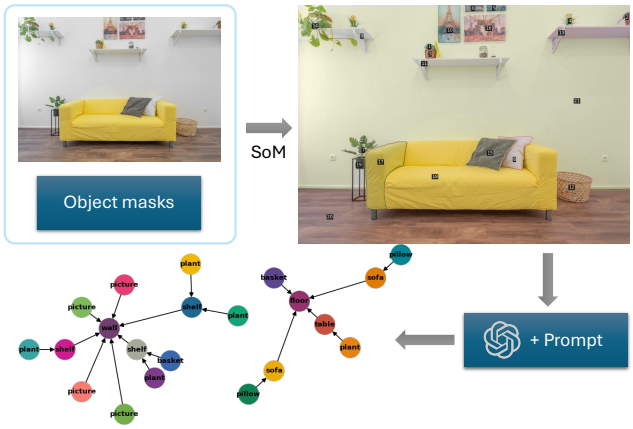

Figure 1. Illustration of scene graph generation using GPT-4o.

We specify to the LMM the desired output response in JSON format so it can be successfully executed by the program afterward. We also limit object supporting relations to be selected from ["placed on", "mounted on"]. To encourage the LMM to obey the provided numeric object marks and reduce hallucination errors, we use interleaved text prompt by incorporating the object marks and template response format into the scene graph generation prompt directly for symbolic reference.

Prompt for extracting a scene graph for objects in a image. Note the objects were already identified and provided as input.

A.3. Architecture reconstruction

To obtain a planar reconstruction of the architecture, we apply a three-step process: 1) segmenting out objects and in-

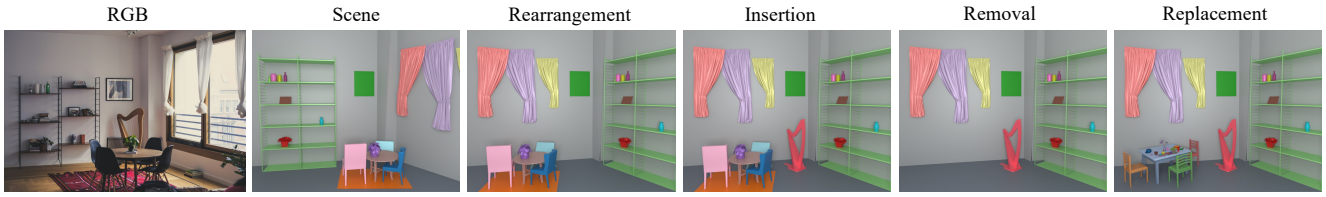

Figure 2. We show the potential of Diorama in flexible scene editing in terms of different editing types, including object rearrangement, insertion, removal and replacement.

painting to obtain empty rooms, 2) using depth estimation to obtain 3D points, and 3) clustering points by normals and plane fitting to obtain planar architecture surfaces.

Segmentation and inpainting to obtain empty room. We begin by obtaining object masks that are used to inpaint the images, leaving us with unfurnished scenes. For SSDB, we use dichotomous segmentation methods such as BiRefNet [22] and MVANet [20]. To maximize the recall of predicted masks, we apply each method separately twice by filling the mask predicted in the first iteration with white color then re-running the models on such images and merging the masks from both iterations.

We find that BiRefNet excels at producing sharp and complete masks for large objects and dense object arrangements, while MVANet has higher recall for scenes without a single cluster of objects. Therefore, we opt to merge the masks produced by these two methods. Additionally, we use SEEM [24] for all other experiments as we find that scenes featuring objects heavily dispersed around the scene are challenging for dichotomous segmentation networks. We note that the benefit of using dichotomous segmentation compared to other segmentation networks is in completeness of the masks (e.g., they cover the full area of the object with higher probability). Ensuring that the masks are as complete as possible is crucial for the further steps of PlainRecon.

After obtaining a segmentation of the objects, we 'erase' the objects by applying inpainting to defurnish the room. We use LaMa [17] to inpaint the background (similar to Yu et al. [21]). While recent advances in generative models are producing higher-quality inpainting, we find that even specialized inpainting for removal methods such as CLIP-Away [1] still suffer from hallucinating rather than inpainting empty background. This step should ideally produce perfectly defurnished and sharp image. In practice, the resulting image is not fully defurnished and the inpainted region is blurry. While the former remains a bottleneck of the pipeline, we notice that the latter is frequently alleviated by robustness of monocular depth estimation models which is the next step of the pipeline.

