Supplementary Material
MemorySeg: Online LiDAR Semantic Segmentation with a Latent Memory

Enxu Li, Sergio Casas, Raquel Urtasun
Waabi, University of Toronto
{tli, sergio, urtasun}@waabi.ai

In this supplementary material, we first describe the implementation details including the Memory Refinement Module (in Sec. 1.1) and the added motion header (in Sec. 1.2) for separating moving and static actors in SemanticKITTI [2]. Subsequently, we show the addition results of the following:

- MEMORYSEG compared against state-of-the-art methods on test and validation set of SemanticKITTI [2] single-scan benchmark in Sec. 2.1
- MEMORYSEG compared against our baseline on validation set of SemanticKITTI [2] multi-scan benchmark in Sec. 2.2
- MEMORYSEG compared against state-of-the-art and our baselines on validation set of nuScenes [3] in Sec. 2.3
- ablation analysis of memory voxel size, padding neighbourhood size, instance cutMix and test-time augmentation in Sec. 2.4
- visualization of the latent memory in Sec. 2.5
- additional qualitative results of MEMORYSEG compared with our baseline in Sec. 2.6

1. Implementation Details

1.1. Details on Memory Refinement Module

Memory Refinement Module (MRM) is an improved version of ConvGRU [1] that updates the latent memory with the current observation embeddings as follows,

$$\begin{aligned} r_t &= \text{sigmoid}[\Psi_r(X'_{F,t}, H'_{F,t-1})], \\ z_t &= \text{sigmoid}[\Psi_z(X'_{F,t}, H'_{F,t-1})], \\ \hat{H}_{F,t} &= \tanh[\Psi_u(X'_{F,t}, r_t \cdot H'_{F,t-1})], \\ H_{F,t} &= \hat{H}_{F,t} \cdot z_t + H'_{F,t-1} \cdot (1 - z_t), \end{aligned} \tag{1}$$

where $X'_{F,t}$ is the current observation embeddings at time t, $H'_{F,t-1}$ is the latent memory embeddings at $t-1$, and $H_{F,t}$ is the updated latent memory. Ψ_r, Ψ_z, Ψ_u are a single sparse 3D convolutional layer in the vanilla sparse ConvGRU [1]. However, we introduce a new design where they are implemented as sparse 3D convolutional blocks. These blocks integrate downsampling layers to expand the receptive field and upsampling layers to restore the embeddings to their original size. We provide a more detailed illustration of this design in Fig. 2.

1.2. Details on Motion Header

In this section, we explain how we implemented the motion header in the decoder to classify movable actors as either moving or non-moving in the SemanticKITTI dataset [2]. We observed that there are no static motorcyclists or bicyclists in the training set. This means that using separate logits for moving and non-moving classes, as conventionally designed, will fail to identify any static bicyclists or motorcyclists, as there is no training data to learn from. To address this issue,

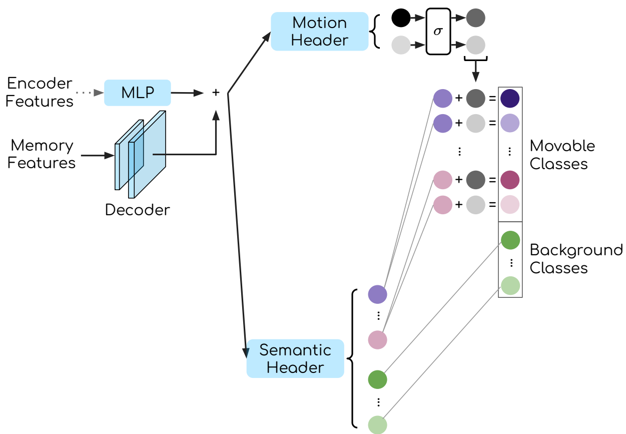

Figure 1. Illustration of the modification of the decoder for the multi-scan benchmark of SemanticKITTI [2]. σ denotes the *Log-SoftMax* operation to normalize the motion features. The motion logits are broadcasted and added to the movable logtis from the semantic header in the *Log* space.

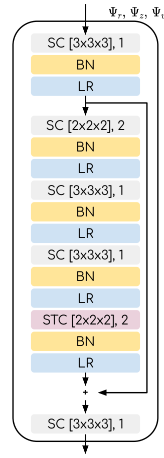

Figure 2. Illustration of the sparse convolutional blocks (Ψ_r, Ψ_z, Ψ_u) in MRM. SC: sparse 3D convolution [kenel size], stride. STC: 3D sparse transpose convolution [kernel size], stride. BN: BatchNorm. LR: LeakyReLU.

we added a motion header to perform binary segmentation of moving and non-moving objects, and later fused it with the semantic header. By training the network to distinguish between moving and non-moving objects, such as pedestrians and vehicles, we aim to enable the network to recognize static and moving bicyclists and motorcyclists, even in the absence of any training data for those classes. Specifically, we apply *LogSoftMax* to normalize the motion logits and add them to each of the semantic logits that belong to movable classes. Consequently, we form moving and static logits for each of the movable logit. The implementation details are illustrated in Fig. 1. The network is supervised by applying segmentation loss (*i.e.* weighted combination of cross entropy, Lovasz softmax, and the proposed regularizer) on the motion logits, semantic logits and the final logits.

