Three Cars Approaching within 100m! Enhancing Distant Geometry by Tri-Axis Voxel Scanning for Camera-based Semantic Scene Completion

Supplementary Material

A. Dataset and Metric

Dataset. We evaluate ScanSSC on SemanticKITTI [1] and SSCBench-KITTI-360 [10] datasets. SemanticKITTI is derived from the KITTI Odometry [3] benchmark, consisting of 22 outdoor scenes captured by LiDAR scans and stereo images. These 22 scenes are split into 10 training scenes, 1 validation scene, and 11 test scenes. The ground truth voxel grids have dimensions of 256×256×32, with each voxel measuring (0.2m, 0.2m, 0.2m), annotated with 21 semantic classes (19 semantics, 1 empty and 1 unknown). SSCBench-KITTI-360 is extracted from the KITTI-360 [11], comprising 7 training scenes, 1 validation scene, and 1 test scene. It includes 19 semantic classes (18 semantics and 1 free).

Metric. Following standard practices in related works [6, 7, 9, 17], we use the mean Intersection over Union (mIoU) to assess the overall performance of semantic scene completion (SSC) and Intersection over Union (IoU) to evaluate the performance of semantic-agnostic scene completion.

B. More Details

Implementation Details. We train ScanSSC for 25 epochs on 4 NVIDIA A6000 GPUs with a batch size of 4. The AdamW optimizer [12] is used with $\beta_1 = 0.9$, $\beta_2 = 0.99$, and a maximum learning rate of 3×10^{-4}. For the learning rate schedule, we employ a multi-step scheduler, reducing the learning rate by a factor of 0.1 at the 20^{th} epoch.

Architectural Details. Similar to related works [2, 5, 16, 17], we employ a 2D UNet image encoder built upon a pretrained EfficientNetB7 [15]. Following previous stereo-based methods [6, 9, 17], we utilize the MobileStereoNet [14] as the depth estimator. In the viewing transformation, we adopt the depth network from CGFormer [17], which modifies the BEVDepth [8]. There are 3 deformable attention layers for cross-attention and 2 for self-attention, with 8 sampling points per reference point in both heads. The spatial mixing network consists of 3 stages, each with 2 residual blocks [4].

C. Computational Cost

We report the computational cost of ScanSSC compared to CGFormer [17] in Tab. C.1. ScanSSC shows competitive efficiency, with only a slight increase in parameters and inference time. However, it achieves notable performance improvements, with a 0.13 increase in IoU and a 0.77 increase in mIoU on the SemanticKITTI test set, highlighting the effectiveness of our method.

Method	Params (M)	Inference Time (ms)	IoU	mIoU
CGFormer	122	566	44.41	16.63
ScanSSC	145	674	**44.54**	**17.40**

Table C.1. Comparison of computational costs with CGFormer [17]. The inference time for a single sample of SemanticKITTI [1] validation set is measured on 1 NVIDIA A6000 GPU.

D. Additional Ablation Studies

We provide additional ablation studies to evaluate the effectiveness of the subcomponents of ScanSSC. Consistent with the manuscript, all experiments are conducted on the SemanticKITTI [1] validation set.

Ablation Study of Tri-Feature Fusion Methods. We perform an ablation study to demonstrate the validity of ScanSSC's tri-feature fusion method by replacing it with three alternative methods, one at a time (Tab. D.1). 'Concat→Linear' denotes the concatenation of the three axis-specific features along the channel dimension, followed by a linear layer to directly compute the output feature. 'Average' refers to an element-wise average of the three features, while 'Weighted Sum' denotes a weighted summation of the three features using voxel-wise learnable parameters $L \in \mathbb{R}^{\hat{X} \times \hat{Y} \times \hat{Z} \times 3}$. We observe that the proposed tri-feature fusion method results in a significantly higher mIoU value than the three alternative methods, demonstrating the effectiveness of voxel-wise adaptive fusion of axis-specific features.

Method	IoU	mIoU
Concat → Linear	45.90	16.32
Average	46.17	16.51
Weighted Sum	45.90	16.28
Tri-Feature Fusion	45.95	**17.12**

Table D.1. Ablation study on the tri-feature fusion method of ScanSSC.

