DINO in the Room: Leveraging 2D Foundation Models for 3D Segmentation

Supplementary Material

1. Implementation Details

Our implementation builds upon PTv3 [10]. As such, we use the same hyperparameter settings wherever possible. However, unlike PTv3, DITR also utilizes images, for which we add image-specific augmentations. Below, we provide detailed implementation settings for each dataset.

Indoor Semantic Segmentation. Following PTv3, we train the model with a batch size of 12 for 800 epochs on ScanNet [4] and ScanNet200 [7], and for 3000 epochs on the smaller S3DIS [1] dataset. For ScanNet++ [11], we again follow PTv3, dividing the large point clouds into overlapping $6\,\mathrm{m}$ by $6\,\mathrm{m}$ tiles with a stride of $3\,\mathrm{m}$, and training for 400 epochs. Optimization is performed using the AdamW [5] optimizer, with a 1cycle [8] learning rate schedule. The maximum learning rate is set to 6×10^{-3} with the learning rate peaking at 5 % of the training process. Point cloud augmentations include random rotation, random dropout of points, random scaling, random flipping, elastic distortion, and color jittering. Grid sampling is applied with a grid size of $2\,\mathrm{cm}$.

Before being fed into DINOv2 [6], ScanNet images are resized to 420×560 pixels while S3DIS images are resized to 518×518 pixels, preserving the aspect ratio and ensuring a comparable number of image tokens to those used during DINOv2 training. Images undergo augmentation with random horizontal flipping and color jittering. During training, 10 images are randomly selected per 3D scene, while during validation, 10 temporally equidistant images are chosen for consistent evaluation.

We only inject features for points that are visible in the selected images. To prevent injecting features for occluded points, we compare the depth of each projected point with the raw depth value from the RGB-D sensor and exclude points where the depth difference exceeds a small margin of error ($20\,\mathrm{cm}$). Additionally, to ensure that image features are consistently assigned to points at similar distances, points that are closer than $1\,\mathrm{m}$ or farther than $4\,\mathrm{m}$ are filtered out. Given that the 3D scene is captured from multiple camera angles with overlapping regions, most points fall in this range in at least one image.

When distilling DINOv2 [6] features into the 3D model, the ScanNet [4] and Structured3D [12] datasets are used for joint training. Each batch consists of samples from only one dataset, with half of the batches derived from ScanNet and the other half from Structured3D. We train the model for a total of 160k steps with a batch size of 12. After distillation, the pretrained model is fine-tuned separately on the ScanNet, ScanNet200, and S3DIS datasets. For ScanNet and S3DIS, the model is fine-tuned for 10 % of the default number of epochs, and, except for the segmentation head, the learning rate is set to 10 % of its original value. For ScanNet200, fine-tuning is performed using the default training setup described earlier. This approach reflects differences in dataset complexity: ScanNet and S3DIS, with fewer classes, achieve strong performance with minimal fine-tuning, while ScanNet200, which has a larger and more challenging class set, benefits from longer training to fully utilize the model's capacity.

Outdoor Semantic Segmentation. Following PTv3, the model is trained with a batch size of 12 for 50 epochs on nuScenes [3], SemanticKITTI [2] and Waymo [9]. Again, optimization is performed using the AdamW optimizer, with a 1cycle learning rate schedule. The maximum learning rate is set to 2×10^{-3} with the learning rate peaking at 4 % of the training process. Point cloud augmentations include random rotation, random scaling, and random flipping. Grid sampling is performed with a grid size of $5\,\mathrm{cm}$.

To obtain DINOv2 features, nuScenes [3] images are resized to 378×672 pixels, SemanticKITTI [2] to 378×1246, and Waymo [9] to 308×672. This keeps the original aspect ratios while also yielding a number of image tokens similar to what was used during DINOv2's training. We make an exception for the wide-aspect-ratio images in SemanticKITTI, using a higher resolution to avoid excessively reducing the vertical resolution. As with the indoor datasets, images undergo random horizontal flipping and color jittering augmentations.

For distillation, we jointly train the model on the nuScenes, SemanticKITTI, and Waymo datasets. Each batch consists of samples of one dataset and batches are equally distributed across the datasets. We train the model for a total of 350k steps with a batch size of 12. After distillation, the pretrained model is fine-tuned separately on nuScenes, SemanticKITTI, and Waymo. For the SemanticKITTI dataset, the model is fine-tuned for 10 % of the default number of epochs and, except for the segmentation head, the learning rate is set to 10 % of its original value. For nuScenes and Waymo, fine-tuning is performed using the default training setup described earlier.