2. Additional Results

2.1. SemanticKITTI single-scan results

Tab. 1 compares MemorySeg with state-of-the-art approaches on the test set of SemanticKITTI single-scan benchmark. This is a more competitive benchmark focusing on single-scan semantic segmentation where previous research has focused on proposing various architectures [4, 16, 20] or knowledge distillation techniques [8]. Our results show that MemorySeg can still outperform these methods, which are highly optimized for this benchmark. Tab. 2 compares our apporach with the others on the validation set of the same benchmark. Please note that most prior works have only reported the mIoU metric on the validation set. Therefore, we only included comparison of mIoU in this table but presented detailed class-wise IoUs of our approach in Tab. 5.

2.2. SemanticKITTI multi-scan validation results

Tab.3 compares MemorySeg with the 5-frame-baseline (5FB). The 5FB uses the same network architecture of MemorySeg but without the memory update module. Additionally, the input is 5 consecutive LiDAR scans projected to the most recent ego vehicle frame. In contrast, our approach processes only one scan at a time. The results indicate that MemorySeg significantly outperforms 5FB in almost all categories. The improvement is most prominent in the case of movable objects such as *moving bicyclist* and *other vehicle*. The 5FB can only reason about motion over a short time interval (i.e., 5 frames of data or approximately 0.5 seconds), as processing longer sequences all at once is computationally infeasible. However, MemorySeg employs a latent 3D memory to encode information from a more extended period, enabling the network to better comprehend the motion of moving actors.

2.3. nuScenes validation results

In Tab.4, we compare against state-of-the-art methods and our baseline on the validation set of nuScenes [3]. MemorySegagain outperforms all methods with the largest gains observed in smaller objects such as *bicycle*, *pedestrian*, and *traffic*

Method	**mIoU**	Car	Bicycle	Motorcycle	Truck	Other Vehicle	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other Ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic Sign
PointNet [11]	14.6	46.3	1.3	0.3	0.1	0.8	0.2	0.2	0.0	61.6	15.8	35.7	1.4	41.4	12.9	31.0	4.6	17.6	2.4	3.7
RangeNet++ [10]	52.2	91.4	25.7	34.4	25.7	23.0	38.3	38.8	4.8	91.8	65.0	75.2	27.8	87.4	58.6	80.5	55.1	64.6	47.9	55.9
RandLANet [9]	53.9	94.2	26.0	25.8	40.1	38.9	49.2	48.2	7.2	90.7	60.3	73.7	20.4	86.9	56.3	81.4	61.3	66.8	49.2	47.7
PolarNet [18]	54.3	93.8	40.3	30.1	22.9	28.5	43.2	40.2	5.6	90.8	61.7	74.4	21.7	90.0	61.3	84.0	65.5	67.8	51.8	57.5
SqueezeSegv3 [15]	55.9	92.5	38.7	36.5	29.6	33.0	45.6	46.2	20.1	91.7	63.4	74.8	26.4	89.0	59.4	82.0	58.7	65.4	49.6	58.9
TemporalLidarSeg[6]	58.2	94.1	50.0	45.7	28.1	37.1	56.8	47.3	9.2	91.7	60.1	75.9	27.0	89.4	63.3	83.9	64.6	66.8	53.6	60.5
KPConv [13]	58.8	96.0	30.2	42.5	33.4	44.3	61.5	61.6	11.8	88.8	61.3	72.7	31.6	90.5	64.2	84.8	69.2	69.1	56.4	47.4
SalsaNext [5]	59.5	91.9	48.3	38.6	38.9	31.9	60.2	59.0	19.4	91.7	63.7	75.8	29.1	90.2	64.2	81.8	63.6	66.5	54.3	62.1
Meta-RangeSeg [14]	61.0	93.9	50.1	43.8	43.9	43.2	63.7	53.1	18.7	90.6	64.3	74.6	29.2	91.1	64.7	82.6	65.5	65.5	56.3	64.2
FusionNet [17]	61.3	95.3	47.5	37.7	41.8	34.5	59.5	56.8	11.9	91.8	68.8	77.1	30.8	**92.5**	**69.4**	84.5	69.8	68.5	60.4	66.5
TornadoNet [7]	63.1	94.2	55.7	48.1	40.0	38.2	63.6	60.1	34.9	89.7	66.3	74.5	28.7	91.3	65.6	85.6	67.0	71.5	58.0	65.9
SPVNAS [12]	67.0	97.2	50.6	50.4	56.6	58.0	67.4	67.1	50.3	90.2	67.6	75.4	21.8	91.6	66.9	86.1	73.4	71.0	64.3	67.3
Cylinder3D [20]	67.8	97.1	67.6	64.0	**59.0**	58.6	73.9	67.9	36.0	91.4	65.1	75.5	32.3	91.0	66.5	85.4	71.8	68.5	62.6	65.6
(AF)2S3-Net [4]	69.7	94.5	65.4	**86.8**	39.2	41.1	**80.7**	**80.4**	**74.3**	91.3	68.8	72.5	**53.5**	87.9	63.2	70.2	68.5	53.7	61.5	**71.0**
RPVNet [16]	70.3	**97.6**	**68.4**	68.7	44.2	**61.1**	**75.9**	74.4	**73.4**	**93.4**	**70.3**	**80.7**	33.3	**93.5**	**72.1**	**86.5**	**75.1**	**71.7**	64.8	61.4
PVKD [8]	71.2	97.0	67.9	**69.3**	53.5	60.2	75.1	73.5	50.5	**91.8**	**70.9**	**77.5**	**41.0**	92.4	69.4	**86.5**	**73.8**	**71.9**	**64.9**	65.8
MemorySeg [ours]	**71.3**	**97.4**	**68.1**	69.1	**58.7**	**65.7**	75.2	**76.4**	56.2	89.8	65.6	74.8	32.1	91.9	67.8	85.2	73.7	70.5	**66.4**	**70.1**