Loss Scaling Coefficient for Scan Loss. We conduct an ablation study on the loss scaling coefficient of the Scan Loss, $\mathcal{L}_{scan}$, as shown in Fig. D.1. When the coefficient is

set to 1, the highest mIoU score of 17.12 is observed, while it decreases as the coefficient moves further from 1 overall. We find that the training mIoU consistently increases proportionally with λ_{scan} throughout the entire training. From this result, we infer that an excessively high value of λ_{scan} can lead to overfitting of the model, highlighting the importance of selecting an appropriate scaling coefficient.

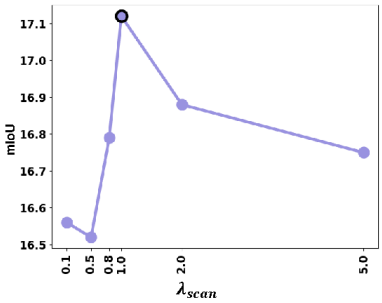


Figure D.1. Performance comparison by loss scaling coefficient for Scan Loss.

Scan Loss *vs*. General Cross-Entropy. Since the proposed Scan Loss is equivalent to the cross-entropy [18] of the cumulatively averaged logit, incorporating it can be seen as analogous to either modifying the distribution of the loss coefficient for the existing voxel-wise cross-entropy or simply increasing its scale. Hence, we compare the performance of ScanSSC when it is substituted with a simple coefficient adjustment for the voxel-wise cross-entropy loss, as shown in Tab. D.2. For (a), to assign higher weights to distant voxels, we first generate a bilinearly interpolated weight along each axis, ranging from 0 to 1 in the corresponding near-to-far direction, then use the average of these weights as the coefficient for the existing cross-entropy loss (λ_{tri}). For (b), we simply apply a higher scalar weight (λ_{ce}) to the voxel-wise cross-entropy loss. Here, we set λ_{ce} to 10, as the converged loss scales of both methods are similar.

Method	IoU	mIoU
(a) $\lambda_{tri}\mathcal{L}_{ce}$	46.15	16.50
(b) $\lambda_{ce}\mathcal{L}_{ce}$	45.04	16.74
$\mathcal{L}_{ce}$ + $\mathcal{L}_{scan}$ (Ours)	45.95	**17.12**

Table D.2. Performance comparison between incorporating Scan Loss and adjusting the coefficient for voxel-wise cross-entropy loss [18]. The other training losses remain unchanged during training.

Comparing with (a), we observe that Scan Loss significantly enhances mIoU by leveraging the semantic distribution of previous voxels to transmit signals to the target voxel. This demonstrates that instead of merely assigning higher weights to distant voxels, utilizing the semantic distribution of previous voxels to propagate signals more effectively is a superior approach. In addition, when compared to (b), the result demonstrates that the simple increase in the weight of the existing cross-entropy does not lead to performance improvement, as it results in significantly lower IoU and mIoU scores.

Ablation Study of Subsidiary Components. We conduct additional experiments to validate the importance of the subsidiary components, the spatial mixing network, and tri-feature fusion. Since tri-feature fusion can not be removed entirely, we replace it with "Concat→Linear" which is represented in Tab. D.1. As shown in Tab. D.3, the spatial mixing network and tri-feature fusion contribute to performance improvement, demonstrating that enhancing regional spatial patterns and adaptively fusing features are effective.

Method	Spatial-mixing net.	Tri-feature fusion	IoU	mIoU
(a)			44.91	15.90
(b)	✓		45.90	16.32
(c)		✓	45.08	16.11
ScanSSC	✓	✓	**45.95**	**17.12**

Table D.3. Ablation study on the subsidiary components of the ScanSSC.

