Supplementary Material for SEGCloud: Semantic Segmentation of 3D Point Clouds

Lyne P. Tchapmi lynetcha@stanford.edu
Christopher B. Choy chrischoy@ai.stanford.edu
Iro Armeni iarmeni@cs.stanford.edu
JunYoung Gwak jgwak@stanford.edu
Silvio Savarese ssilvio@stanford.edu

Stanford University

This document presents additional details and qualitative results for the framework presented in our main paper. Section S1 reports the particulars of our framework's implementation. The following section S2 offers details and results on the effect of using end-to-end training versus separate CRF intialization. The remaining of the document focuses on additional aspects of the evaluation and experiments. The experimental setup is detailed in Section S3. The characteristics of the datasets used in our evaluation are outlined in Section S3.1. Section S3.2 defines the metrics used in evaluating our framework. Finally, qualitative results of our framework on all four datasets are illustrated in Section S3.3.

S1. Implementation

This section provides additional implementation details, including procedures for 3D data augmentation, data preparation, training, as well as the programming framework.

S1.1. Augmentation Procedures for 3D data

Most of the datasets we used are small to medium in scale. To make up for the lack of data, we perform a series of augmentations for 3D data. We apply the following data augmentations on-the-fly to increase randomness in the data and save storage space.

Color Augmentation: Color augmentation is a popular data augmentation technique for image datasets. We leverage it in our work by randomly varying the R, G and B channels of each observation within the range ± 2.5 for each channel.

Geometric augmentation: We also leverage 2 simple geometric augmentations: *random rotation* and *scaling*. We randomly rotate 3D observations around the axis along the gravity direction to mimic a change of viewpoints in a scene. During training, we sample rotation angles in the continuous range of $[0°, 360°]$ and rotate the point cloud on-the-fly. We also scale the data by a small factor that is uniformly sampled in the range $[0.9, 1.1]$ to make the network invariant to small changes in scale.

Points Subsampling: We also use a random subsampling of points in highly dense datasets, specifically, the Stanford Large-Scale 3D Indoor Spaces Dataset (S3DIS) [1] and the Semantic3D.net [4]. During training, we sample points in a scene by a factor empirically chosen based on the number of points in the given point cloud crop (see Table S1). For point clouds having more than $1e^5$ points, the subsampling factor for S3DIS is kept at 10 since the density of the point cloud is relatively constant in this dataset. The Semantic3D.net dataset on the other hand has varying density and we use three values of the sub-sampling factor (10, 50 and 100), as shown in Table S1. This sub-sampling process aims at building invariance to missing points, and increasing the speed of the training process. At test time, the algorithm is evaluated on all input points without sub-sampling.

The above random transformations and sub-sampling allow us to increase the effective size of each dataset and can help the network build invariance to rotation/viewpoint changes, as well as reduced and varying context.

Table S1. **Cloud Sub-sampling Factor (For training-only)**

Threshold (#points) / **Dataset**	$1e^5$	$1e^6$	$1e^7$
S3DIS	10	10	10
Semantic3D.net	10	50	100

S1.2. Input Preparation

The large scale 3D observations are split into areas of at most $5m$ in the X, Y and Z dimensions, where Z is the gravity axis. One notable exception is the S3DIS dataset, which provides fully reconstructed 3D point clouds of indoor buildings spaces. For this dataset, we limit the X and Y dimensions to $5m$ like rest of the datasets, but keep the entire Z extent, which allows to include both the ceiling and floor in every crop. During training, such $5m$ cropped sub-area overlap with adjacent sub-areas by $0.5m$. There is no overlap at test time in order to obtain a single prediction per point.

Sub-areas are then voxelized with a $5cm$ resolution to obtain a maximum input volume of $100 \times 100 \times 100$. This granularity provides a balance between memory requirements and an adequate representation of the 3D space without information loss. Each voxel has one to five associated channels that correspond to its binary occupancy (1-occupied, 0-empty), RGB value normalized within the range $[0, 1]$, and sensor intensity when available (Semantic3D.net dataset). The sensor intensity is mean centered and normalized using the mean and range of the training data distribution.