2. Impact of Image Selection Strategy

In Tab. 1, we analyze the impact of varying the number of images and the image selection strategy during inference while using the same trained model. For ScanNet200, we

Image selection		ScanNet200			nuScenes	
# Images	Method	$mIoU_6$	$mIoU_{10}$	% Vis.	mIoU	% Vis.
0	eq.dist.	32.8	31.8	0.0	37.5	0.0
1	eq.dist.	36.5	35.7	9.8	50.9	11.2
3	eq.dist.	38.2	38.0	24.6	72.7	40.5
6	eq.dist.	39.2	40.1	41.7	83.1	77.0
10	eq.dist.	39.9	41.2	56.7	—	—
6	random	38.6	38.4	37.0	—	—
10	random	39.2	40.5	49.5	—	—

Table 1. **Impact of image selection strategy during inference.** We compare two inference-time image selection methods: random selection and temporally-equidistant frame selection (eq.dist.). For ScanNet200, we compare models trained on 6 ($mIoU_6$) or 10 random images ($mIoU_{10}$). For nuScenes, the model is trained using all six camera views. During inference, we use different subsets of cameras: no camera; the front camera alone; the front, rear-left, and rear-right cameras; all six cameras.

evaluate two models, one trained with 6 images per scene ($mIoU_6$) and another with 10 ($mIoU_{10}$) to evaluate the impact of the number of training images. For nuScenes, we only evaluate a model trained on all available images. We find that, on ScanNet200, the model trained with 6 images benefits from seeing more images during inference than during training, and performs better than the one trained with 10 images when fewer images are used during inference. Furthermore, both models demonstrate robustness to seeing fewer images during inference, with performance degrading gracefully as they are trained with randomly selected images. However, on nuScenes, the model always receives the full surrounding visual context during training. As a result, the performance degrades significantly when frames are missing during inference. Finally, on ScanNet200, we also show the effectiveness of using temporally-equidistant images during inference, resulting in higher point coverage and consistently outperforming the alternative of selecting frames randomly.

3. Qualitative Results

Distillation. We apply Principal Component Analysis (PCA) to the point features generated by the distilled D-DITR model and visualize the first three principal components as RGB colors. This provides a qualitative view of the semantic information captured during distillation pretraining. Note that this model operates without image input during inference.

Figs. 1 to 5 highlight the rich semantic features distilled from DINOv2. Points belonging to the same semantic class exhibit similar colors, reflecting alignment along the first three principal axes. For instance, in Figs. 1 and 2, objects such as tables, chairs, books, and walls consistently map to distinct colors across the scene. These results show that the model captures fine-grained semantic representations of 3D scenes without requiring semantic labels during training. Furthermore, Figs. 1 and 2 reveal that objects like chairs and tables are segmented into finer subparts in feature space, surpassing the granularity of the datasets' annotations, which this model does not use. This semantically rich feature representation significantly enhances the model's performance when used as initialization before fine-tuning for semantic segmentation, as is evident by the quantitative results in the main paper.

Semantic Segmentation. Figs. 6 to 8 compare the predictions of DITR and PTv3, with a focus on points where DITR benefits from DINOv2 features.

In Fig. 6, we observe that PTv3 misclassifies a 'cart' as a 'shelf' due to the structural similarities between these classes in the 3D point cloud. In contrast, DITR accurately identifies the carts by incorporating visual cues from 2D images via DINOv2 feature injection.

In Fig. 7 we show another perspective of the same scene. Here, PTv3 misclassifies a 'cabinet' as a 'wall', as the cabinet appears as a flat surface that is flush with the wall and has a similar color, making it almost impossible to identify it as a cabinet based on the point cloud only. However, DITR correctly classifies the cabinet, as it is clearly visible in one of the image views. This highlights how leveraging strong image features can complement and enhance geometric features, enabling the model to differentiate between structurally similar classes.

Finally, in Fig. 8, we present an example from the nuScenes dataset. PTv3 misclassifies a pedestrian crossing the street as a car, overlooks several pedestrians on the sidewalk, and confuses a bus with a building. These errors occur in areas distant from the ego vehicle, where LiDAR data is sparse, and objects are represented by only a few points. In contrast, DITR accurately predicts these instances, leveraging contextual information from the respective image for disambiguation.

References

[1] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3D Semantic Parsing of Large-Scale Indoor Spaces. In *CVPR*, 2016. 1

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *ICCV*, 2019. 1, 6

[3] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A Multimodal Dataset for Autonomous Driving. In *CVPR*, 2020. 1, 7, 8, 11

[4] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet:

Richly-annotated 3D Reconstructions of Indoor Scenes. In *CVPR*, 2017. 1, 4, 5, 9, 10

[5] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019. 1

[6] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, et al. DINOv2: Learning Robust Visual Features without Supervision. *TMLR*, 2024. 1

[7] David Rozenberszki, Or Litany, and Angela Dai. Language-Grounded Indoor 3D Semantic Segmentation in the Wild. In *ECCV*, 2022. 1

[8] Leslie N. Smith and Nicholay Topin. Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, 2019. 1

[9] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in Perception for Autonomous Driving: Waymo Open Dataset. In *CVPR*, 2020. 1

[10] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point Transformer V3: Simpler, Faster, Stronger. In *CVPR*, 2024. 1, 9, 10, 11

[11] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, 2023. 1

[12] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3D: A Large Photo-realistic Dataset for Structured 3D Modeling. In *ECCV*, 2020. 1