Ablation Study of Using Tri-Axes Features. To demonstrate the validity of using features from all three axes, we visualize and compare the results obtained using features from each axis with those obtained through ScanSSC. The results obtained using each axis individually are represented as 'Depth Only,' 'Width Only,' and 'Height Only,' and are shown in Fig. D.2. Overall, ScanSSC, which utilizes features from all three axes, demonstrates significantly more plausible results. Using features from only a single axis tends to result in inaccurate predictions for distant vehicles, side road areas, and occluded objects. In contrast, ScanSSC, which combines features from all three axes, achieves significantly more reliable and accurate predictions. We hypothesize that this improvement arises from the complementary nature of features from each axis, which together enable a more comprehensive understanding of the entire scene.

E. Analysis

Quantitative Result for Distant Geometry. This study aims to improve the overall reconstruction in distant regions. To support this, quantitative analysis results are presented in Tab.2 of the manuscript. Additionally, for a more detailed analysis, Tab. E.1 provides group-wise mIoU for distant $1/2$ regions along each axis ($1/4$ on both sides for the width axis), following the categorization from the official SemanticKITTI [1] website.

This result highlights ScanSSC's effectiveness in distant

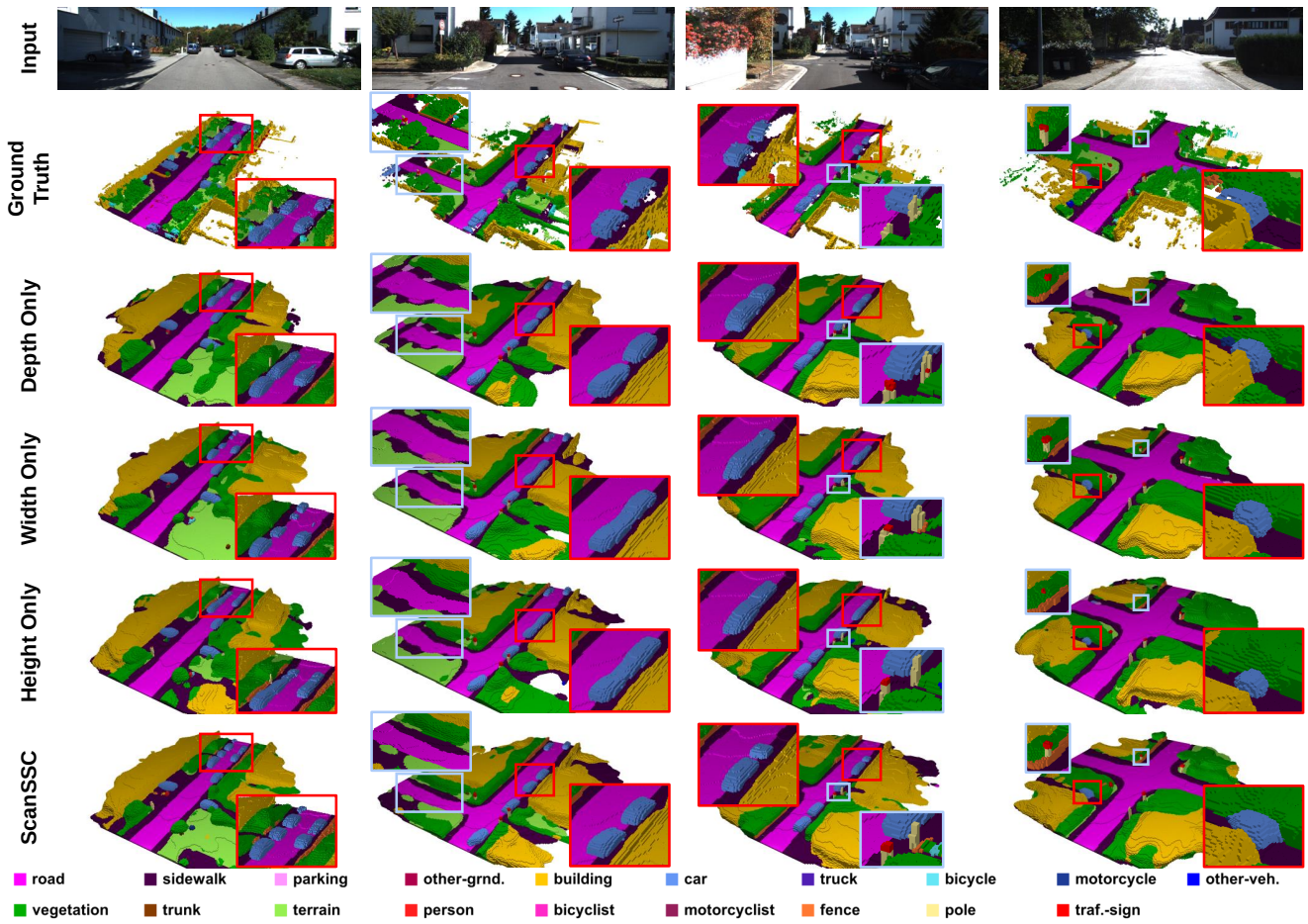


Figure D.2. Visualization and comparison of the results of applying the Scan Module and Scan Loss to each individual axis and ScanSSC on the SemanticKITTI [1] validation set.

		Large class group				Small class group			
Method	Axis	Ground	Structure	Nature	Total	Vehicle	Human	Object	Total
CGFormer	Dep.	24.87	17.66	**19.57**	21.98	6.92	0.07	2.86	3.95
ScanSSC	Dep.	**26.29**	**17.89**	19.23	**22.60**	**7.03**	**0.19**	**3.19**	**4.11**
CGFormer	Wid.	14.23	**15.81**	**16.51**	15.28	1.24	0.15	1.35	0.97
ScanSSC	Wid.	**16.26**	14.65	15.96	**15.95**	**1.39**	**0.57**	**1.49**	**1.19**
