Supplementary Materials for Point Scene Understanding via Disentangled Instance Mesh Reconstruction

Jiaxiang Tang[1], Xiaokang Chen[1], Jingbo Wang[2], and Gang Zeng[1]

[1] Key Laboratory of Perception (MoE), School of AI, Peking University
[2] Chinese University of Hong Kong
{tjx, pkucxk}@pku.edu.cn, wj020@ie.cuhk.edu.hk, zeng@pku.edu.cn

A Data Processing

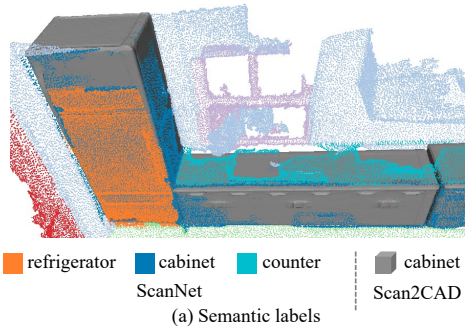
In this section, we detail the data processing in our experiments.

A.1 Semantic Segmentation Labels

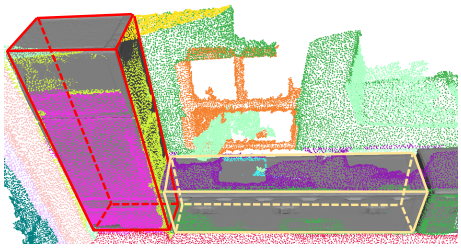
To enable training with both ScanNet point-level annotations and Scan2CAD instance-level annotations, we first make a compatible label system that contains 25 semantic classes, including 2 stuff classes and 23 object classes. This label system basically extends the ScanNet 20-class label system by subdividing some classes, to make sure we can convert safely to the CAD 8-class label system. The full label mapping is shown in Figure 1 (d). For the semantic segmentation task, we use all the 25 classes as the supervision.

A.2 Instance Segmentation Labels

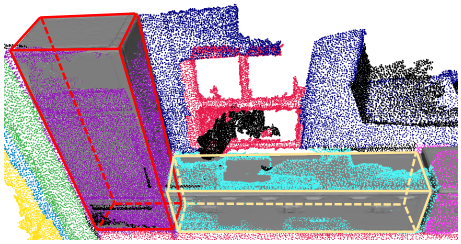
Due to the complicated containment relationships between these two label systems as shown in Figure 1 (d), the ScanNet instance segmentation labels are still inconsistent with Scan2CAD models. For example, the *cabinet* class in Scan2CAD may refer to *cabinet, counter, refrigerator* in ScanNet instance segmentation annotations. Directly using the ScanNet annotations may lead to inaccurate instance supervision for our task, such as the instance centers and point-wise offsets. Figure 1 (a) and (b) shows an example of such a condition. Therefore, we need to merge the instances that belong to the same CAD class. Since these instances mainly belong to the *cabinet* class, we use the instance bounding boxes from the Scan2CAD dataset to rectify the ScanNet instance segmentation annotations, as illustrated in Figure 1 (c). These new instance labels are used to supervise the instance segmentation task in our pipeline. Please note that we do not manually annotate this dataset, but only rearrange the existing annotations to make the training consistent.



(a) Semantic labels



(b) Original Instance labels



(c) Rectified Instance labels

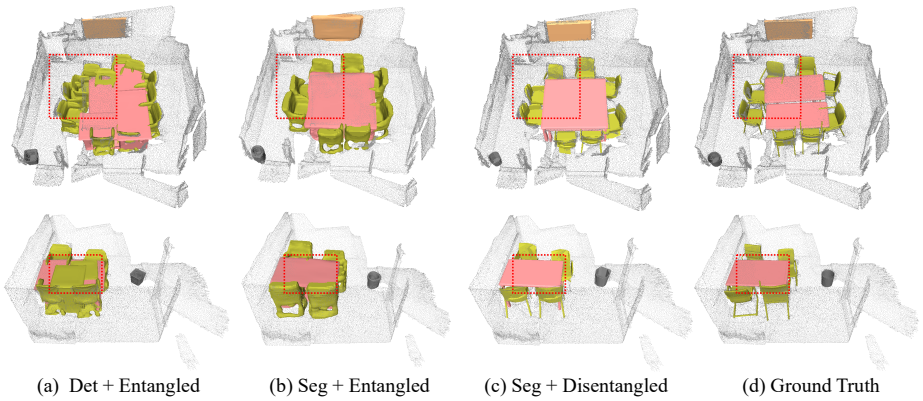
Compatible	ScanNet	CAD
wall	wall	-
floor	floor	
chair	chair	chair
toilet	toilet	
sofa	sofa	sofa
kitchen_cabinet	cabinet	-
cabinet		cabinet
counter	counter	
refrigerator	refrigerator	
sink	sink	bathtub
bathtub	bathtub	
display	-	display
other_shelf		bookshelf
bookshelf	bookshelf	
table	table	table
desk	desk	
other_table	otherfurniture	
trash_bin		trash_bin
otherfurniture		-
bed	bed	
door	door	
window	window	
picture	picture	
curtain	curtain	
shower curtain	shower curtain	