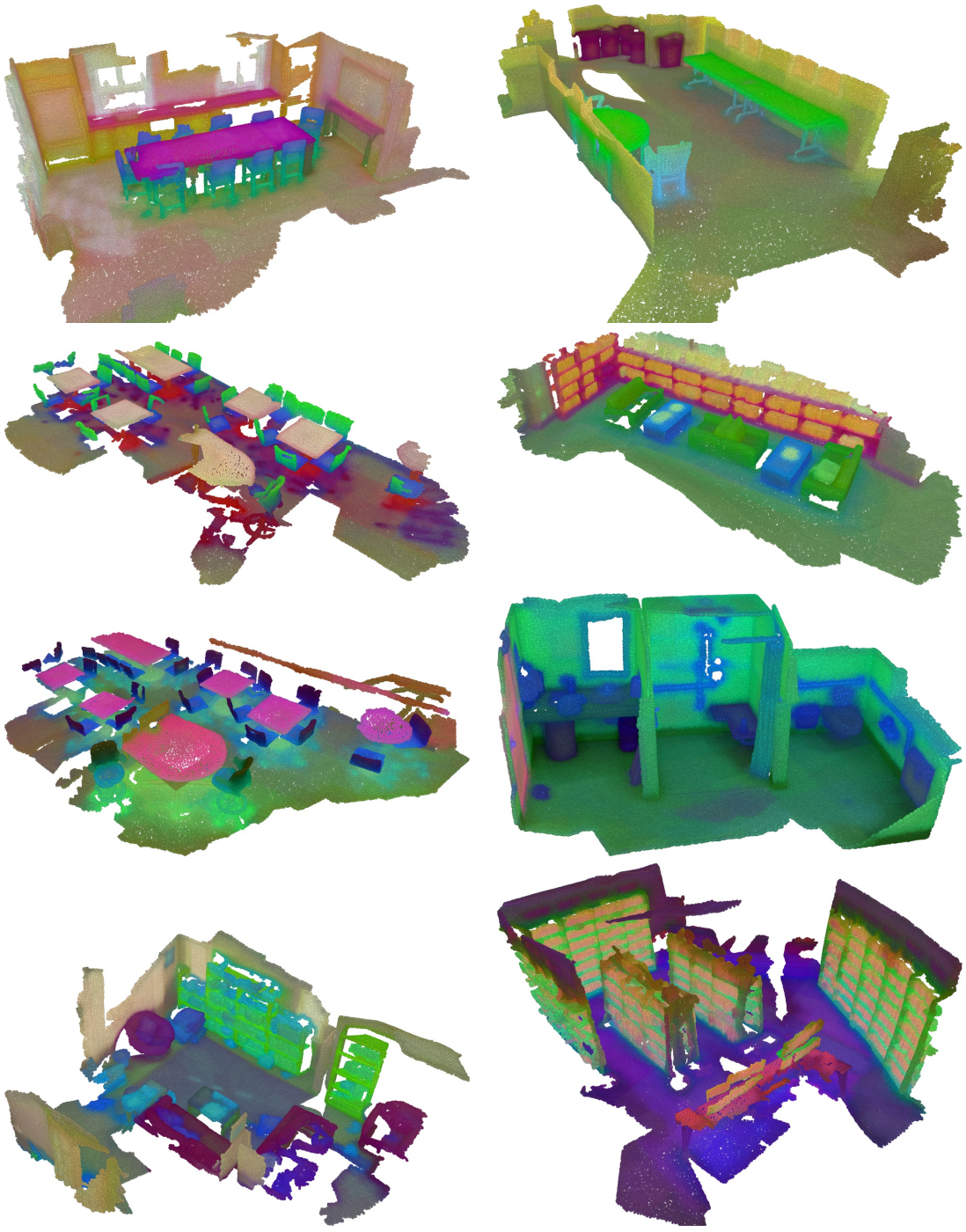


Figure 1. **PCA visualization of distilled D-DITR features for ScanNet [4].** We visualize the first three principal components of the point features learned by the D-DITR model. PCA projections are computed per scene and hence there is no correspondence of colors between figures.