Depth estimation to obtain point cloud. After we have a defurnished room, we apply depth estimation to obtain a 3D point cloud of the scene. We compare two depth estimation models applied on the inpainted images - DepthAnythingV2 (DAv2) [19] and Metric3D (M3D) [7]. As we require normals for clustering points into planes, we leverage M3D normal estimation capabilities. We note that the predicted normals are more robust to the blurriness of the inpainting step compared to depth. We project the pixels to point clouds using the predicted metric depth. At the end of this process, we have point clouds with normals of unfurnished scenes, where the variation in normals should now be exclusively explained by presence of different planes in scene architecture.

Plane identification and fitting. The next step is normals-based clustering of the points into planar segments. We begin by pre-processing point clouds by applying voxel downsampling and removing statistical outliers with both using implementations from Open3D [23]. Since voxel downsampling bins the points in each voxel into one, the resulting point cloud has an approximately even distance between pairs of neighboring points. This ensures stability of hyperparameters in the subsequent steps and filters some noise introduced by depth estimation models. Next we run K-means clustering ($k_m = 12$ to account for noise) on normals to determine seed normals. We iteratively select the seed with the largest number of corresponding points, and cluster all the points that have normals within the angle threshold ($\alpha = 10$ in our experiments). We further apply DBScan [2] to separate the initial cluster as it is possible to have multiple walls with identical normals in the scene. The algorithm terminates when we either run out of seed normals or have less than the threshold number of unclustered points left ($N_{\text{min}} = 200$). Finally, we propagate the instance labels to full point clouds using KNN ($k_n = 1$). We assign the $floor$ label to the largest cluster with a normal pointing sufficiently upwards. Similarly, we filter out ceiling clusters as those points with a downward facing normal.

Once we have obtained the point clusters, we fit bounded planar segments to the points to obtain the final architecture. We start by fitting a plane using RANSAC [3] to obtain a plane equation, and then estimate the bounding boxes of architecture elements. To obtain tight boxes, we compute convex hulls of point cloud segments, project the convex hull vertices onto 2D plane and apply the rotating calipers algorithm [14] to obtain the bounding box with the small-

Method	Scene-aware Alignment ↑				Collision ↓	Relation ↑
	rAcc	tAcc	sAcc	Acc		
BM baseline	0.34	0.84	0.44	0.15	12.29	0.58
ZSP [5]	0.32	0.88	0.50	0.18	8.87	0.59
ZSP w/ DinoV2	0.33	0.81	0.48	0.20	9.80	0.57
ZSP w/ ft DinoV2	0.34	0.83	0.50	0.21	9.97	0.59
GigaPose [11]	0.36	0.89	**0.67**	0.24	7.43	0.61
Ours	**0.41**	**0.91**	0.61	**0.28**	**6.93**	**0.61**

Table 1. Additional comparison of zero-shot pose estimation methods for the 9-DoF CAD alignment task on SSDB images using estimated depth by Metric3DV2.

Groups	#Cats / #Insts	rAcc ↑	tAcc ↑	sAcc ↑	Acc ↑
Household	466 / 5577	0.46	**0.87**	**0.71**	0.32
Furniture	24 / 1137	**0.54**	0.86	**0.71**	**0.39**
Occluded	369 / 3175	0.44	0.84	0.62	0.28
Complete	401 / 3539	**0.50**	**0.89**	**0.78**	**0.38**
Supported	468 / 5715	**0.48**	**0.87**	**0.71**	**0.34**
Supporting	79 / 1634	0.46	0.84	0.69	0.31

Table 2. Averaged alignment results across different SSDB object groups. For each group, we count the number of fine-grained categories and object instances. Occluded objects are those with occlusion ratio above a threshold 5% of pixels.

