LaserNet: An Efficient Probabilistic 3D Object Detector for Autonomous Driving

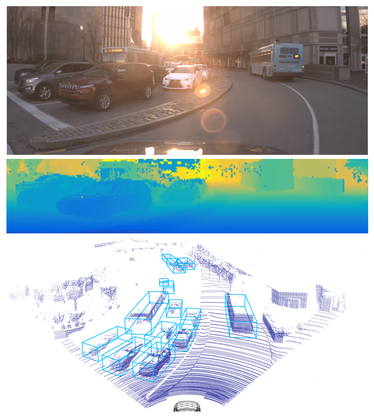
Gregory P. Meyer* gmeyer@uber.com
Ankit Laddha* aladdha@uber.com
Eric Kee ekee@uber.com
Carlos Vallespi-Gonzalez cvallespi@uber.com
Carl K. Wellington cwellington@uber.com

Uber Advanced Technologies Group

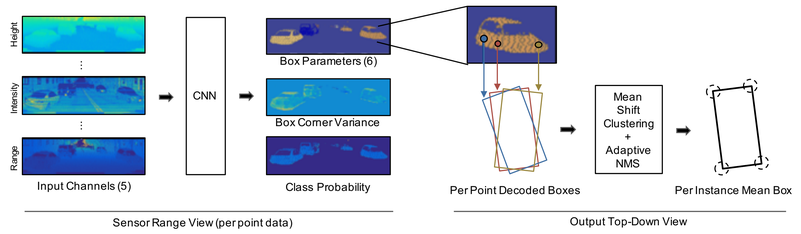
CVPR LONG BEACH CALIFORNIA June 16-20, 2019

Introduction

- 3D object detection is a key capability for autonomous driving.
- LaserNet is an efficient and probabilistic 3D object detector based on LiDAR.
- LiDAR is inherently dense from the sensor's point of view but sparse when projected into 3D space.
- The efficiency of our detector is due to operating in the dense range view instead of a sparse top-down view.
- Our method captures the uncertainty of a detection by predicting the distribution of bounding box corners.
- On a large benchmark dataset, LaserNet achieves state-of-the-art detection performance with significantly lower runtime.



Overview



- Our method uses the inherent **range view** representation of the LiDAR.
- A **fully-convolutional** neural network produces a set of predictions for each point.
- For each point, we predict a set of class probabilities, and we regress a probability distribution over bounding boxes in the **top-down view**.
- These per-point predictions are combined through **mean shift clustering**.
- The entire detector is trained **end-to-end** with the loss defined on the box corners.
- At inference, a novel **adaptive non-maximum suppression** algorithm is utilized.

Results

- Our approach is evaluated and compared against previous methods on two datasets:
  - The ATG4D object detection dataset (1.2 million training sweeps)
  - The KITTI object detection benchmark (7,481 training sweeps)
- Following the KITTI benchmark, only detections within the front 90° field of view of the sensor and up to 70 meters are considered.

Table 1: BEV Object Detection Performance on ATG4D

Method	Input	Vehicle $AP_{0.7}$	Bike $AP_{0.5}$	Pedestrian $AP_{0.5}$
LaserNet (Ours)	LiDAR	**85.34**	**61.93**	**80.37**
PIXOR	LiDAR	80.99	-	-
PIXOR++	LiDAR	82.63	-	-
ContFuse	LiDAR	83.13	57.27	73.51
ContFuse	LiDAR+RGB	85.17	61.13	76.84

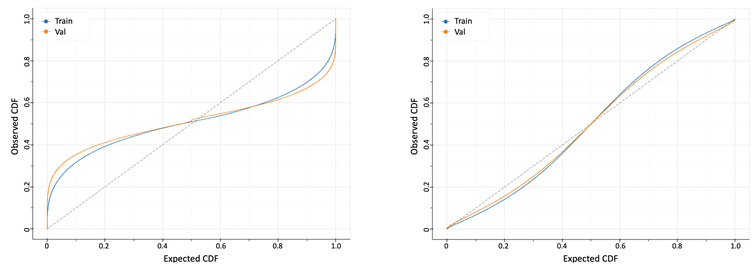
Table 2: Ablation Study on ATG4D

Predicted Distribution	Image Spacing	Mean Shift	IoU Threshold	NMS Type	Vehicle $AP_{0.7}$
Mean-only	Laser	Yes	0.1	Hard	77.05
Unimodal	Uniform	Yes	0.1	Hard	79.14
Unimodal	Laser	No	0.1	Hard	80.22
Unimodal	Laser	Yes	0.1	Hard	80.92
Multimodal	Laser	Yes	0.1	Hard	81.80
Multimodal	Laser	Yes	N/A	Soft	84.43
Multimodal	Laser	Yes	Adaptive	Hard	83.68
Multimodal	Laser	Yes	Adaptive	Soft	**85.34**