S1.3. Training

Training is performed in a 2-step process similar to [7]. This process is illustrated in Figure S1. In the first training stage, we use the Trilinear Interpolation layer to map the voxel-wise predictions to point-wise predictions and minimize the point-wise loss. We train 3D-FCNN with Trilinear Interpolation layer for 200 epochs with a learning rate between $1e^-5$ and $1e^-3$, and reduce it by a factor of 10 every 50 epochs. In the second training stage, we combine the pre-trained 3D-FCNN, the Trilinear Interpolation layer and the CRF, and train the whole system end-to-end. The base learning rate in this stage is set to a value between $1e^-7$ and $1e^-5$, and the training is performed for 2 epochs. We use a learning rate multiplier of $1e^4$ and $1e^3$ for the CRF s bilateral weights and compatibility matrix, however we did not extensively study the effect of these parameters. In most cases, the training of the second stage converges within a few hundred iterations (Convergence is determined using a validation set). In the CRF formulation, although the kernel weights w_s, w_b and the compatibility matrix μ are learned using gradient descent, the kernel bandwidth parameters $\theta_\alpha, \theta_\beta, \theta_\gamma$ are not learned within our efficient variational inference framework. Thus, we used grid search or fixed values for some parameters following [5]. We fix θ_γ at 5cm, θ_β at 11, and use a validation set to search for an optimal value of θ_α. We limit our search to the range $[0.1, 3.2]m$. When no RGB information is available, we instead searched for θ_γ in the same range and did not use the bilateral filter. The kernel weights and compatibility matrix are learned during training. Similar to [7] we use 5 CRF iterations during training and 10 CRF iterations at test time.

S2. Effect of end-to-end training vs separate CRF initialization

We performed an experiment to evaluate the effect of end-to-end training versus separately initializing the CRF module. For the separate initialization, we set the theta parameters to the optimal joint training values we found during end-to-end training, the spatial weight to 3, and the bilateral to 5 for all experiments. Results show that joint training performs better than separate CRF initialization especially in mAcc metric (see Table S3).

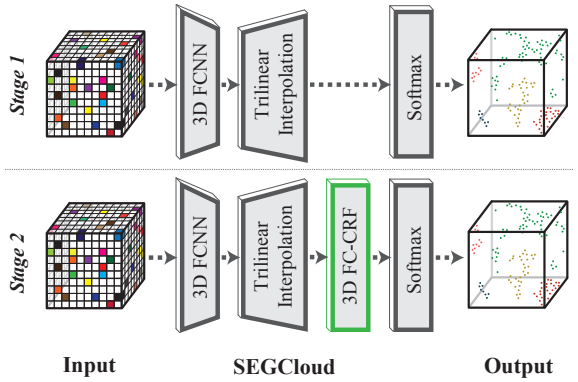

Figure S1. We follow a 2-stage training by first optimizing over the point-level unary potentials (no CRF) and then over the joint framework for point-level fine-grained labeling.

S3. Experimental and Evaluation Setup

S3.1. Datasets

We now present the characteristics of the datasets we use to evaluate our framework. The datasets we chose for evaluation are Semantic3D.net [4], the Stanford Large-Scale 3D Indoor Spaces Dataset (S3DIS) [1], KITTI [3, 2], and NYU V2 [6]. As shown in Table S2, our framework is general in that it can handle point clouds from various sources, both indoor and outdoor environments, as well as partial and fully reconstructed point clouds. Specifically, two of the datasets are collected from indoor environments and two from outdoor environments. They also cover a variety of data acquisition methods, including laser scanners (Semantic3D.net, KITTI), Kinect (NYU V2), and MatterPort (S3DIS). Moreover, the S3DIS is a fully reconstructed point cloud dataset, while NYU V2 provides point clouds extracted from a single frame RGB-D camera. The size of the training sets also vary from 12 million training points for the KITTI dataset to 1.9 billion training points for Semantic3D.net (excluding the validation set).

S3.2. Evaluation Metrics

We use two main metrics for our evaluation: *mean class accuracy (mAcc)* and *mean class IOU (mIOU)*, where IOU is defined similarly to the Pascal segmentation convention. Accuracy per class is defined as:

$$acc_i = \frac{tp_i}{gt_i} = \frac{tp_i}{tp_i + fn_i}, \tag{1}$$

where tp_i is the number of true positives of class i, fn_i is the number of false negatives of class i and gt_i is the total number of ground-truth elements of class i. The mean class

Table S2. **Datasets Characteristics**

	KITTI [3, 2]	NYU V2 [6]	S3DIS [1]	Semantic3D.net [4]
Scene	outdoor	indoor	indoor	outdoor
Point Cloud type	partial	partial	full	partial
Sensor type	Laser	Kinect	MatterPort	Laser
Number of training points	12million	125million	228million	1.9billion

Table S3. **Effect of CRF initialization:** End-to-end training vs Manual

Dataset	End-to-end		manual	
	mIOU	mAcc	mIOU	mAcc
Semantic3D.net	**61.30**	**73.08**	60.72	69.69
S3DIS	**48.92**	**57.35**	47.09	53.6
KITTI	**36.78**	**49.46**	36.34	46.34
NYUV2	**43.45**	**56.43**	41.63	52.28

accuracy is then defined as:

$$mAcc = \frac{1}{N}\sum_{i=1}^{N} acc_i, \tag{2}$$

where N is the number of classes.