# retrievals	Ground Truth Depth				Estimated Depth			
	rAcc	tAcc	sAcc	Acc	rAcc	tAcc	sAcc	Acc
1	0.22	0.92	0.52	0.11	0.21	0.88	0.47	0.10
4	0.32	0.96	0.68	0.20	0.32	0.93	0.63	0.18
8	**0.37**	**0.97**	**0.72**	**0.23**	**0.37**	**0.94**	**0.68**	**0.21**

Table 3. 9D CAD alignment results given a different number of retrieved 3D shapes using either ground-truth or estimated depth.

est area. Finally, we convert vertices back into 3D. While such initialization provides good plane bounds, we need to refine the plane further to account for imperfections of previous steps. We begin by making sure that all the walls are orthogonal to the floor, this is simply done by solving a system of linear equations that finds a vector orthogonal to the intersection line of floor and wall and lies on the floor plane. We proceed to use this vector as a new normal for our wall equation. Then we proceed to refine the plane bounds by finding the intersections of each architecture element, projecting two closest vertices of one plane onto the intersection line and adjusting the vertices of the plane we projected onto to the projected vertices of the other plane, effectively connecting them. Finally, we duplicate the vertices with a small offset along the direction opposite to normal and export the meshes of architectural components.

Ablation	Collision ↓	Scene Structure ↑		
		orientation	placement	overall
all-in-one	12.80	0.19	0.92	0.16
stage-wise	**3.78**	**0.98**	**0.95**	**0.93**

Table 4. Comparison between all-in-one and stage-wise optimization.

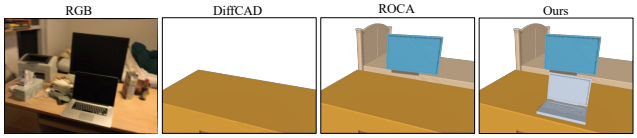

Figure 3. Qualitative comparison between different ground-truth evaluation sets used in DiffCAD, ROCA and ours.

A.4. Multimodal 3D shape retrieval

For each observed object in the image, we use a two-step process for retrieving matching shapes from our shape database. We first use a text query to retrieve objects matching the semantic category, and then an image query to re-rank the retrieved candidates. We find this two-stage approach helps ensure that retrieved objects are of the correct semantic class, as it is possible for semantically different objects to be geometrically similar (e.g. books vs cardboard box).

We construct the text query from the template *"a photo of CLASS"* (similar to in detection phase). For the image query, we use object crop extracted from the original image and mask out background and occluders using bounding box b_i and mask m_i. We also carefully separate retrieval of supporting and non-supporting objects according to the supporting hierarchy in the scene graph since supporting objects need to be pre-processed to represent only one object entity (not containing smaller sub-objects).

A.5. Zero-shot object pose estimation.

We pre-compute T multiview renderings and depth maps for each retrieved CAD model s_i. Each query object crop I^{o_i} and set of multiview images $\{I^{s_{i,j}}\}$ is encoded into normalized patch features $\mathcal{F}(I)$ using DinoV2. For each patch embedding from the query image, we construct a semantic correspondence to a patch embedding from each rendering view with the minimal cyclical distance [5, 11]. We compute correspondence score as cosine similarity between patch embeddings. Specifically, given a pair of query image I_q and reference rendering image I_r, we construct a "cyclical distance" map and define a similarity score using their corresponding DINOv2 patch features, f_q and f_r, and feature masks m_q and m_r. For each query feature f_q^i at the patch location i, we compute its cyclical patch i' in I_q as

Method	supervision	#hypo	bed	bkshlf	cabinet	chair	sofa	table	bin	bathtub	display	others	cls. avg	ist. avg
ROCA	✓✓	-	7.02	3.62	7.56	20.03	5.26	9.16	13.19	8.11	13.21	0.00*	8.72	12.87
DiffCAD	✓	1	7.02	0.00	3.36	9.38	6.58	1.58	0.00*	0.00*	0.00*	0.00*	2.79	4.59
DiffCAD	✓	5	12.28	0.72	6.72	14.2	7.89	1.58	0.00*	0.00*	0.00*	0.00*	4.34	6.84
DiffCAD	✓	10	7.02	1.45	4.21	11.41	5.26	1.26	0.00*	0.00*	0.00*	0.00*	3.06	5.41
Ours	✗	-	0.00	0.74	4.24	9.72	3.95	4.27	0.00	0.00	10.36	5.56	3.33	6.66

Table 5. Object-focus alignment accuracy on the Scan2CAD benchmark. ROCA is fully-supervised using in-domain data. DiffCAD is weakly-supervised using synthetic data.