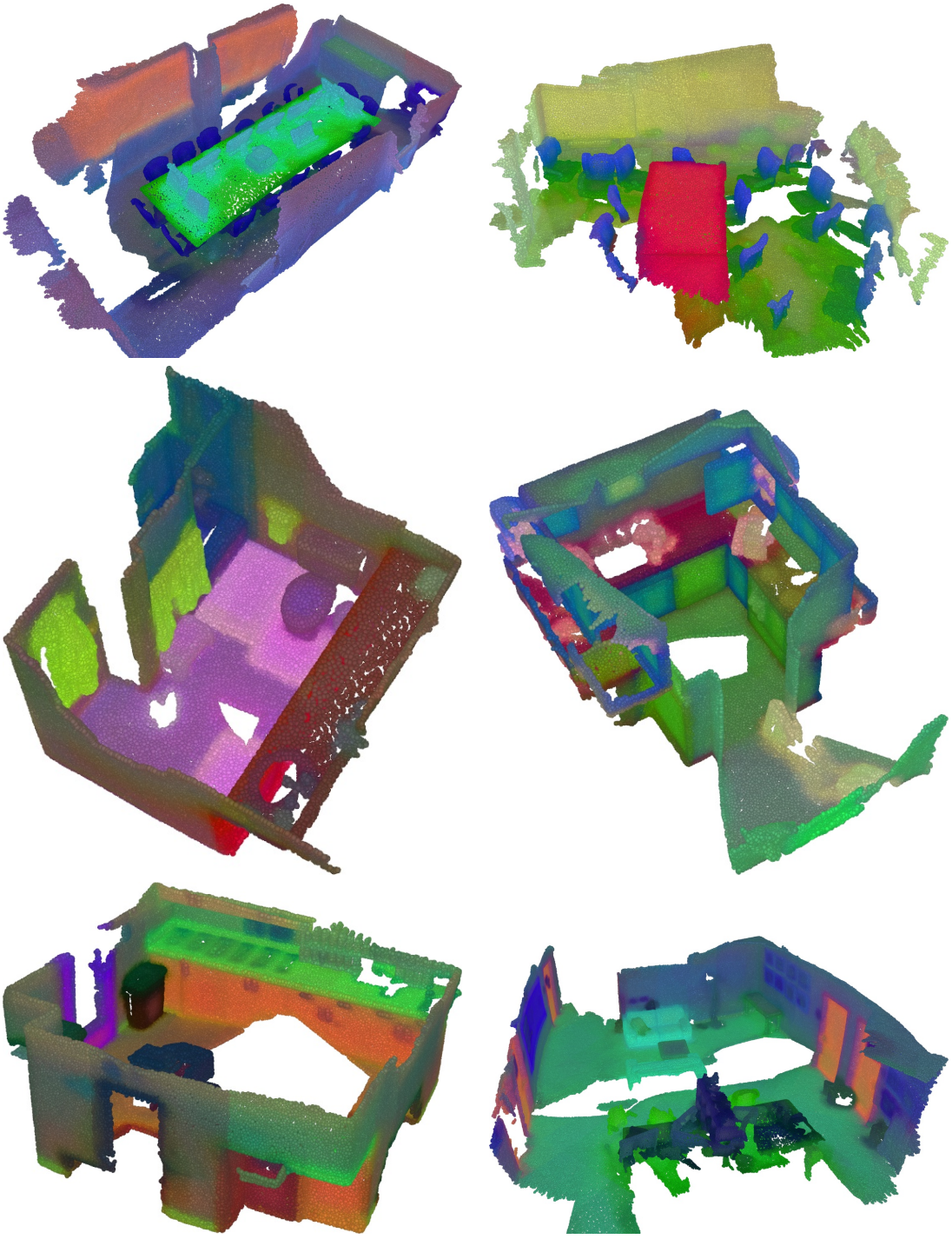


Figure 2. **(Cont.) PCA visualization of distilled D-DITR features for ScanNet [4].** We visualize the first three principal components of the point features learned by the D-DITR model. PCA projections are computed per scene and hence there is no correspondence of colors between figures.