We define per class IOU following the Pascal convention as:

$$IOU_i = \frac{tp_i}{gt_i + fp_i} = \frac{tp_i}{tp_i + fn_i + fp_i}, \tag{3}$$

where tp_i, gt_i, fn_i are defined as above, and fp_i is the number of false positives of class i. Note that IOU is a more difficult metric than accuracy since it doesn't simply reward true positives, but also penalizes false positives. From the definition above, we obtain mean class IOU as:

$$mIOU = \frac{1}{N}\sum_{i=1}^{N} IOU_i. \tag{4}$$

S3.3. Visualizations

In this section, we include more qualitative segmentation results for all datasets. The results showcase the initial segmentation of the standalone 3D-FCNN-TI followed by the final result of the SEGCloud framework.

References

[1] I. Armeni, O. Sener, A. Zamir, H. Jiang, and S. Savarese. 3D Semantic Parsing of Large-Scale Indoor Spaces. *CVPR*, pages 1534–1543, 2016. S1, S2, S3

[2] A. Geiger. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 3354–3361, Washington, DC, USA, 2012. IEEE Computer Society. S2, S3

[3] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research*, 32(11):1231 – 1237, Sept. 2013. S2, S3

[4] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys. SEMANTIC3D.NET: A NEW LARGE-SCALE POINT CLOUD CLASSIFICATION BENCHMARK. *to appear in ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci 2017*, 2017. S1, S2, S3

[5] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*. 2011. S2

[6] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. S2, S3

[7] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr. Conditional Random Fields as Recurrent Neural Networks. *International Conference on Computer Vision*, pages 1529–1537, 2015. S2