Table 1. Comparison to the state-of-the-art methods on the test set of SemanticKITTI [2] single-scan benchmark. We include LiDAR-only published approaches at the time of submission. Metrics are provided in [%]. Top two entries of each classes are bolded.

Method	**mIoU**
RandLANet [9]	57.1
PolarNet [18]	54.9
TornadoNet [7]	64.5
SPVNAS [12]	64.7
Cylinder3D [20]	65.9
PVKD [8]	66.4
RPVNet [16]	69.6
MemorySeg [ours]	70.8
MemorySeg [ours] + TTA	**71.5**

Table 2. Comparison to the state-of-the-art methods on the validation set of SemanticKITTI [2]. Metrics are provided in [%].

Method	**mIoU**	Car	Bicycle	Motorcycle	Truck	Other Vehicle	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other Ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic Sign	car (m)	bicyclist (m)	person (m)	motorcyclist (m)	other-vehicle (m)	truck (m)
5FB	53.5	95.4	56.0	75.6	74.5	53.5	**25.5**	0.0	0.0	92.7	46.6	78.8	0.5	90.1	58.6	89.2	72.8	75.3	66.2	**53.7**	**78.6**	87.2	64.6	0.0	3.4	0.0
MemorySeg	**58.5**	**95.9**	**64.8**	**86.2**	**96.3**	**66.4**	23.7	0.0	0.0	**95.5**	**55.7**	**83.9**	**5.0**	**91.4**	**62.7**	**89.7**	**73.9**	**78.1**	**66.7**	52.1	72.7	**94.4**	**72.0**	0.0	**35.7**	0.0

Table 3. Comparison to our 5-frame baseline (5FB) on the validation set of SemanticKITTI [2]. Movable actors are further divided into moving and static. Metrics are provided in [%].

Method	**mIoU**	**FW-mIoU**	Barrier	Bicycle	Bus	Car	Construction	Motorcycle	Pedestrain	Traffic Cone	Trailer	Truck	Drivable	Other Flat	Sidewalk	Terrain	Manmade	Vegetation
RangeNet++ [10]	65.5	–	66.0	21.3	77.2	80.9	30.2	66.8	69.6	52.1	54.2	72.3	94.1	66.6	63.5	70.1	83.1	79.8
PolarNet [18]	71.0	–	74.7	28.2	85.3	90.9	35.1	77.5	71.3	58.8	57.4	76.1	96.5	71.1	74.7	74.0	87.3	85.7
Salsanext [5]	72.2	–	74.8	34.1	85.9	88.4	42.2	72.4	72.2	63.1	61.3	76.5	96.0	70.8	71.2	71.5	86.7	84.4
Cylinder3D [20]	76.1	–	76.4	40.3	91.2	**93.8**	51.3	78.0	78.9	64.9	62.1	84.4	96.8	71.6	**76.4**	75.4	90.5	87.4
RPVNet [16]	77.6	–	78.2	43.4	92.7	93.2	49.0	85.7	80.5	66.0	66.9	84.0	96.9	73.5	75.9	**76.0**	90.6	88.9
SFB	76.7	89.2	77.6	42.0	92.7	92.5	44.7	83.8	79.1	65.1	66.2	81.6	96.7	75.9	75.1	75.2	90.2	88.7
MemorySeg [ours]	**81.1**	**90.0**	**78.8**	**57.0**	**95.2**	92.9	**60.0**	**89.3**	**86.3**	**70.8**	**73.8**	**87.2**	**96.9**	**76.4**	75.8	75.3	**91.5**	**89.8**