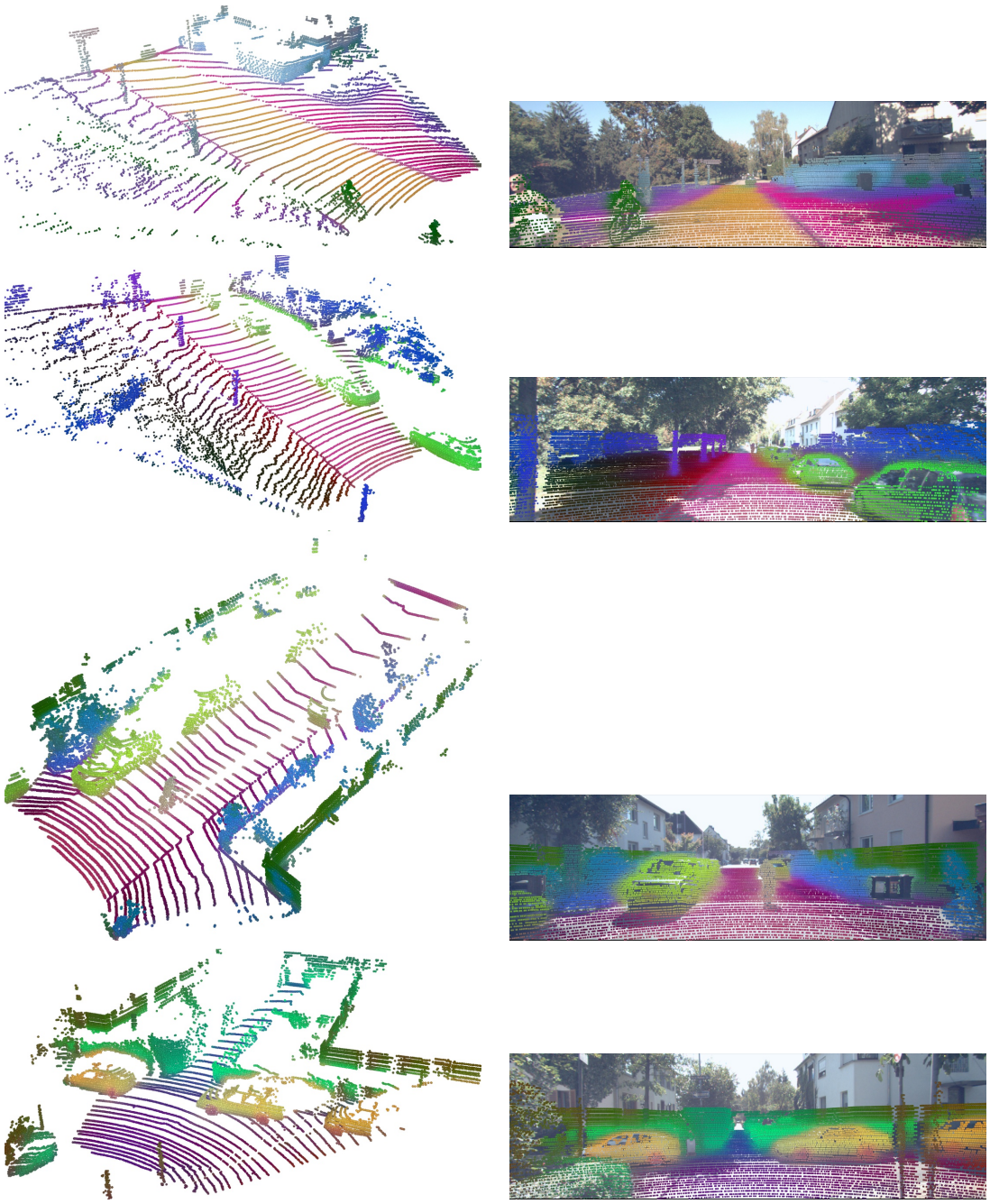


Figure 3. **PCA visualization of distilled D-DITR features for SemanticKITTI [2].** We visualize the first three principal components of the point features learned by the D-DITR model (left), as well as the corresponding 2D projections (right).

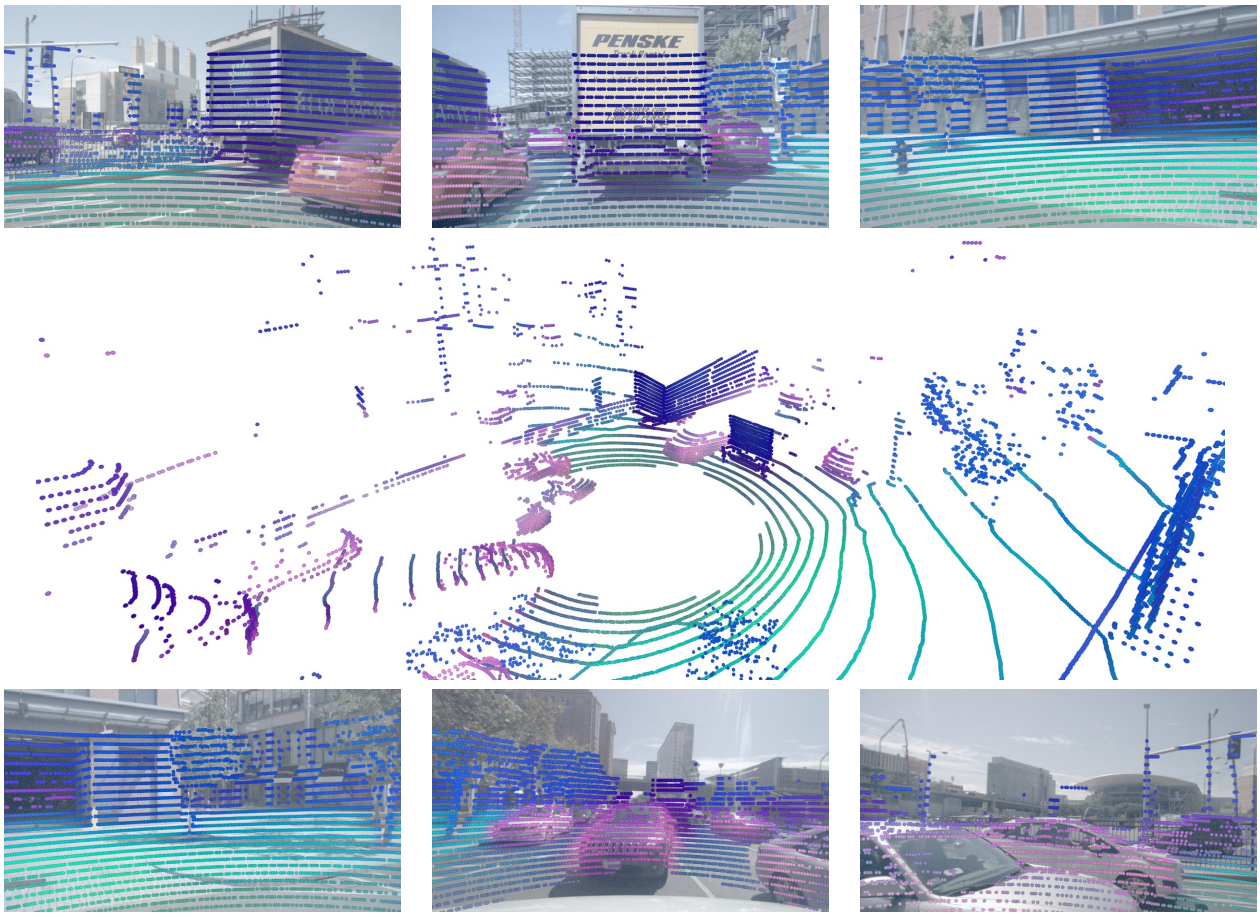


Figure 4. **PCA visualization of distilled D-DITR features for nuScenes [3].** We visualize the first three principal components of the point features learned by the D-DITR model.