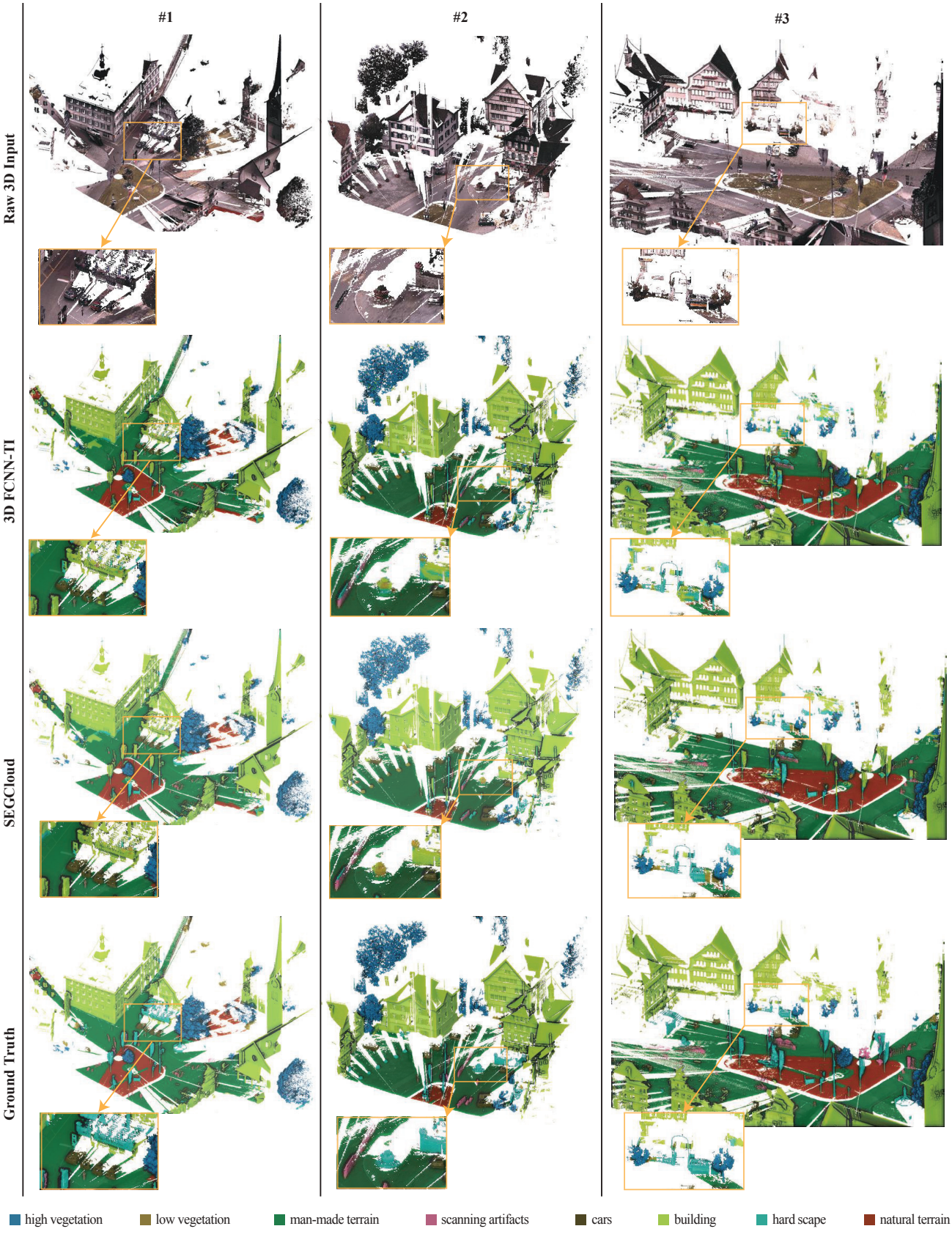

Figure S2. **Qualitative results on the Semantic3D.net dataset**

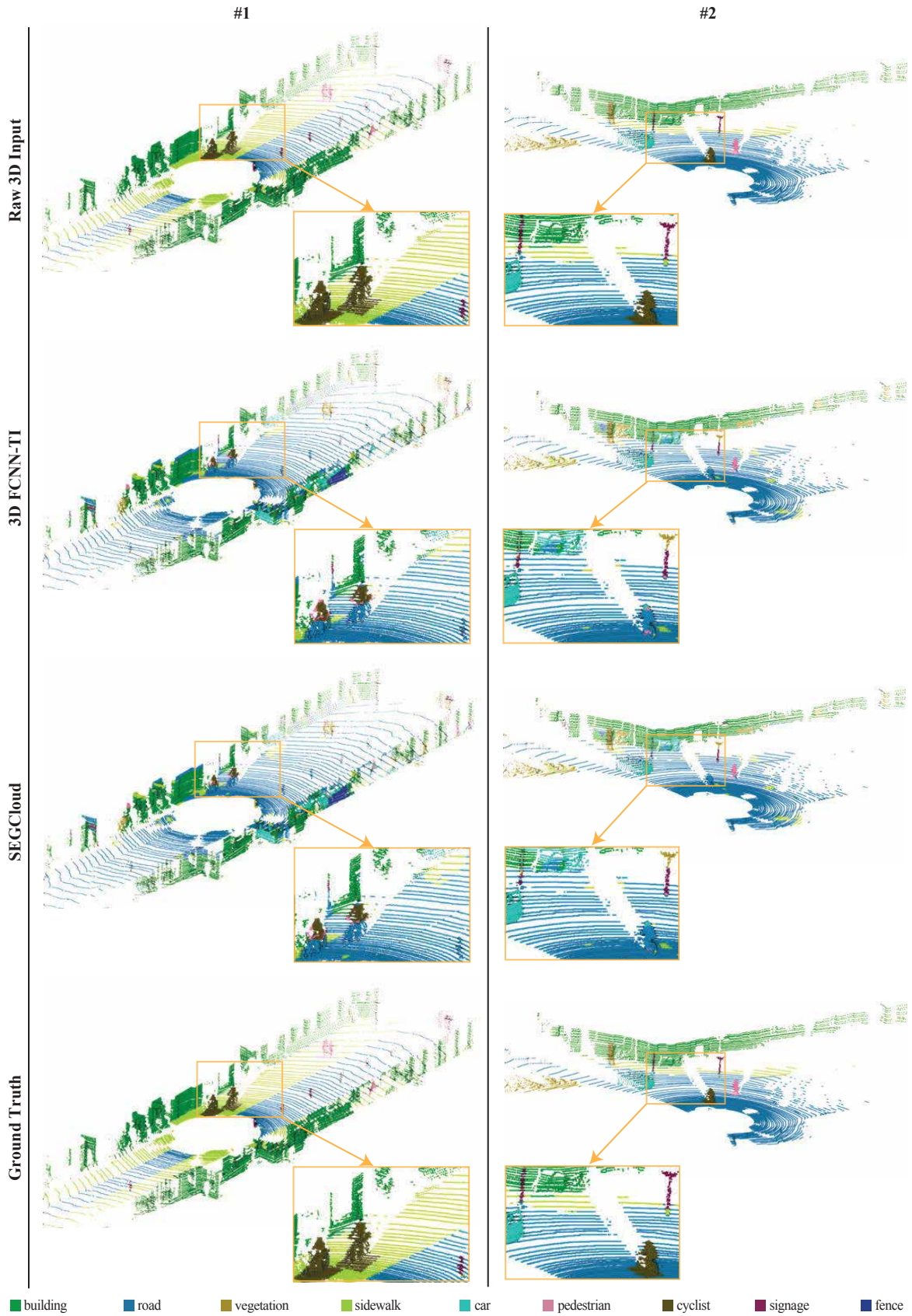