Figure 4. More comparison examples on SSDB.

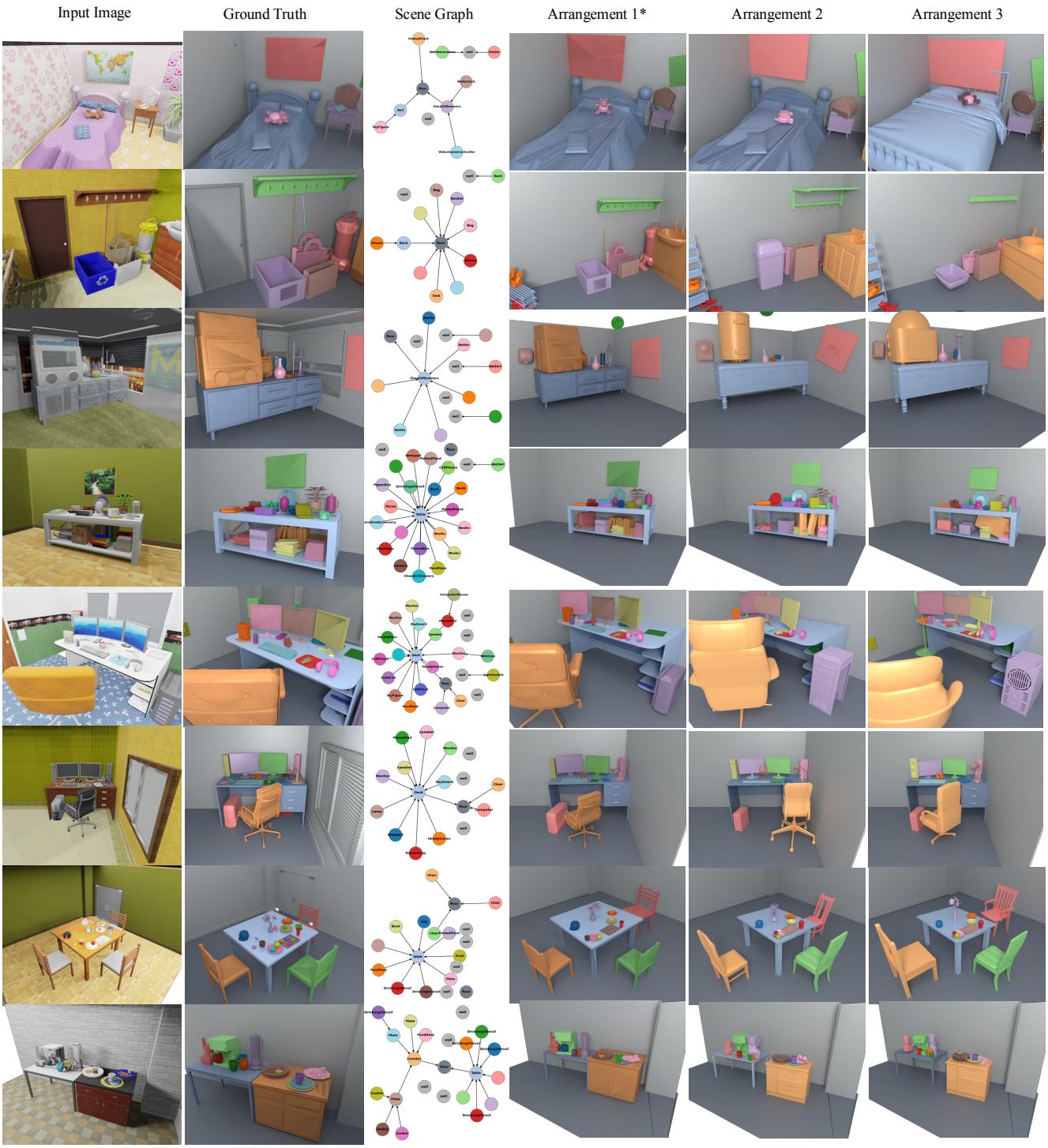

Figure 5. More Diorama examples on SSDB images.