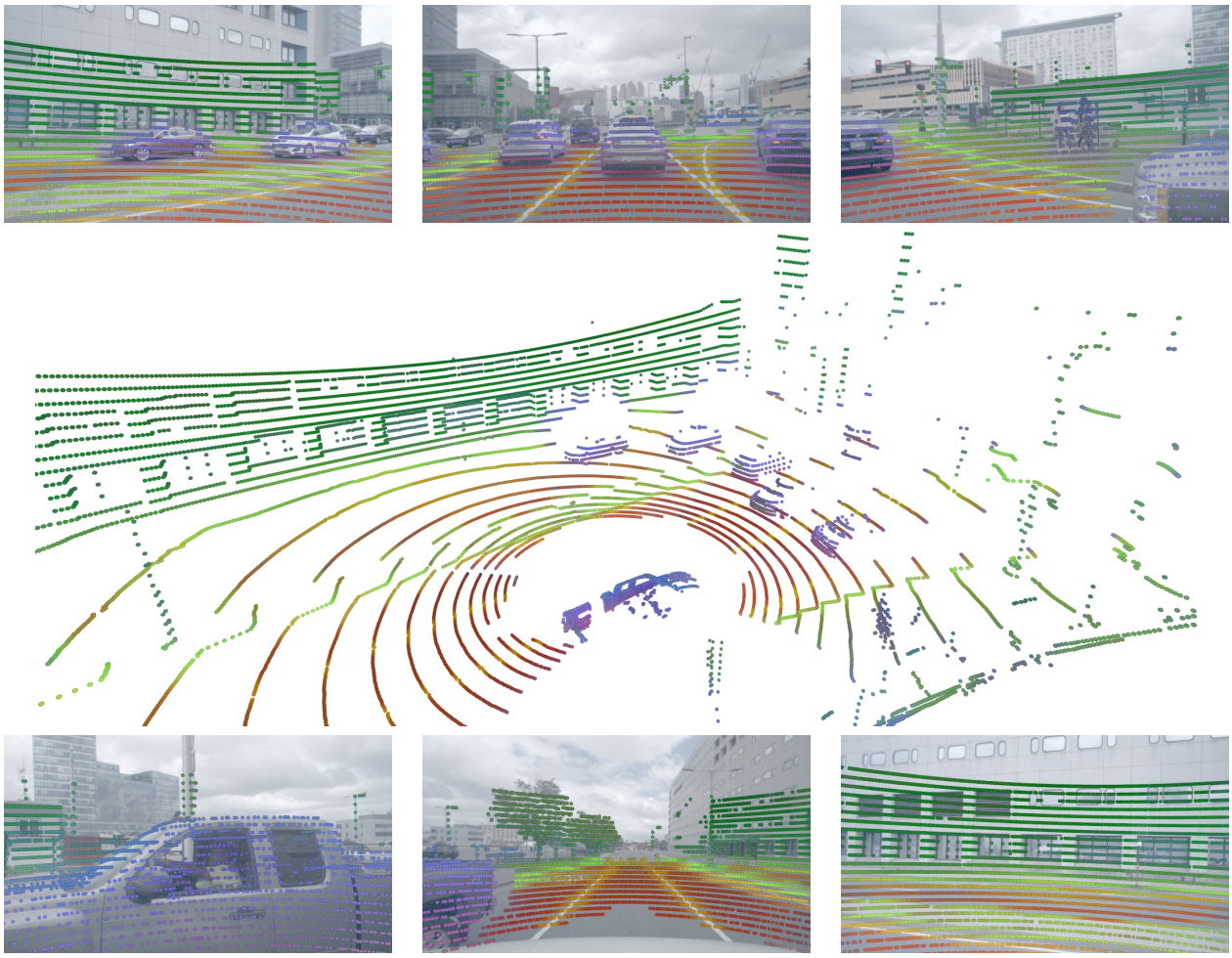
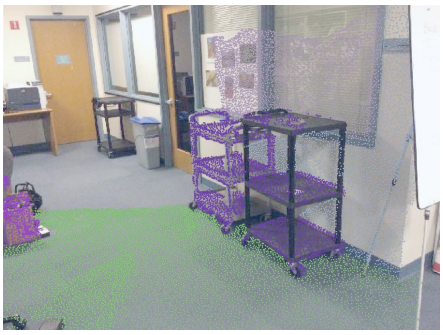
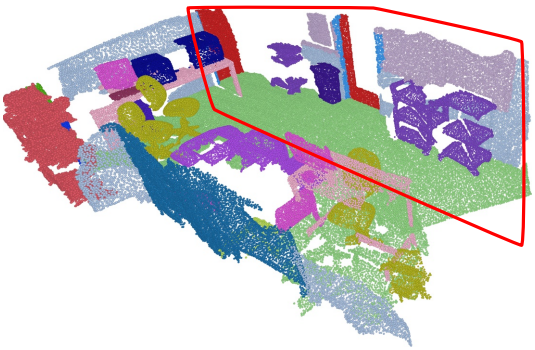


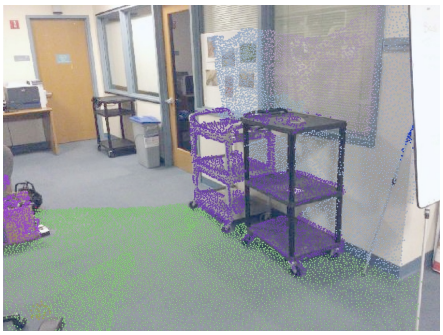
Figure 5. **PCA visualization of distilled D-DITR features for nuScenes [3].** We visualize the first three principal components of the point features learned by the D-DITR model.