CGFormer	Hgt.	29.90	**23.15**	25.73	27.49	13.23	2.79	6.73	8.61
ScanSSC	Hgt.	**31.69**	21.50	**25.91**	**28.25**	**13.78**	**3.44**	**7.11**	**9.14**

Table E.1. Per-group mIoUs on the SemanticKITTI [1] validation set.

regions, demonstrating its superior performance on both large and small geometries, particularly outperforming CGFormer [17] in all small class groups.

Analysis of the Non-Axis-Aligned Cases. Since the proposed Scan Module and Scan Loss operate axis-wise, ScanSSC is effective in most axis-aligned driving scenarios, as demonstrated by the qualitative results in the manuscript. However, this raises the question of whether ScanSSC's operation might be less effective in non-axis-aligned scenes. To investigate this, we conduct additional evaluations of ScanSSC in non-axis-aligned scenarios. Since the SemanticKITTI benchmark dataset does not explicitly categorize curve road scenes, we manually classify these cases. Numerically, ScanSSC significantly outperforms CGFormer, achieving a mIoU of 14.56 and an IoU of 42.38, compared to CGFormer's 13.77 and 41.96. As shown in Fig. E.1, ScanSSC performs comparably overall without side effects. Specifically, its performance is on par for small objects; however, it reconstructs roads significantly better in distant regions.

Analysis of the Various Scene Conditions. We conduct additional analyses to demonstrate the superiority of ScanSSC under various conditions (e.g., shadow, occlusion). Since existing benchmark datasets for SSC do not categorize various environments, we manually filter shady and highly occluded scenarios from the RGB images of the SemanticKITTI dataset.

As shown in Fig. E.2, in the shady scenario, ScanSSC clearly distinguishes both nearby and distant vehicles covered by shadows, whereas CGFormer fails to do so. In addition, in the occluded scenario, ScanSSC effectively reconstructs the right-side road obscured by nearby vehicles and accurately identifies distant cars that are partially occluded. These results demonstrate ScanSSC's robustness across di-