Table 4. Comparison to the state-of-the-art methods on the validation set of nuScenes [3] LiDAR semantic segmentation benchmark. We include LiDAR-only published approaches that report their validation mIoU. Metrics are provided in [%].

cone. Those are particularly challenging for semantic segmentation networks due to the sparsity of the point clouds in this dataset. Nonetheless, our method overcomes this limitation by leveraging a 3D latent memory to enhance semantic reasoning of the sparse points.

Method	mIoU	Car	Bicycle	Motorcycle	Truck	Other Vehicle	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other Ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic Sign
SFB w/o cutMix	66.2	96.0	54.2	76.7	78.5	53.5	71.2	92.0	0.7	94.5	49.3	82.2	3.8	91.0	63.8	88.7	71.2	76.0	64.6	49.4
SFB	67.2	96.9	60.0	79.5	76.9	67.0	74.0	91.2	1.6	94.5	49.4	82.0	6.1	90.6	61.0	88.0	69.0	74.4	64.8	50.8
M1	69.5	97.5	62.4	88.0	79.1	74.4	83.7	93.2	2.5	95.3	55.7	83.5	3.3	90.4	61.8	88.1	69.5	74.7	64.9	52.2
M2	69.7	97.6	58.4	86.2	94.5	77.8	83.1	94.0	0.0	94.9	54.5	83.0	3.9	90.9	63.5	87.3	68.8	71.5	65.1	50.0
M3	69.7	97.3	59.4	84.9	82.5	73.7	83.1	93.7	8.9	94.9	50.1	82.4	4.6	90.8	61.6	89.2	70.8	77.2	66.0	52.6
M4 [ours]	70.8	97.4	61.5	89.1	93.0	76.2	83.6	95.0	0.3	95.3	52.4	83.0	7.5	91.4	64.1	89.3	73.6	77.2	65.6	50.4
M4 [ours] + TTA	**71.5**	97.9	64.6	89.3	95.4	81.9	84.6	95.2	0.0	95.6	52.9	83.9	2.9	91.3	62.9	89.7	74.4	77.8	66.5	51.8

Table 5. Class-wise IoUs of our ablated methods on the validation set of SemanticKITTI [2]. Metrics are provided in [%].

Memory Vox Size	mIoU	Car	Bicycle	Motorcycle	Truck	Other Vehicle	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other Ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic Sign
$v_m = 0.25$ m	70.2	97.1	65.4	89.4	90.3	69.2	78.1	94.9	2.4	95.2	53.9	83.1	4.9	91.3	63.2	89.3	69.8	77.1	65.3	53.7
$v_m = 0.5$ m	**70.8**	97.4	61.5	89.1	93.0	76.2	83.6	95.0	0.3	95.3	52.4	83.0	7.5	91.4	64.1	89.3	73.6	77.2	65.6	50.4
$v_m = 1.0$ m	70.1	97.2	59.2	84.8	91.5	74.2	79.4	93.7	0.3	95.1	55.4	82.5	14.5	90.7	61.3	88.7	71.5	76.1	64.2	50.9
$v_m = 2.0$ m	67.9	97.2	55.1	76.3	79.8	73.6	74.3	92.0	1.4	94.6	53.4	82.0	7.3	90.8	61.9	87.9	71.1	74.4	65.4	52.3

Table 6. Ablation results of different memory voxel size v_m on the validation set of SemanticKITTI [2]. Metrics are provided in [%].

APM Neighbours	mIoU	Car	Bicycle	Motorcycle	Truck	Other Vehicle	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other Ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic Sign
k=3	69.7	97.2	61.3	84.8	90.8	73.5	74.5	92.8	3.5	95.2	52.2	82.9	6.1	91.6	64.2	88.4	73.4	75.0	65.2	52.1
k=5	**70.8**	97.4	61.5	89.1	93.0	76.2	83.6	95.0	0.3	95.3	52.4	83.0	7.5	91.4	64.1	89.3	73.6	77.2	65.6	50.4
k=8	70.0	97.3	59.7	84.8	93.7	75.1	79.9	94.0	0.4	94.9	54.2	82.6	6.4	91.4	63.0	88.7	72.0	75.8	64.9	51.7

Table 7. Ablation results of different padding neighbourhood sizes k in APM on the validation set of SemanticKITTI [2]. Metrics are provided in [%].