(d) Label mapping

Fig. 1: **Semantic and Instance Segmentation Labels.** (a) The *cabinet* class in Scan2CAD may refer to *cabinet, counter, refrigerator* in ScanNet. (b) The original instance labels in ScanNet. A single object is divided into multiple parts, such as the refrigerator marked with the red box. (c) The rectified instance labels are consistent with Scan2CAD meshes. (d) Label mapping between different tasks.

B Experimental Results

Besides the results analyzed in the main paper, we conduct more ablations on the proposed method and show more qualitative comparisons.



(a) Det + Entangled (b) Seg + Entangled (c) Seg + Disentangled (d) Ground Truth

Fig. 2: More qualitative comparisons. The segmentation backbone can improve the quality of proposals by reducing false positives as marked by the red boxes, and the disentangled mesh reconstruction strategy improves the quality of generated meshes.

Table 1: **Ablation study on the proposed method.** RfD-Net is referred to as 'Det + Entangled', and our method is referred to as 'Seg + Disentangled'.

pipeline	IoU@0.25	CD@0.1	LFD@5000
Det + Entangled	42.52	46.37	28.59
Seg + Entangled	43.95	47.46	28.67
Seg + Disentangled	**46.34**	**52.39**	**29.47**

B.1 More Ablations on the Proposed Method

We perform ablation studies to verify the two contributions of the proposed method: (i) instance segmentation → instance mesh completion pipeline, and (ii) disentangled instance mesh reconstruction (DIMR) strategy. Results are listed in Table 1. We design three experiments: (i) 'Det + Entangled', (ii) 'Seg + Entangled', and (iii) 'Seg + Disentangled'. 'Entangled' means directly learning occupancy values of complete objects based on incomplete point observations, which is proposed in [3]. 'Disentangled' means our DIMR strategy. RfD-Net [3] is referred to as 'Det + Entangled' and the proposed method is referred to as 'Seg + Disentangled'. Comparing first two rows in Table 1, we find that the segmentation-then-completion pipeline achieves slightly better performance than detection-then-completion pipeline, due to the reduction of false positive proposals (we also show some visualizations in Figure 2 to verify this). Furthermore, when we adopt the disentangled instance mesh reconstruction strategy, the performance improves significantly, *e.g.*, from 43.95 to 46.34 in terms of IoU@0.25. Overall, DIMR makes the largest contribution to the performance gain.

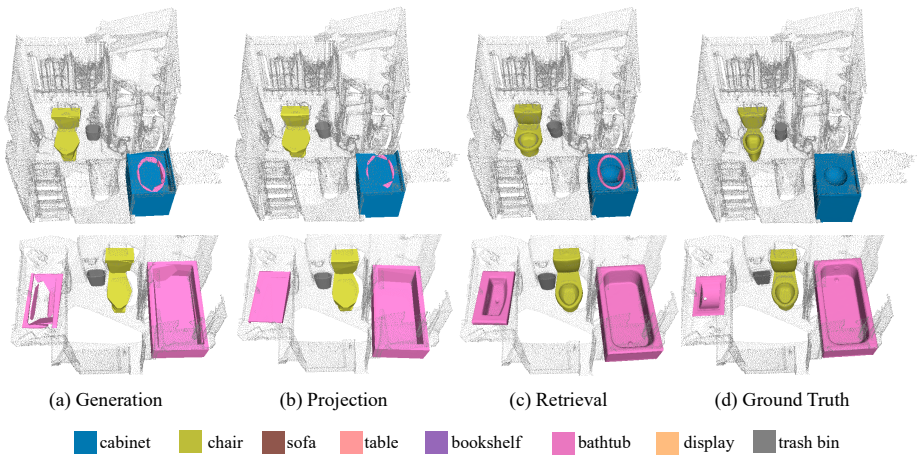


Fig. 3: More comparisons on mesh generation, mesh retrieval and assisted mesh generation.