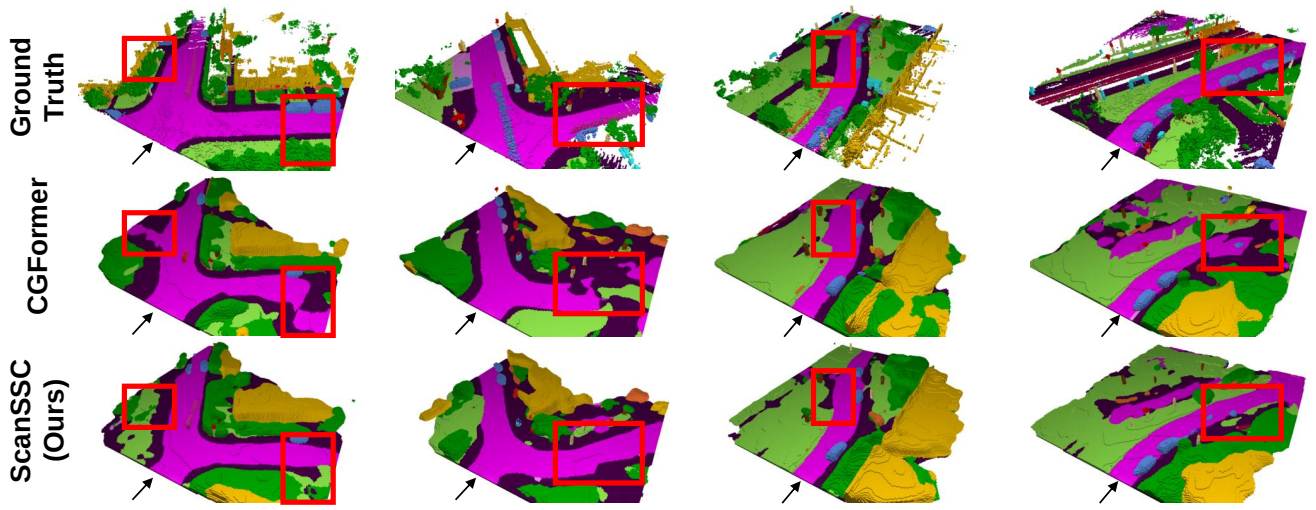


Figure E.1. Visualization results of the non-axis-aligned cases of CGFormer [17] and ScanSSC on the SemanticKITTI [1] validation set.

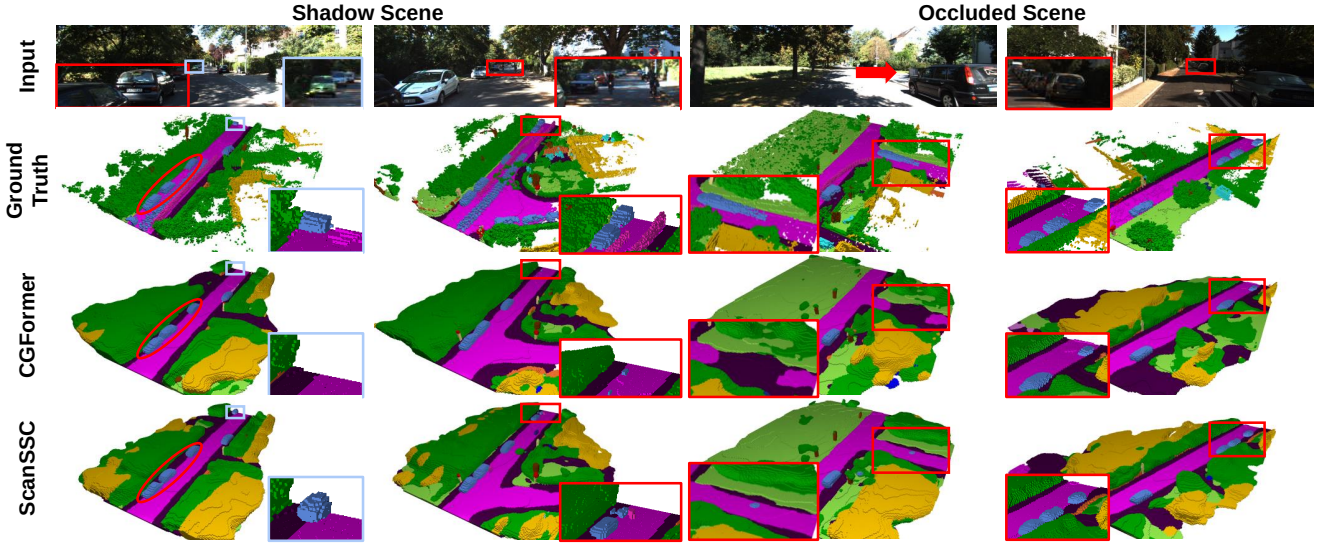


Figure E.2. Visualization results of CGFormer [17] and ScanSSC under various conditions (e.g., shadow, occlusion) on the SemanticKITTI [1] validation set.