2.4. Ablations

Tab.5 presents the detailed class-wise IoUs of the model for ablation presented in the main text. Please note that we follow the semantic class mapping of the single-scan benchmark while conducting ablation analysis on SemanticKITTI. This is due to the significant class imbalance that arises when attempting to separate all movable actors into moving and static classes. For instance, the validation set will not include any moving trucks or moving motorcyclists, and there will be less than 1000 points of static bicyclists. This can lead to increased noise in the ablation results. Therefore, we maintain the 19 semantic classes during the ablation process to ensure more robust results.

Influence of memory voxel size Tab. 6 shows the results of our ablation experiments using different voxel sizes (v_m) to retain latent memory. We found that smaller object classes, such as *bicycle* and *motorcycle*, benefited from using smaller voxel sizes. However, using a large voxel size, such as 2m, resulted in much worse performance for these classes, possibly because it mixed different objects within the same voxel. Overall, using a memory voxel size of 0.5m produced the best results.

Influence of padding neighbourhood size in APM We present the results of our experiment on using different neighborhood sizes to aggregate embeddings for padding, as shown in Tab. 7. Specifically, we tested neighborhood sizes of 3, 5, and 8 entries. We found that changing the padding neighborhood size had only a minimal effect on the background classes, which are typically static and do not move. This is because the closest entries usually have the most influence, so varying the neighborhood size had little impact. However, we observed more significant differences in the movable actors. For example, increasing the padding neighborhood size was most beneficial for the *truck* class, where larger receptive fields are needed to aggregate potentially moving trucks. Overall, aggregating the closest 5 entries (k=5) produced the most favorable results.

Influence of instance cutMix The SemanticKITTI [2] dataset contains many frames without movable actors, such as pedestrians and riders. The training set, on average, has only 0.63 pedestrians and 0.18 riders per frame. To address this significant class imbalance, we created an instance library that includes movable instances from the training sequences similar to [19]. In each training iteration, we randomly select 5 instances from the library and add them to the scene. The sampling weight is determined by the inverse frequency of the class. This approach has been effective, as shown in Tab. 5, resulting in an improvement from 66.2% to 67.2%, with the most significant gains coming from the movable actors added during training.

Influence of test-time augmentation We follow existing works [20, 8] to apply test-time augmentation (TTA) for further improving the segmentation results. Specifically, we randomly sample an augmentation that includes rotation on the Z axis from $-\pi$ to π and a global scaling factor ranging from 0.95 to 1.05. We apply this same augmentation to the entire sequence during inference and repeat the process 10 times, each with a different augmentation. We then average the prediction results from each of the 10 passes to obtain the final prediction. Our experiments demonstrate that TTA improves the mIoU by 0.7%, with a slight improvement observed in every class IoU, as shown in the last row of Tab. 5.

2.5. Visualization of memory

We present a visualization of the 3D latent memory in Fig. 3, where PCA is used to reduce the embedding dimension to RGB. On the right side of the memory, we display the prediction generated by our network on the single scan. It is difficult to identify objects in a single scan due to the lack of semantic information and sparse observations, and occluded regions have no observations at all. In contrast, our latent memory is much denser, contains rich semantics that help separate different classes, and provides contextual details in occluded areas.

2.6. Qualitative comparison

Fig. 4 shows a qualitative comparison with our baseline. Please focus your attention to the two vehicles parked on the far left, highlighted with red circles. These scenarios are difficult for semantic segmentation because of the limited observations and partial occlusions. Despite these challenges, MemorySeg consistently segments the object accurately without any flickering. Conversely, the single-frame baseline fails to identify the parked vehicle in some frames, and the segmentation results fluctuate over time.

Furthermore, we present another qualitative example from the nuScenes [3] dataset in Fig. 5, where we demonstrate substantial improvements in the background classes. Those classes often require an understanding of the surrounding environment to be segmented correctly. Our method improves contextual reasoning by accumulating past observations using a latent memory representation. Hence, while the single-frame baseline (SFB) is prone to errors, our approach yields accurate and reliable results.