Figure S3. **Qualitative results on the KITTI dataset**

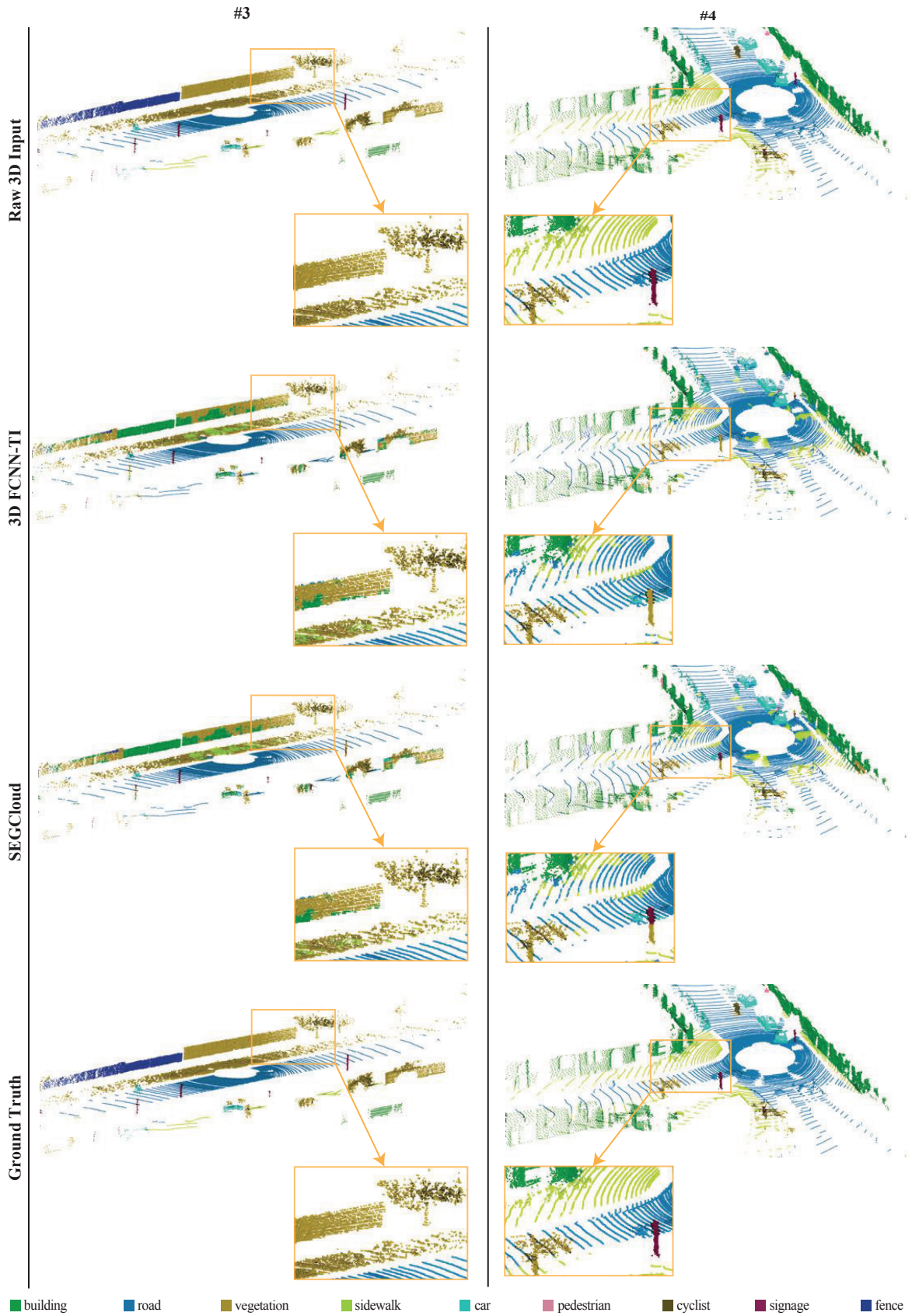

Figure S4. **Qualitative results