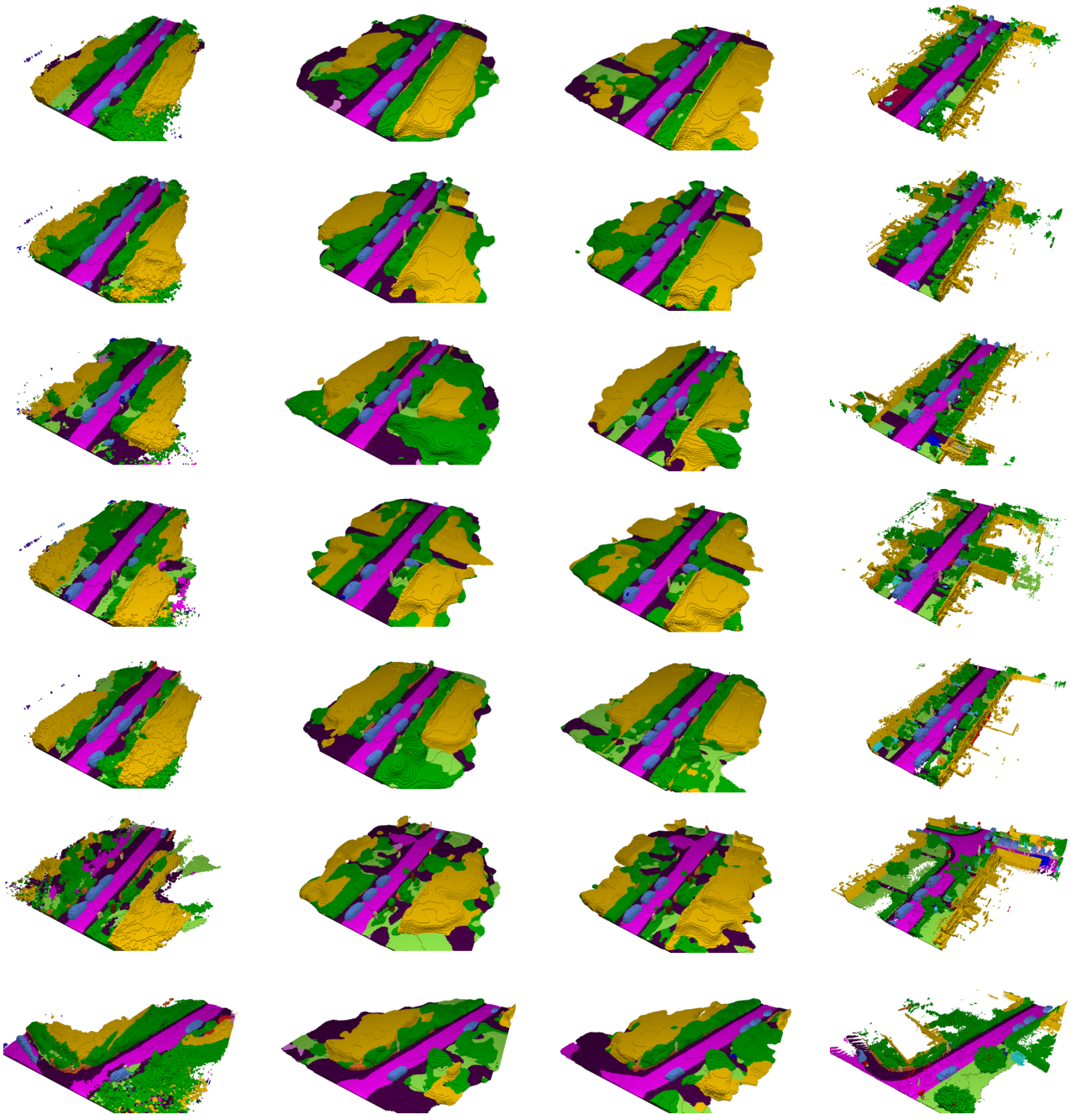
verse and challenging scenes.

