Supplementary Material - STROBE: Streaming Object Detection from LiDAR Packets

Davi Frossard [1,2] **Simon Suo** [1,2] **Sergio Casas** [1,2] **James Tu** [1,2]
Rui Hu [1] **Raquel Urtasun** [1,2]
[1] Uber Advanced Technologies Group [2] University of Toronto
`{frossard, suo, sergio.casas, james.tu, rui.hu, urtasun}@uber.com`

A Implementation Details

A.1 Baselines

In this section, we discuss the details of our baselines, and in particular how we adapt them to the LiDAR packet stream setting. We directly adapt the reference implementation provided by the original authors whenever possible. We found that it is necessary to make minor modification to architecture and training procedures to account for the difference in dataset (PACKETATG4D vs. KITTI) and evaluation setup.

HDNet We augment the input to HDNet to 10 sweeps for better detection performance [1]. In the LiDAR packet stream setting, we adapt HDNet to use our *Regional Convolutions* as opposed to regular 2D convolutions. This modification is necessary, since the increased sparsity in rasterizing just a single packet violates the implicit assumption made by the normalization layers.

PointPillars We make the following modifications to the reference implementation [2]:

- *Voxel Feature Encoding*: We use a single Voxel Feature Encoding (VFE) layer with 8 input features and output dimension of 64. To account for small batch size during training, we also replace BatchNorm with GroupNorm.
- *Multi-Class*: Instead of training separate models for vehicle vs. cyclists and pedestrians as proposed in the original paper, we directly use shared backbone with class-specific headers to output multi-class detection. This ensures the model has similar capacity as other approaches.

PointRCNN We make the following modifications to the reference implementation (https://github.com/sshaoshuai/PointRCNN):

- *Point Resampling*: For PACKETATG4D, since we process the full rectangular region of interest instead of the front view sector in KITTI, we pass in 35,000 points instead 16,384 points to maintain similar point resolution.
- *Model Capacity*: We found that it was critical to increase model capacity to achieve comparable performance. There are two main reasons: 1) the PointNet backbone does not automatically scale with the number of input points, and 2) the original architecture is designed for single class, whereas we perform multi-class detection. In particular, we double the feature dimension for the PointNet backbone, and keep triple the points at each stage of downsampling. We also add an extra layer in the final MLP regression header.
- *Proposal Sampling*: We modify the definition of "near" and "far" used for proposal sampling. In KITTI, it was sufficient to determine the farness based on distance in x-axis, since we are only processing the front view sector. We extend the definition to incorporate y-axis distance as well.
- *Rare Example Mining*: Since cyclists are rare in PACKETATG4D, we found that it was essential explicitly mine for cyclists when sampling RoIs during training.

4th Conference on Robot Learning (CoRL 2020), Cambridge MA, USA.

- *Pedestrian ROI*: When sampling pedestrian RoIs, we use distance based thresholds instead of IoU to be consistent with our evaluation metric for pedestrians.

B Additional Evaluation Details and Results

B.1 History-Aware Detection

One issue in object detection from LiDAR point cloud is that actors can be fully occluded. To infer the true state of the world from partial observation, it is necessary to leverage the history and physical motion constraints. StrObe achieves this through the spatial memory module, which is updated with every new packet. As shown in the manuscript, this is fundamental for accurate detection as it allows the model to use larger context to do detection within the limited bounds of a single packet. This approach also has the benefit of allowing the model to recall objects through occlusion, still producing detections when there are few or no LiDAR points.

So far we have presented results on the common setting in which only actors with at least 1 LiDAR point are considered. Here, we present additional results on the more challenging *history-aware detection* setting, in which the perception model is responsible for all actors with at least 1 LiDAR point in the past 1 second (i.e. 10 sweeps or 100 packets). Similarly, we consider all models at both packet and sweep granularity, and at detection emission time (latency mAP) and observation time (common mAP).

Model	**Packet Stream**						**Full Sweep**					
	Vehicle		Pedestrian		Cyclist		Vehicle		Pedestrian		Cyclist	
	0.5	0.7	0.5	0.3	0.3	0.5	0.5	0.7	0.5	0.3	0.3	0.5
HDNET [3]	70.3	57.9	68.0	60.5	19.0	13.4	75.8	54.5	**78.5**	**68.0**	51.1	31.3
PointPillars [4, 2]	56.8	40.5	47.2	43.4	14.0	5.1	76.7	55.3	70.1	65.7	50.1	31.1
PointRCNN [5]	59.9	53.9	46.6	42.0	23.5	21.4	65.8	51.2	51.5	49.2	28.0	23.7
Our StrObe	**86.7**	**74.0**	**77.2**	**69.0**	**56.2**	**35.7**	**83.3**	**63.2**	74.7	65.8	**58.7**	**37.6**

Table 1: **Latency mAP:** Labels are considered at detection emission times.

Model	**Packet Stream**						**Full Sweep**					
	Vehicle		Pedestrian		Cyclist		Vehicle		Pedestrian		Cyclist	
	0.5	0.7	0.5	0.3	0.3	0.5	0.5	0.7	0.5	0.3	0.3	0.5
HDNET [3]	70.3	58.1	68.1	60.6	19.3	13.7	**86.1**	**73.3**	**82.5**	**72.6**	**66.1**	**43.5**
PointPillars [4, 2]	56.9	40.7	47.3	43.5	14.0	5.1	77.5	64.4	70.4	65.7	50.8	32.6
PointRCNN [5]	60.0	54.0	43.7	42.3	25.8	23.3	73.5	60.6	51.2	49.8	28.4	24.6
Our StrObe	**86.9**	**74.1**	**77.2**	**69.1**	**56.2**	**35.7**	84.4	72.6	75.1	67.1	59.1	39.4

Table 2: **Common mAP**: Labels are considered at their observation times.

The results in Tables 1 and 2 show that our proposed memory module allows StrObe to retrieve detections even in absence of points, as evidenced by the small difference in comparison to the numbers shown in the standard setting of the manuscript. HDNET uses the approach of concatenating 10 sweeps at the input level; while effective, this leads to redundant computation due to the same frame being used several times. Since PointPillars and PointRCNN do not use past observations they are not well suited for this setting.

References

[1] S. Casas, W. Luo, and R. Urtasun. Intentnet: Learning to predict intention from raw sensor data. In *Conference on Robot Learning (CoRL)*, pages 947–956, 2018.

[2] Y. Yan, Y. Mao, and B. Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10): 3337, 2018.

[3] B. Yang, M. Liang, and R. Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In *Conference on Robot Learning (CoRL)*, pages 146–155, 2018.

[4] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019.

[5] Y. Chen, S. Liu, X. Shen, and J. Jia. Fast point r-cnn. In *International Conference on Computer Vision (ICCV)*, pages 9775–9784. IEEE, 2019.