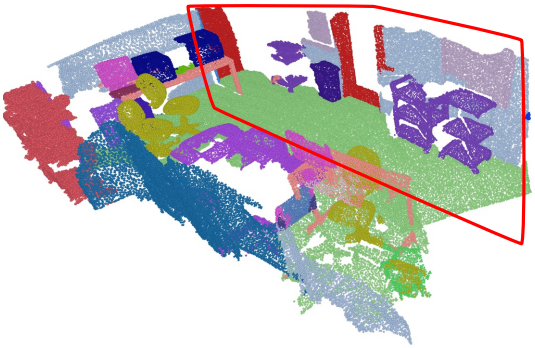
2D projected ground truth



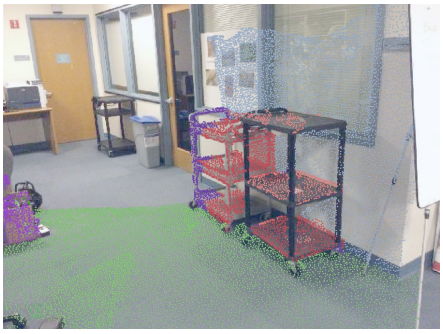
Full 3D ground truth



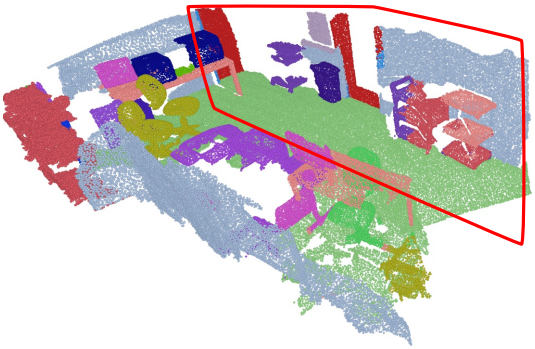
2D projected DITR predictions



Full 3D DITR predictions

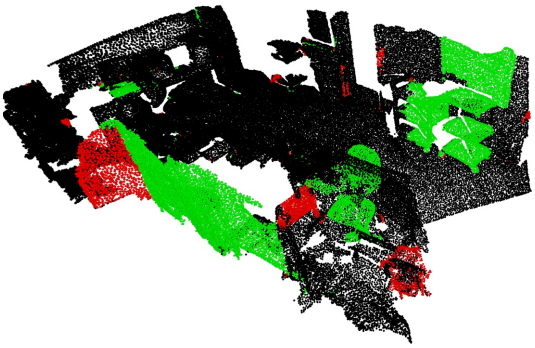


2D projected PTv3 predictions



Full 3D PTv3 predictions

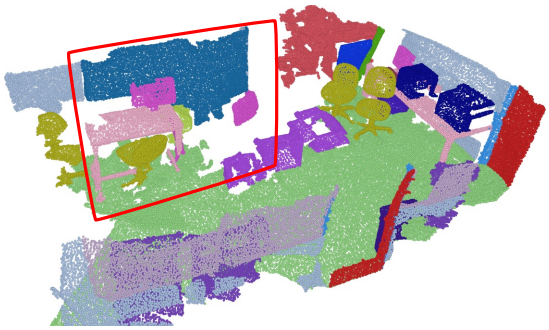
3D point cloud input



Difference between PTv3 and DITR predictions

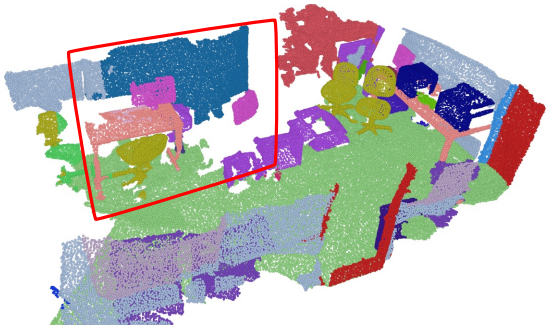
Figure 6. **DITR and PTv3 [10] segmentation for ScanNet [4].** The wheels of the cart are barely visible in the 3D point cloud, likely causing PTv3 to misclassify it as a shelf. By contrast, DITR can leverage semantically rich image features to correctly identify this cart. The bottom-right visualization illustrates the differences between PTv3 and DITR predictions. Points where both methods are correct or both are incorrect are shown in black. Points correctly predicted by only one method are color-coded: green for DITR and red for PTv3.