follows:

$$i' = \underset{w, m_q^w > 0}{\operatorname{argmax}} S(f_q^w, f_r^j)$$

$$j = \underset{k, m_r^k > 0}{\operatorname{argmax}} S(f_q^i, f_r^k)$$

where $S(\cdot, \cdot)$ denote as cosine similarity and a cyclical distance map is constructed as D with $D_i = -||i, i'||_2$. In the end, we build the feature correspondences by taking at most K query-reference patch pairs $(i, j) \in \mathcal{N}$ with top-K mini-

Figure 6. More examples for real-world internet images.

mal cyclical distances in D. To ensure only keeping correspondences with strong similarity, we empirically set similarity score threshold as 0.7. The overall similarity score between I_q and I_r is defined as the average of similarities of feature correspondences:

$$\text{sim}(I_q, I_r) = \frac{1}{|\mathcal{N}|} \sum_{(i,j)\in\mathcal{N}} S(f_q^i, f_r^j)$$

The query image is matched to the most similar multiview rendering with the maximal averaged correspondence-wise feature similarity to produce a coarse pose hypothesis.

A.6. Scene optimization

During scene optimization, we take the retrieved objects, initial object pose estimates, and optimize the placements so that the relationships in the scene graph are respected. In addition to support relations described in the scene graph, we further establish an adherence relationship between each object on the floor and potential walls using a heuristic algorithm that judging whether the closest distance from the object's surface center to a candidate wall is within a predefined threshold. We hence also optimize object position and rotation based on both the support and adherence relationship. In the main paper, we described the different stages of optimization. Here we provide more details on how support is enforced while ensuring there is sufficient space, and no inter-penetrations.

Support. A key relationship is that of support, for which we want to ensure that each object is properly placed in contact with their supporting object. To do so, for each object in our database, we precompute candidate *support surfaces* by identifying roughly planar surfaces on the object. Based on estimated initial object poses and the scene graph $\mathbf{G}$ describing object relationships, for each a pair of interacting objects, we compute a pair of contact surface from the supported object and support surface from the supporting object. The support surface is determined by selecting the surface with the minimal distance to the center of the supported object. The contact surface is selected to be the one having the closest direction with the support direction.

Space. Particularly for the *Space* stage, we define a *supporting volume* for each object by extruding its identified support surface to an extent hitting another surface in the vertical direction. An object is properly supported only if it does not exceed the bounds of its corresponding supporting volume. We formulate the term e_{vol} as the sum of distance from the corners of the contact surface to the sides of the supporting volume and the vertical distance between the centers of the object bounding box and the supporting volume.

Optimization. In each optimization stage, we use a separate SGD optimizer with initial learning rate 0.01 and momentum 0.9 for corresponding pose parameters, except for the *Space* stage where we set initial learning 0.001 for the

Figure 7. More examples of applying Diorama in a text-to-scene setting.

scale parameter. We also decay the learning rate by 0.1 every 50 steps. We run 200 steps in total in each optimization stage. We describe the objective function of each optimization stage as below, including weight hyperparameters:

$$\text{Stage 1: } e_1 = 3 \cdot e_{\text{align}} + e_{\text{sem}} \tag{1}$$

$$\text{Stage 2: } e_2 = 5 \cdot e_{\text{place}} + e_{\text{rel}} \tag{2}$$

$$\text{Stage 3: } e_3 = e_{\text{vol}} \tag{3}$$

$$\text{Stage 4: } e_4 = 5 \cdot e_{\text{place}} + e_{\text{col}} \tag{4}$$

B. Additional experiments

Implementation details. We render SSDB scene images of size 1008×784. Considering computation efficiency and good coarse pose selection, we render 180 gray-color multiviews of size 224×224 for each 3D shape from predefined camera viewpoints to focus on geometry-wise semantic similarity and leave out effect of texture. For zero-shot pose estimation, we use the fine-tuned ViT-L of DinoV2 [12] following [11] to embed 14×14 image patches. We run experiments on one Nvidia RTX 4090 GPU.