F. Additional Qualitative Results

We present additional qualitative comparisons with VoxFormer [9] and CGFormer [17], as visualized in Fig. F.1. These results are randomly selected from the SemanticKITTI [1] validation set.

G. Pytorch-like Pseudocode of Scan Module and Scan Loss

To facilitate a comprehensive understanding of the proposed Scan Module and Scan Loss, we present the PyTorch-like [13] pseudocode for each in Algorithm G.1 and Algorithm G.2, respectively.



(a) VoxFormer [9] (b) CGFormer [17] (c) ScanSSC (Ours) (d) Ground Truth

Figure F.1. More qualitative comparison results on the SemanticKITTI [1] validation set.

Algorithm G.1 PyTorch Style Pseudocode of Scan Module.

```
import torch
import torch.nn as nn

class ScanModule(nn.Module):
  def __init__(self, dim):
    # declare axis-specific Scan Blocks
    self.dep_block = ScanBlock(dim)
    self.wid_block = ScanBlock(dim)
    self.hgt_block = ScanBlock(dim)

  def forward(self, x):
    X_, Y_, Z_, C = x.size()
    x_dep = x.permute(1, 2, 0, 3).flatten(0,1)
    x_wid = x.permute(0, 2, 1, 3).flatten(0,1)
    x_hgt = x.flatten(0,1)

    # axis-specific masks
    dep_attn_mask =
      torch.triu(torch.ones(X_, X_), diagonal=1)==1
    dep_attn_mask[:, :X_//2] = False
                # depth-axis margin region
    wid_attn_mask = torch.tril(
        torch.ones(Y_//2, Y_//2), diagonal=-1)==1
    wid_attn_mask = torch.cat((wid_attn_mask,
        wid_attn_mask.flip(dim=[-1])), dim=-1)
    wid_attn_mask = torch.cat((wid_attn_mask,
        wid_attn_mask.flip(dims=[-2])), dim=0)
    wid_attn_mask[:, Y_//4:-(Y_//4)] = False
                # width-axis margin region
    hgt_attn_mask =
      torch.tril(torch.ones(Z_), diagonal=-1)==1

    # axis-wise voxel scanning
    x_dep = self.dep_block(x_dep, dep_attn_mask)
    x_wid = self.wid_block(x_wid, wid_attn_mask)
    x_hgt = self.hgt_block(x_hgt, hgt_attn_mask)

    x_dep =
      x_dep.reshape(Y_, Z_, X_, C).permute(2, 0, 1, 3)
    x_wid =
      x_wid.reshape(X_, Z_, Y_, C).permute(0, 2, 1, 3)
    x_hgt = x_hgt.reshape(X_, Y_, Z_, C)

    return x_dep, x_wid, x_hgt

class ScanBlock(nn.Module):
  def __init__(self, dim):
    self.norm1 = nn.LayerNorm(dim)
    self.masked_sa = nn.MultiheadAttention(dim)
    self.norm2 = nn.LayerNorm(dim)
    self.ff1 = nn.Linear(dim, dim*2)
    self.activation = nn.ReLU()
    self.ff2 = nn.Linear(dim*2, dim)

  def forward(self, x, attn_mask):
    B_, L_, C = x.size()

    # Masked Self-Attention
    x_norm1 = self.norm1(x)
    x = x + self.masked_sa(x_norm1, x_norm1, x_norm1,
                           attn_mask = attn_mask)

    # Feed Forward Network
    x_norm2 = self.norm2(x)
    x = x + self.ff2(self.activation(self.ff1(x_norm2)))

    return x
```