on the KITTI dataset**

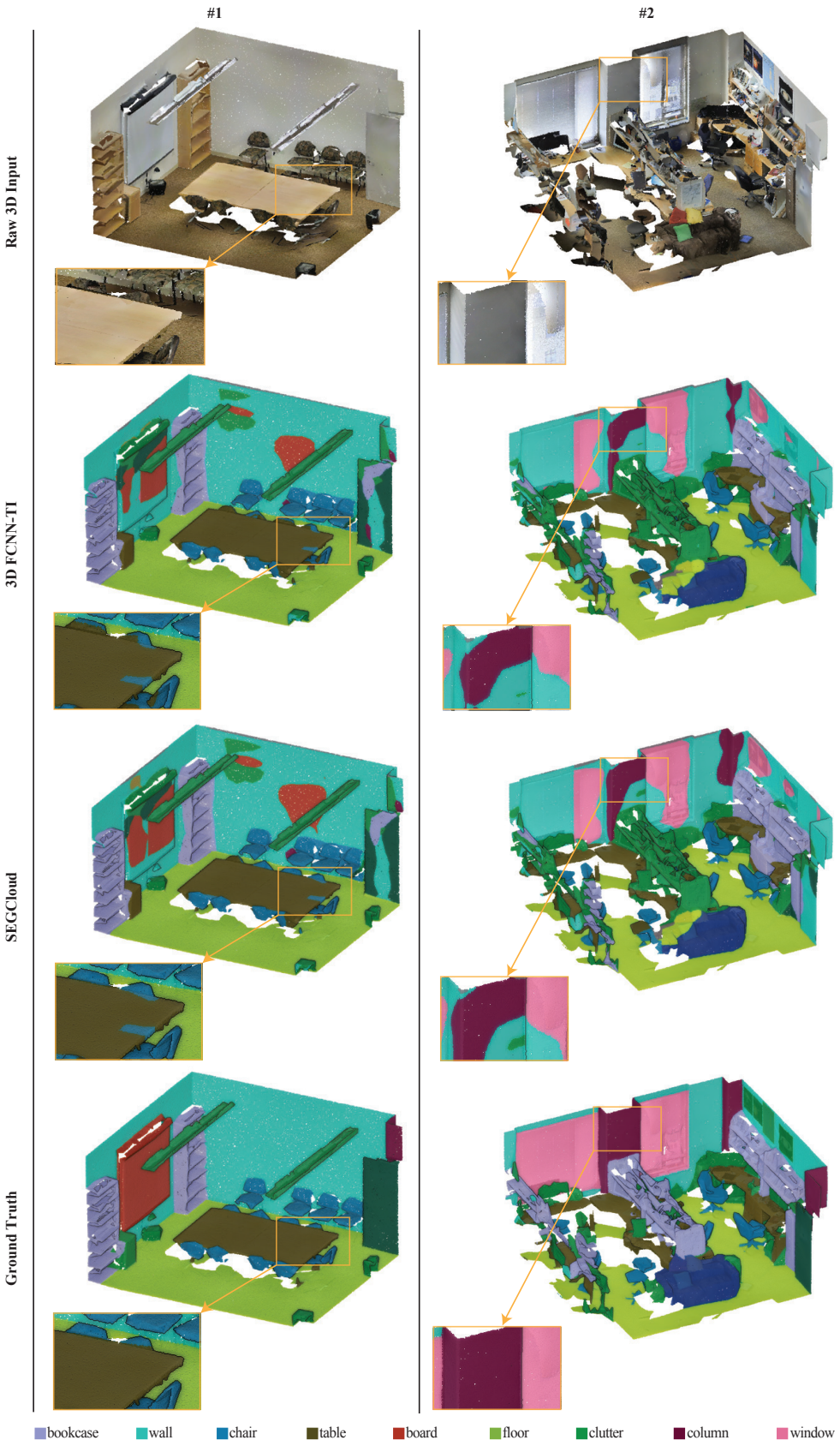

Figure S5. **Qualitative results on the S3DIS dataset**