B.2 More Visual Comparisons between the Variants of Our Method

We also provide more comparisons between the three variants of our method in Figure 3. The BSP mesh decoder used in our generation pipeline suffers from generating concave objects such as the toilet, but our retrieval variant can handle these objects well.

C Details of the Mesh Autoencoder

In this section, we discuss the details of the pre-trained mesh autoencoder, *i.e.*, the BSP-CVAE network.

C.1 Network Structure

We follow [2] to build a BSP-Net with some modifications, as illustrated in Figure 4. The modifications we made include: 1) change the original autoencoder (AE) to a variational autoencoder (VAE), so the encoder outputs latent distributions and is supervised by an extra KL loss; 2) add the class label to the input, so the VAE turns to a conditional VAE (CVAE). These make the network more suitable to our task, since we need to extract latent codes from class-aware CAD models and also generate 3D meshes given latent codes and class labels.

C.2 Training Details

We use the CAD models from the ShapeNet dataset [1] to train the network. Specifically, we use the subset of eight classes provided by [3]. For each CAD

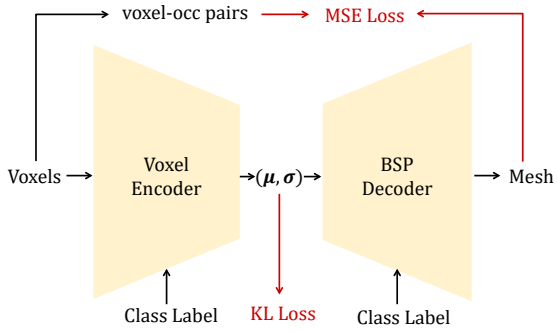


Fig. 4: BSP-CVAE architecture.

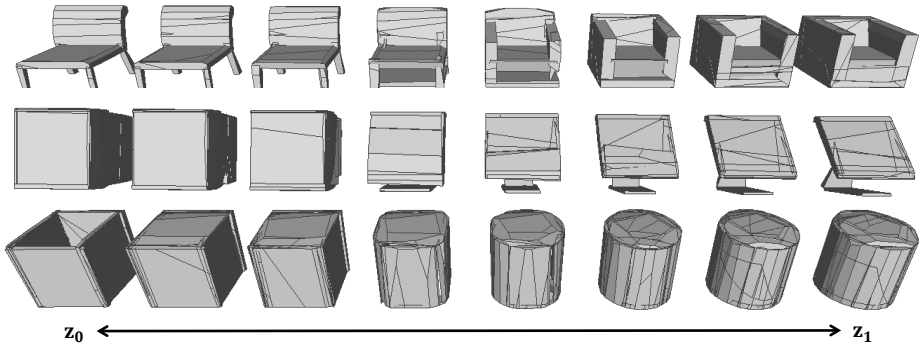


Fig. 5: Latent space interpolation results of BSP-CVAE. We randomly choose two latent codes from ShapeNet meshes (left- and right-most) and interpolate linearly between them and visualize the generated meshes.

mesh, we voxelize it into $64 \times 64 \times 64$ volume as the input for the voxel encoder. 2048 empty voxels and 2048 occupied voxels are sampled from this volume at each training step to optimize the network. We follow [2] to train the BSP-CVAE for 800 epochs, including 400 epochs for the continuous phase and 400 epochs for the discrete phase.

C.3 Latent Code Interpolation for BSP-CVAE

To show the generative property of the trained BSP-CVAE, we provide some latent code interpolation experiments as shown in Figure 5. The model is capable of generating high quality meshes, and successfully learns the structural changes during interpolation of two given shapes, which meets our need for the mesh autoencoder.

References

1. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015) 4
2. Chen, Z., Tagliasacchi, A., Zhang, H.: Bsp-net: Generating compact meshes via binary space partitioning. In: CVPR (2020) 4, 5
3. Nie, Y., Hou, J., Han, X., Niessner, M.: Rfd-net: Point scene understanding by semantic instance reconstruction. In: CVPR. pp. 4608–4618 (June 2021) 3, 4