Algorithm G.2 PyTorch Style Pseudocode of $\mathcal{L}_{scan}$.

```
import torch
import torch.nn.functional as F

def ScanLoss(logit, target):
  P, X_, Y_, Z_ = logit.size()
  # back to front
  cum_x = torch.cumsum(logit.flip((1)), axis=1)
  # sides to center
  cum_y_l = torch.cumsum(logit[:Y_//2], axis=2)
  cum_y_r = torch.cumsum(logit[Y_//2:].flip((2)), axis=2)
  cum_y = torch.cat([cum_y_l, cum_y_r], dim=2)
  # bottom to top
  cum_z = torch.cumsum(logit, axis=3)

  # to logit value scaling
  cum_x /= torch.arange(1, X_+1)
  cum_y /= torch.arange(1, Y_+1)
  cum_z /= torch.arange(1, Z_+1)

  # same with logits
  X_, Y_, Z_ = target.size()
  target = F.one_hot(target).permute(3,0,1,2)
  cum_x_t = torch.cumsum(target.flip((1)), dim=1)
  cum_y_l_t = torch.cumsum(target[:Y_//2], dim=2)
  cum_y_r_t = torch.cumsum(target[Y_//2:].flip((2)),
                                          dim=2)
  cum_y_t = torch.cat([cum_y_l_t, cum_y_r_t], axis=2)
  cum_z_t = torch.cumsum(target, dim=3)

  cum_x_t /= torch.arange(1, X_+1)
  cum_y_t /= torch.arange(1, Y_+1)
  cum_z_t /= torch.arange(1, Z_+1)

  L_scan_x = F.cross_entropy(cum_x, cum_x_t,
                             reduction='mean')
  L_scan_y = F.cross_entropy(cum_y, cum_y_t,
                             reduction='mean')
  L_scan_z = F.cross_entropy(cum_z, cum_z_t,
                             reduction='mean')
  L_scan = L_scan_x + L_scan_y + L_scan_z

return L_scan
```

References

[1] 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 *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019. 1, 2, 3, 4, 5

[2] Anh-Quan Cao and Raoul De Charette. Monoscene: Monocular 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3991–4001, 2022. 1

[3] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012. 1

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1

[5] Yuanhui Huang, Wenzhao Zheng, Yunpeng Zhang, Jie Zhou, and Jiwen Lu. Tri-perspective view for vision-based 3d semantic occupancy prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9223–9232, 2023. 1

[6] Haoyi Jiang, Tianheng Cheng, Naiyu Gao, Haoyang Zhang, Tianwei Lin, Wenyu Liu, and Xinggang Wang. Symphonize 3d semantic scene completion with contextual instance queries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20258–20267, 2024. 1

[7] Bohan Li, Jiajun Deng, Wenyao Zhang, Zhujin Liang, Dalong Du, Xin Jin, and Wenjun Zeng. Hierarchical temporal context learning for camera-based semantic scene completion. *arXiv preprint arXiv:2407.02077*, 2024. 1

[8] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection, 2022. 1

[9] Yiming Li, Zhiding Yu, Christopher Choy, Chaowei Xiao, Jose M Alvarez, Sanja Fidler, Chen Feng, and Anima Anandkumar. Voxformer: Sparse voxel transformer for camera-based 3d semantic scene completion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9087–9098, 2023. 1, 4, 5

[10] Yiming Li, Sihang Li, Xinhao Liu, Moonjun Gong, Kenan Li, Nuo Chen, Zijun Wang, Zhiheng Li, Tao Jiang, Fisher Yu, Yue Wang, Hang Zhao, Zhiding Yu, and Chen Feng. Sscbench: A large-scale 3d semantic scene completion benchmark for autonomous driving, 2024. 1

[11] Yiyi Liao, Jun Xie, and Andreas Geiger. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3292–3310, 2022. 1

[12] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[13] A Paszke. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019. 4

[14] Faranak Shamsafar, Samuel Woerz, Rafia Rahim, and Andreas Zell. Mobilestereonet: Towards lightweight deep networks for stereo matching, 2021. 1

[15] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020. 1

[16] Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Jie Zhou, and Jiwen Lu. Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21729–21740, 2023. 1

[17] Zhu Yu, Runmin Zhang, Jiacheng Ying, Junchen Yu, Xiaohai Hu, Lun Luo, Si-Yuan Cao, and Hui-Liang Shen. Context and geometry aware voxel transformer for semantic scene completion, 2024. 1, 3, 4, 5

[18] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018. 2