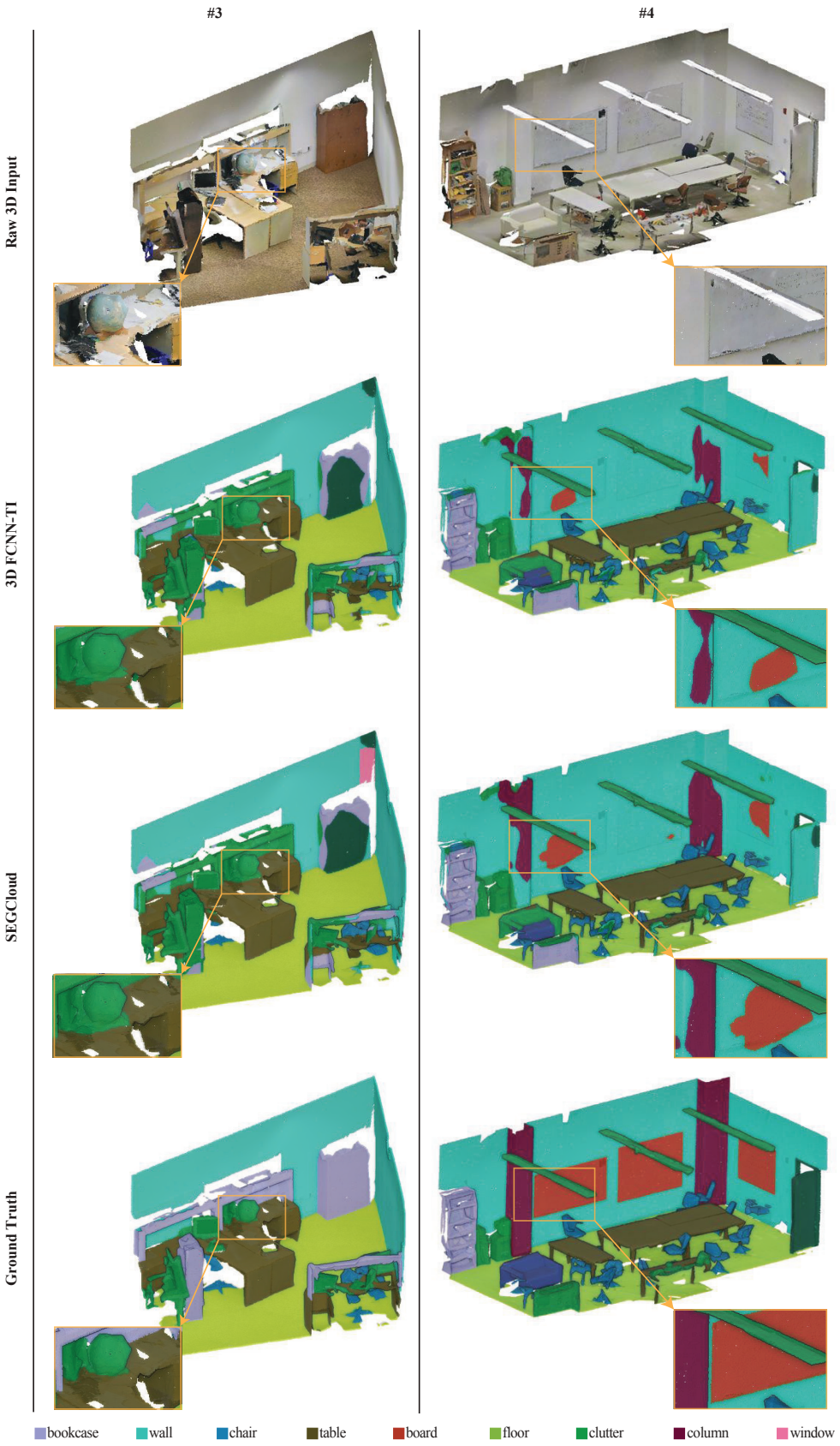

Figure S6. **Qualitative results on the S3DIS dataset**

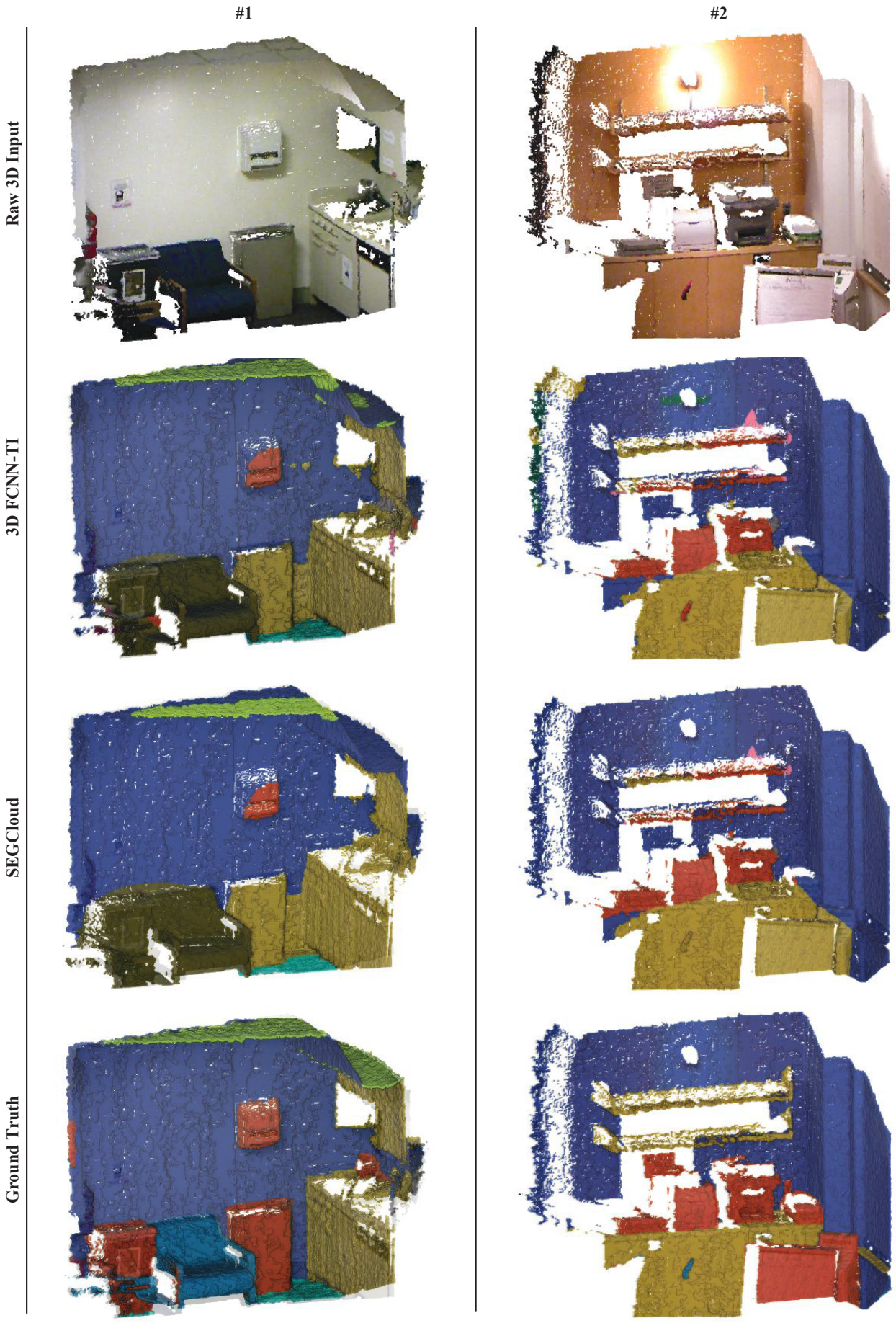