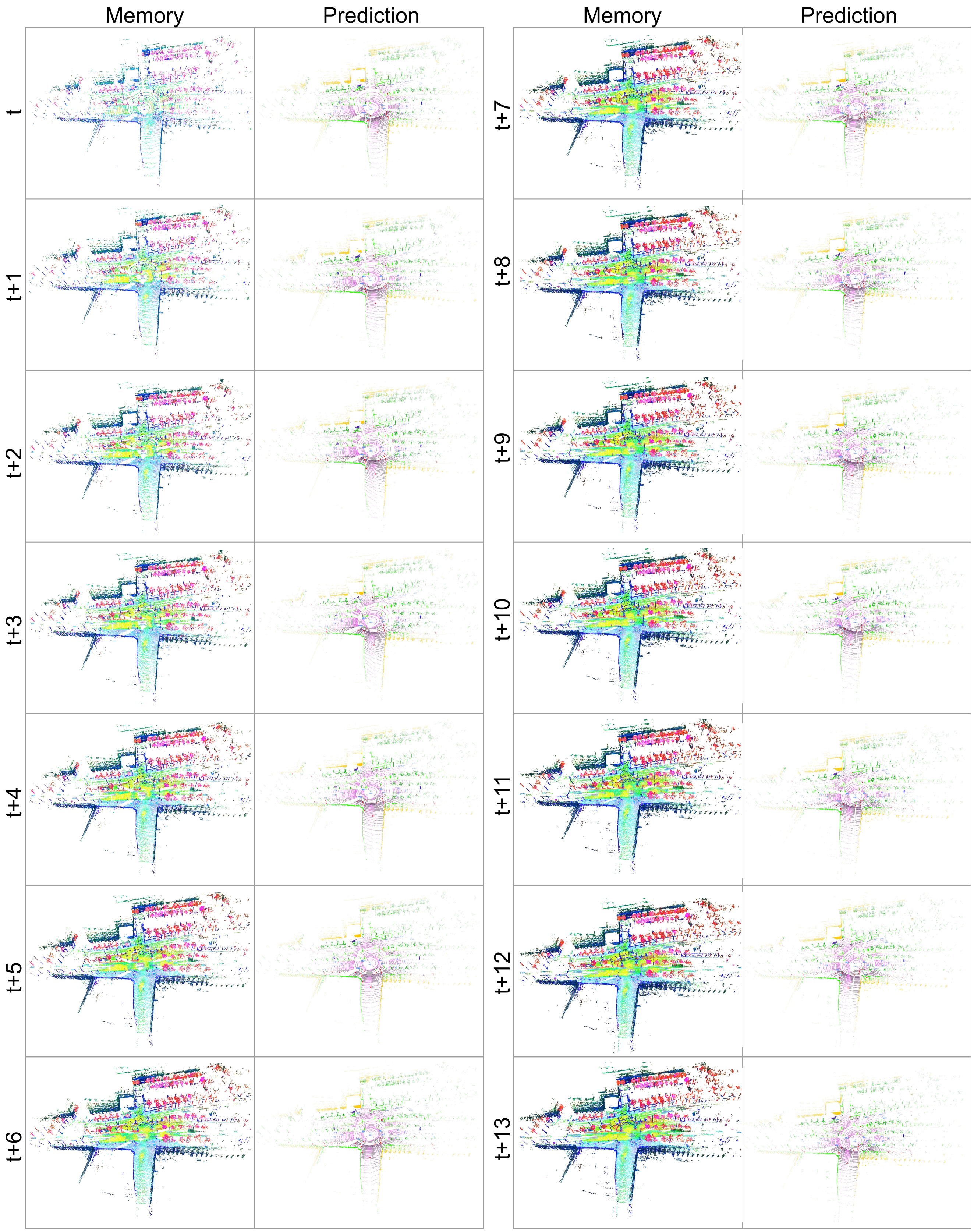

Figure 3. Illustration of the latent memory and prediction when unrolling the LiDAR sequence.