B.1. Details of comparing against ACDC

We run ACDC on all SSDB images with the default configuration. For both ACDC and Diorama, we feed ground-truth 2D object bounding boxes and segmentation masks into systems to avoid 2D perception errors for analysis. Both systems retrieve object from out-of-distribution 3D shape collections. Since ACDC is developed upon the OmniGibson simulation platform, it retrieves from the built-in set of approximately 8,800 OmniGibson CAD objects for convenient deployment in the physics engine. For Diorama, since we do not specify certain CAD file formats to be compatible with a physics engine, we are able to include more 3D shapes for retrieval from different sources. In particular, we compose a set of 25K 3D shapes for Diorama to retrieve from. For ACDC, overall runtime is dominated by network API calls to the LLM (GPT4o). ACDC calls GPT4o three times per detected object, while Diorama calls GPT4o twice in total, irrespective of the number of objects. For the user study, we randomly sample 48 images and corresponding results to ask the participants to assess the quality of single-view 3D scene modeling in terms of object matching and overall scene quality. For object matching, we consider both

semantic correctness and geometric similarity. For overall scene quality, we consider the accuracy of architecture reconstruction and object arrangement, and physical plausibility of the whole scene. The question order is randomly shuffled to the participant.

B.2. Architecture reconstruction evaluation

We compare our proposed PlainRecon against a recent method for obtaining 3D room layout via render-and-compare (RaC) [16]. RaC is a common architecture reconstruction baseline, and though follow-ups exist they introduce marginal improvements while having less reliable or no available public implementation [13, 18]. For a fair comparison, we provide RaC with inputs from more modern backbones compared to the ones used in the original implementation. We use DepthAnythingV2 or Metric3D depth and PlaneRecTR [15] planar segments.

As ACDC outputs plane parameters and masks but does not output actual mesh planes, we perform a simple extraction procedure. First, we back-project the depth used by ACDC to the point cloud. Then, similarly to our method, we run RANSAC for each architectural element based on image segmentation masks to fit the plane, resulting in selecting inlier points that correspond to the largest planar region of the point cloud. We obtain the oriented bounding box from the inliers and extract mesh from it. Finally, we align the normals of the architectural elements with the normals ACDC used to optimize object placement.

B.3. Object alignment on estimated depth

Tab. 1 presents the comparison of different zero-shot pose estimation methods on the 9D CAD alignment task given predicted depth by Metric3D. The results align with our observation under the ground-truth depth setting.

B.4. Performance for different object groups

We also analyze post-optimized object pose according to different object groups each object belongs to in Tab. 2. Since we aim for an open-world system that generalizes to long-tail categories in real life, we divide objects into three subgroups: household objects/common furniture, occluded/complete objects, and supported/supporting objects, rather than a coarse set of preselected categories as in prior work [4, 6]. We find that the performance is reduced for dominant household items, occluded objects, and supporting objects.

B.5. Multiple retrievals

In Tab. 3, we investigate the benefits of having more retrieved 3D shapes for correspondence computation and coarse pose proposal. It turns out that alignment accuracy increases with potentially more different pose initialization.

B.6. Different optimization strategy

We investigate the benefits of using a stage-wise optimization procedure rather than a more common all-in-one strategy where all terms are accumulated for optimizing simultaneously in Tab. 4. It shows that we obtain significant gains by decomposing the entire optimization task into separate stages.

B.7. Quantitative results on ScanNet

Tab. 5 shows quantitative comparison between ROCA, DiffCAD and ours on the proposed evaluation set. Following prior work [4, 6, 9], we report the object alignment accuracy where a CAD model is considered correctly aligned if the translation error $\leq$ 20cm, the geodesic rotation error $\leq 20°$, and the scale ratio $\leq 20\%$. We note that ROCA is trained end-to-end using imperfect Scan2CAD annotations and DiffCAD is trained on the synthetic data per category. Both ROCA and DiffCAD cannot generalize to unseen objects during training (indicated as gray-colored numbers). Our zero-shot method achieves competitive performance against DiffCAD that further exhibits performance degradation under the probabilistic setting due to the partial observations of commonly occluded objects. Fig. 3 visually shows differences between ground-truth evaluation sets used in DiffCAD, ROCA and ours.

C. Qualitative examples

We provide additional examples of generated scenes in Figures 4 to 7. In Figure 5, we provide plausible arrangements based on renders from SSDB, as well as the associated predicted scene-graphs. With Diorama, we can produce alternative arrangements that use different objects, while respecting the spatial relationships of the original image (e.g. picture on the wall, monitor on the desk).