Figure S7. **Qualitative results on the NYU V2 dataset**

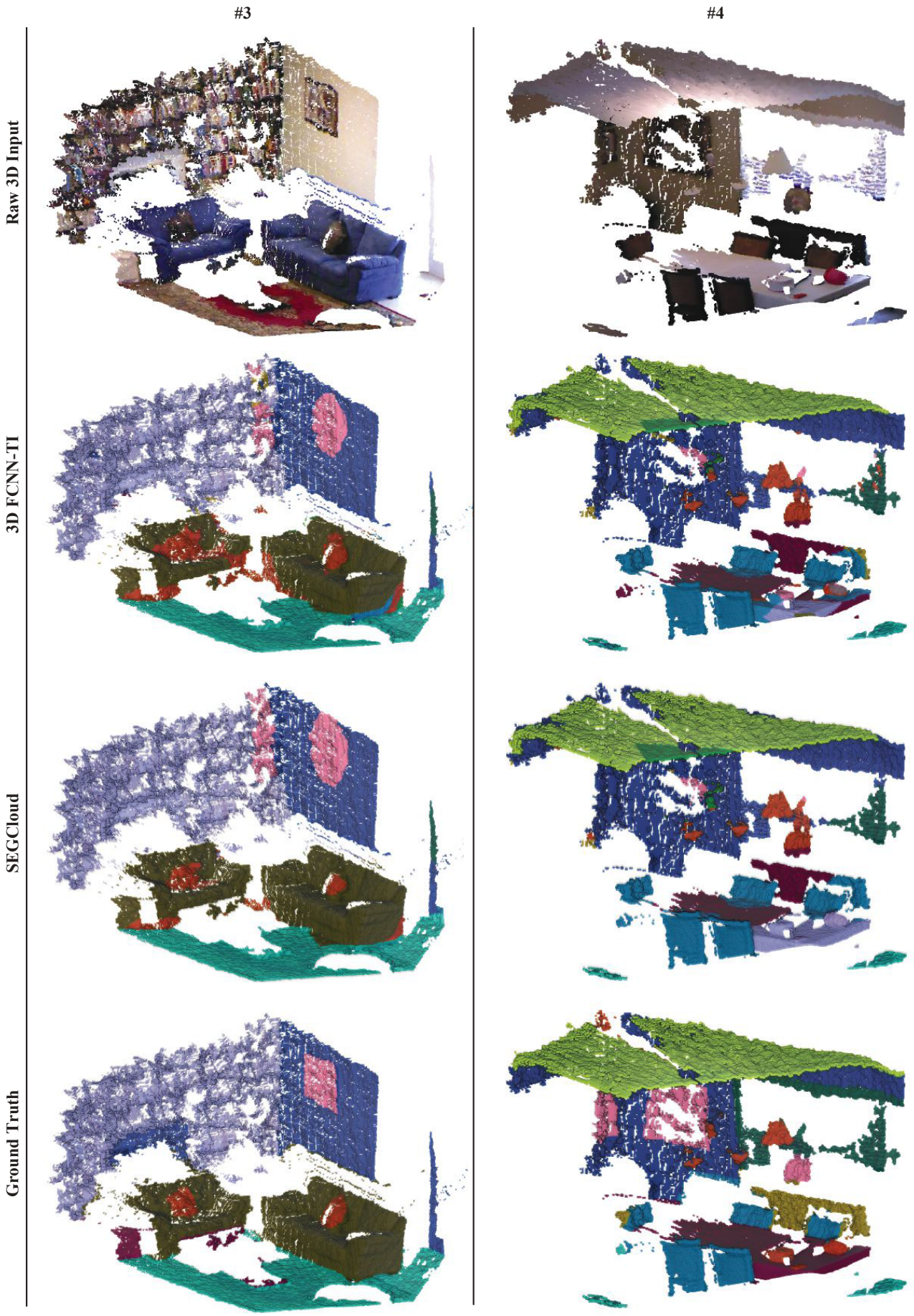

Figure S8. **Qualitative results on the NYU V2 dataset**