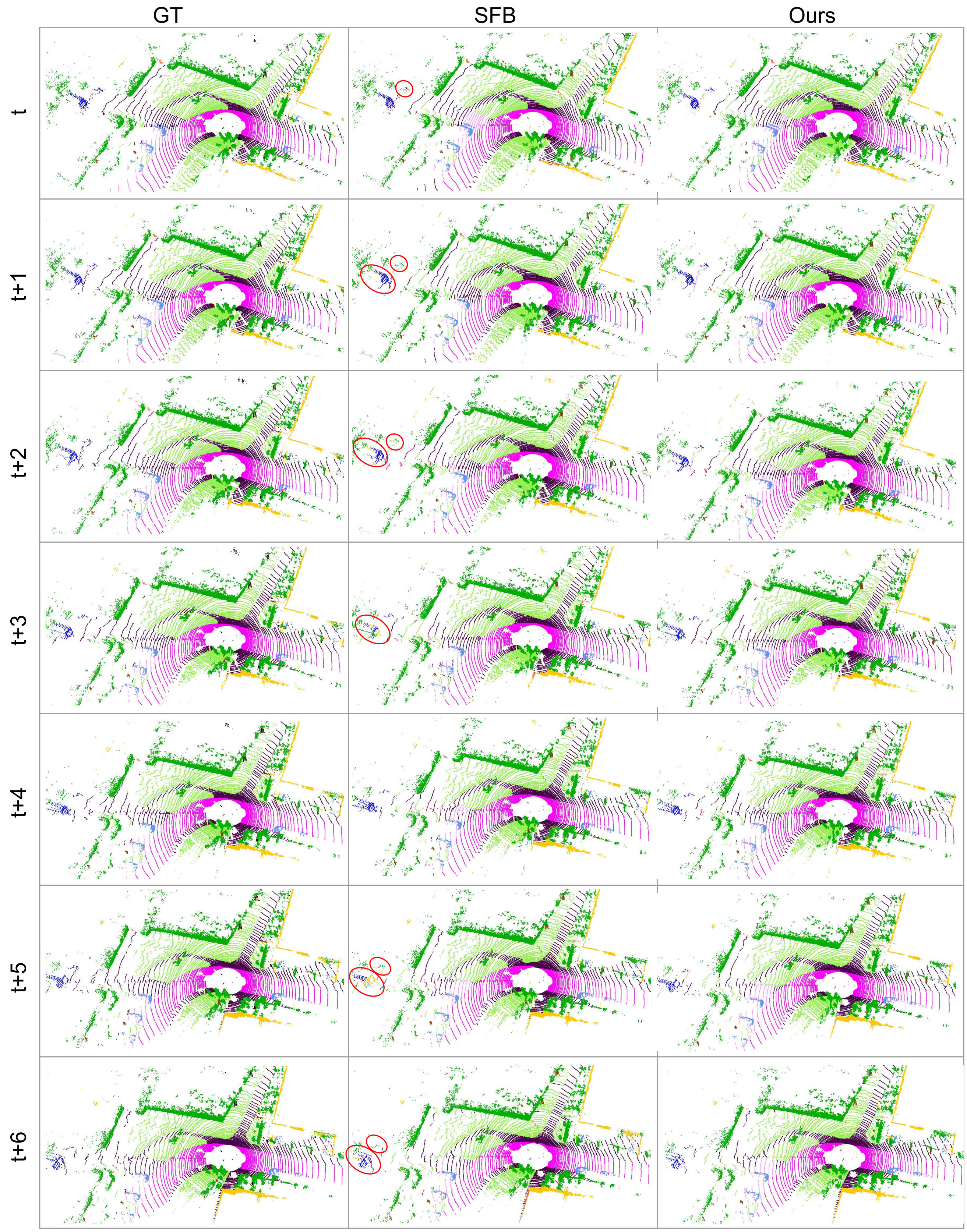

Figure 4. Qualitative comparison with single-frame baseline (SFB). Our approach is able to generate robust segmentation predictions throughout the interval where the SFB produces flickering results. See the vehicle highlighted in red circle.