We further showcase arrangements from real-world images (Figure 6) and images generated via text-to-scene (Figure 7).

References

[1] Yiğit Ekin, Ahmet Burak Yildirim, Erdem Eren Caglar, Aykut Erdem, Erkut Erdem, and Aysegul Dundar. CLIP-Away: Harmonizing focused embeddings for removing objects via diffusion models. In *Advances in Neural Information Processing Systems*, 2024. 2

[2] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996. 2

[3] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2

[4] Daoyi Gao, David Rozenberszki, Stefan Leutenegger, and Angela Dai. DiffCAD: Weakly-supervised probabilistic CAD model retrieval and alignment from an RGB image. *ACM Transactions on Graphics (TOG)*, 43(4):1–15, 2024. 8

[5] Walter Goodwin, Sagar Vaze, Ioannis Havoutis, and Ingmar Posner. Zero-shot category-level object pose estimation. In *European Conference on Computer Vision*, pages 516–532. Springer, 2022. 3

[6] Can Gümeli, Angela Dai, and Matthias Nießner. ROCA: Robust CAD model retrieval and alignment from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 8

[7] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Hao Chen, Kaixuan Wang, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3D v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *arXiv preprint arXiv:2404.15506*, 2024. 2

[8] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 1

[9] F. Langer, G. Bae, I. Budvytis, and R. Cipolla. SPARC: Sparse render-and-compare for CAD model alignment in a single RGB image. In *Proc. British Machine Vision Conference*, London, 2022. 8

[10] Neil Houlsby Matthias Minderer, Alexey Gritsenko. Scaling open-vocabulary object detection. *NeurIPS*, 2023. 1

[11] Van Nguyen Nguyen, Thibault Groueix, Mathieu Salzmann, and Vincent Lepetit. GigaPose: Fast and Robust Novel Object Pose Estimation via One Correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 3, 7

[12] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 7

[13] Denys Rozumnyi, Stefan Popov, Kevis-Kokitsi Maninis, Matthias Nießner, and Vittorio Ferrari. Estimating generic 3D room structures from 2D annotations. In *Advances in Neural Information Processing Systems*, 2023. 8

[14] Michael Ian Shamos. *Computational geometry*. Yale University, 1978. 2

[15] Jingjia Shi, Shuaifeng Zhi, and Kai Xu. PlaneRecTR: Unified query learning for 3D plane recovery from a single view. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9377–9386, 2023. 8

[16] Sinisa Stekovic, Shreyas Hampali, Mahdi Rad, Sayan Deb Sarkar, Friedrich Fraundorfer, and Vincent Lepetit. General 3D room layout from a single view by render-and-compare. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 187–203. Springer, 2020. 8

[17] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor S. Lempitsky. Resolution-robust large mask inpainting with Fourier convolutions. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, 2022. 2

[18] Cheng Yang, Jia Zheng, Xili Dai, Rui Tang, Yi Ma, and Xiaojun Yuan. Learning to reconstruct 3d non-cuboid room layout from a single rgb image. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2022. 8

[19] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv:2406.09414*, 2024. 2

[20] Qian Yu, Xiaoqi Zhao, Youwei Pang, Lihe Zhang, and Huchuan Lu. Multi-view aggregation network for dichotomous image segmentation. *arXiv:2404.07445*, 2024. 2

[21] Tao Yu, Runseng Feng, Ruoyu Feng, Jinming Liu, Xin Jin, Wenjun Zeng, and Zhibo Chen. Inpaint anything: Segment anything meets image inpainting. *arXiv:2304.06790*, 2023. 2

[22] Peng Zheng, Dehong Gao, Deng-Ping Fan, Li Liu, Jorma Laaksonen, Wanli Ouyang, and Nicu Sebe. Bilateral reference for high-resolution dichotomous image segmentation. *CAAI Artificial Intelligence Research*, 3:9150038, 2024. 2

[23] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 2

[24] Xueyan Zou, Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Wang, Lijuan Wang, Jianfeng Gao, and Yong Jae Lee. Segment everything everywhere all at once. In *Advances in Neural Information Processing Systems*, 2023. 2