2D projected ground truth



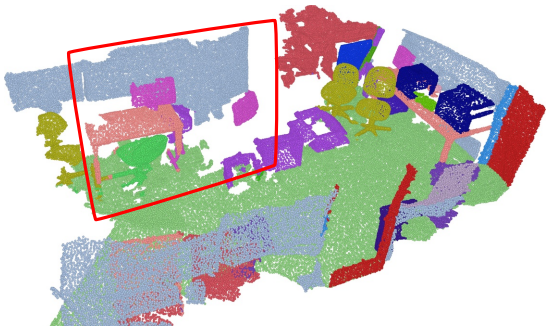
Full 3D ground truth

2D projected DITR predictions



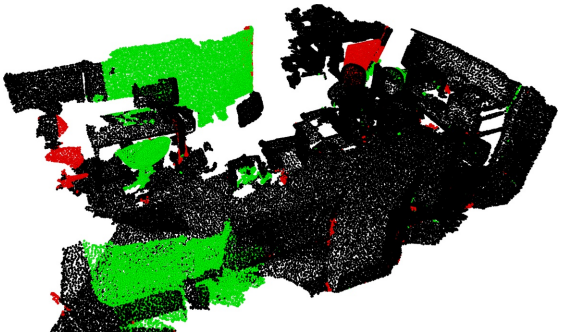
Full 3D DITR predictions

2D projected PTv3 predictions



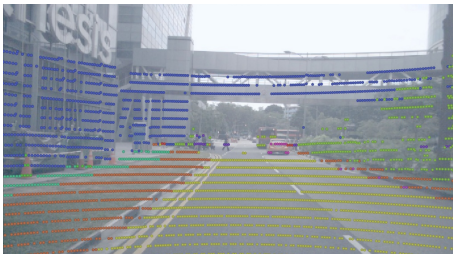
Full 3D PTv3 predictions

3D point cloud input

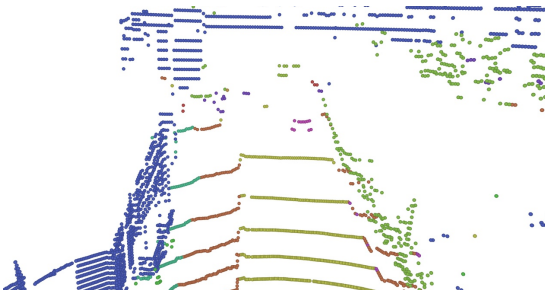


Difference between PTv3 and DITR predictions

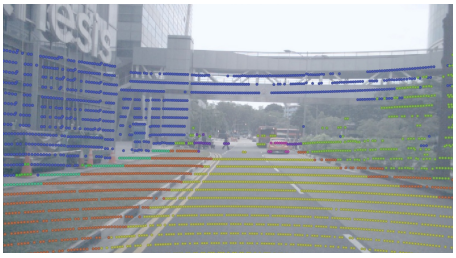
Figure 7. **DITR and PTv3 [10] semantic segmentation for ScanNet [4].** In the 3D point cloud, the cabinet at the back appears as a flat surface flush with the wall, leading PTv3 to misclassify it as part of the wall. However, in the corresponding image, this region is clearly identifiable as a cabinet. The bottom-right visualization illustrates the differences between PTv3 and DITR predictions. Points where both methods are correct or both are incorrect are shown in black. Points correctly predicted by only one method are color-coded: green for DITR and red for PTv3.



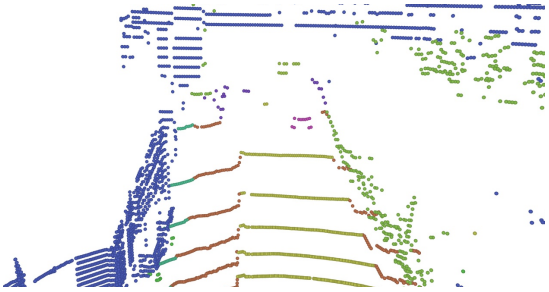
2D projected ground truth



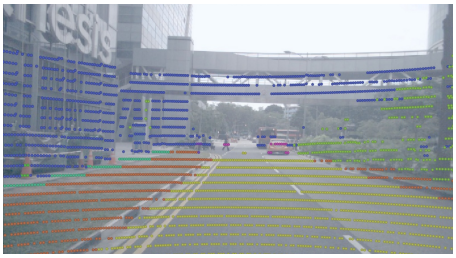
Front view 3D ground truth



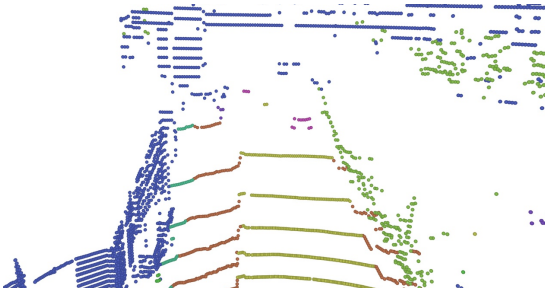
2D projected DITR predictions



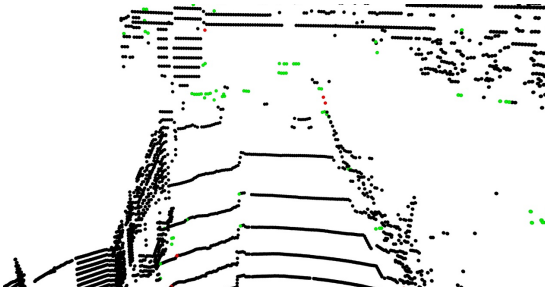
Front view 3D DITR predictions



2D projected PTv3 predictions



Front view 3D PTv3 predictions



Difference between PTv3 and DITR predictions

Figure 8. **DITR and PTv3 [10] semantic segmentation for nuScenes [3].** PTv3 misclassifies the pedestrian crossing the street as a car, overlooks the pedestrians on the left sidewalk and the pedestrian on the right, and mistakes the bus in front of the car for a building. These errors occur in areas distant from the ego vehicle, where LiDAR data is sparse, and objects are represented by only a few points. In contrast, DITR can leverage the corresponding image, where the bus and pedestrian on the street are clearly visible, allowing it to make the correct predictions despite the objects being represented by only a few points. The bottom-right visualization illustrates the differences between PTv3 and DITR predictions. Points where both methods are correct or both are incorrect are shown in black. Points correctly predicted by only one method are color-coded: green for DITR and red for PTv3.