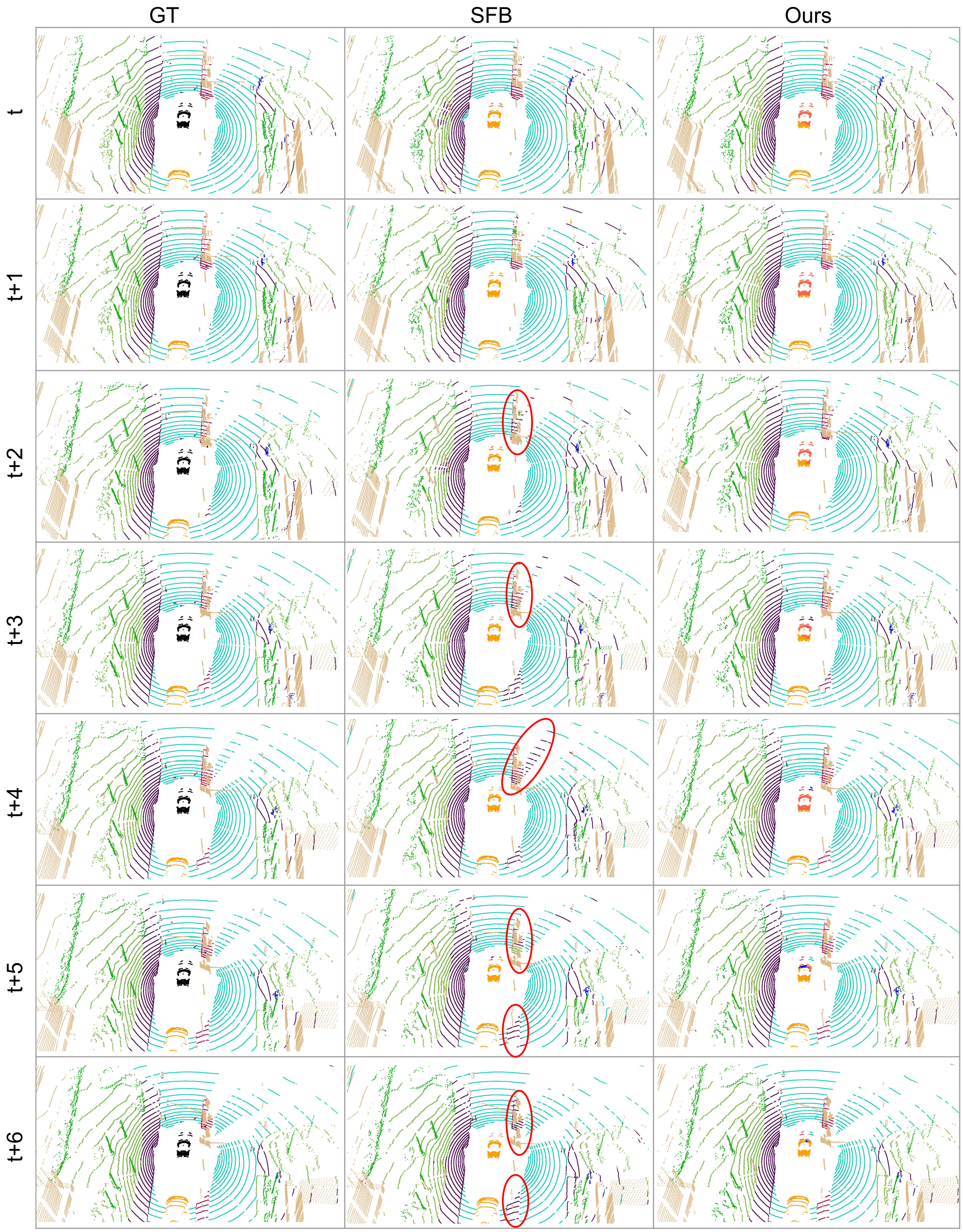

Figure 5. Qualitative comparison with single-frame baseline (SFB). SFB is prone to errors in regions with sparse observations or occlusions, resulting in confusion between sidewalks, roads, and other flat surfaces (highlighted in red). Our approach produces accurate and reliable results.

References

[1] Nicolas Ballas, Li Yao, Chris Pal, and Aaron C. Courville. Delving deeper into convolutional networks for learning video representations. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. 1

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jürgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *2019 IEEE/CVF International Conference on Computer Vision*, pages 9296–9306. IEEE, 2019. 1, 2, 3, 4

[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 *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11621–11631, 2020. 1, 2, 3, 5

[4] Ran Cheng, Ryan Razani, Ehsan Taghavi, Enxu Li, and Bingbing Liu. af2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12547–12556, 2021. 2, 3

[5] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. Salsanext: Fast semantic segmentation of lidar point clouds for autonomous driving. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 655–661, 2020. 3

[6] Fabian Duerr, Mario Pfaller, Hendrik Weigel, and Jürgen Beyerer. Lidar-based recurrent 3d semantic segmentation with temporal memory alignment. In *2020 International Conference on 3D Vision (3DV)*, pages 781–790. IEEE, 2020. 3

[7] Martin Gerdzhev, Ryan Razani, Ehsan Taghavi, and Liu Bingbing. Tornado-net: multiview total variation semantic segmentation with diamond inception module. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9543–9549, 2021. 3

[8] Yuenan Hou, Xinge Zhu, Yuexin Ma, Chen Change Loy, and Yikang Li. Point-to-voxel knowledge distillation for lidar semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8479–8488, 2022. 2, 3, 5

[9] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11108–11117, 2020. 3

[10] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Rangenet++: Fast and accurate lidar semantic segmentation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019. 3

[11] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 3

[12] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII*, pages 685–702. Springer, 2020. 3

[13] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proc. of the IEEE International Conference on Computer Vision*, pages 6411–6420, 2019. 3

[14] Song Wang, Jianke Zhu, and Ruixiang Zhang. Meta-rangeseg: Lidar sequence semantic segmentation using multiple feature aggregation. *IEEE Robotics and Automation Letters*, 7(4):9739–9746, 2022. 3

[15] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. In *European Conference on Computer Vision*, pages 1–19. Springer, 2020. 3

[16] Jianyun Xu, Ruixiang Zhang, Jian Dou, Yushi Zhu, Jie Sun, and Shiliang Pu. Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16024–16033, 2021. 2, 3

[17] Feihu Zhang, Jin Fang, Benjamin Wah, and Philip Torr. Deep fusionnet for point cloud semantic segmentation. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2020. 3

[18] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9601–9610, 2020. 3

[19] Zixiang Zhou, Yang Zhang, and Hassan Foroosh. Panoptic-polarnet: Proposal-free lidar point cloud panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13194–13203, 2021. 4

[20] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9939–9948, 2021. 2, 3, 5