Table 3: BEV Object Detection Performance and Runtime on KITTI

Method	Input	Vehicle $AP_{0.7}$			Runtime	
		Easy	Moderate	Hard	Forward Pass (ms)	Total (ms)
LaserNet (Ours)	LiDAR	78.25	73.77	66.47	**12**	**30**
PIXOR	LiDAR	81.70	77.05	72.95	35	62
PIXOR++	LiDAR	**89.38**	83.70	**77.97**	35	62
VoxelNet	LiDAR	89.35	79.26	77.39	190	225
MV3D	LiDAR+RGB	86.02	76.90	68.49	-	360
AVOD	LiDAR+RGB	88.53	83.79	77.90	80	100
F-PointNet	LiDAR+RGB	88.70	84.00	75.33	-	170
ContFuse	LiDAR+RGB	88.81	**85.83**	77.33	60	-

- On the small dataset, our approach under-performs compared to state-of-the-art methods, but on a significantly larger dataset, our method out-performs the previous work.
- Runtime performance is equally important for the purpose of autonomous driving, and our method is twice as fast as the fastest state-of-the-art method.
- Predicting a probability distribution over bounding boxes is a key aspect of our approach as shown in the ablation study.
- We suspect the KITTI training set does not contain enough examples to accurately learn the distribution (as shown below), explaining the difference in performance.



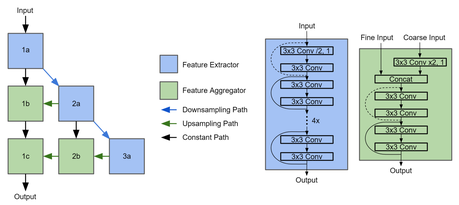
(a) Calibration on KITTI (b) Calibration on ATG4D

Figure 1: Plots showing the calibration of the predicted distribution over bounding boxes on the train and validation sets. A perfectly calibrated distribution corresponds to a line with unit slope (dashed line in the plots).

Method

1 Network Architecture

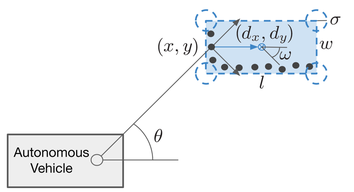
- The range image contains objects that vary from several thousand points to a single point.
- A deep layer aggregation network is used to effectively extract and combine multi-scale features.



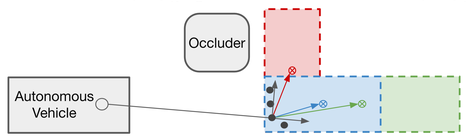
2 Predictions

- The network is trained to predict a set of class probabilities for each point in the image.
- Given a point is on an object, the network predicts a distribution over bounding boxes.
- Instead of directly estimating the box corners, the network predicts a center, orientation, and dimensions relative to the point.

$$\boldsymbol{b}_c = [x, y]^T + \boldsymbol{R}_\theta \, [d_x, d_y]^T$$
$$\phi = \theta + \text{atan2}\,(\omega_y, \omega_x)$$
$$\boldsymbol{b} = \left[\boldsymbol{b}_c + \frac{1}{2}\boldsymbol{R}_\phi \, [l, w]^T, \cdots, \boldsymbol{b}_c + \frac{1}{2}\boldsymbol{R}_\phi \, [-l, -w]^T \right]$$
$$s = \log \sigma$$



- The distribution can be multimodal when the object is only partially observed.
- We model the multimodal probability distribution with a mixture model.
- The network is trained to predict a set of means, $\{d_{x,k}, d_{y,k}, \omega_{x,k}, \omega_{y,k}, l_k, w_k\}_{k=1}^K$, with corresponding variances, $\{s_k\}_{k=1}^K$, and mixture weights, $\{\alpha_k\}_{k=1}^K$.



3 Mean Shift Clustering

- Per-point predictions are combined thorough mean shift clustering.
- For efficiency, mean shift is performed over box centers, and the top-down view is discretized into bins of size Δx by Δy.
- Predictions that fall into the same bin are averaged, and the means are iteratively updated based on neighboring bins.

$$\boldsymbol{m}_i \leftarrow \frac{\sum_{j \in i \cup N(i)} K_{i,j} \, (\boldsymbol{m}_j \cdot |S_j|)}{\sum_{j \in i \cup N(i)} K_{i,j} |S_j|}$$
$$K_{i,j} = \exp\left(-\frac{\|\boldsymbol{m}_i - \boldsymbol{m}_j\|^2}{\Delta x^2 + \Delta y^2} \right)$$
$$\hat{\boldsymbol{b}}_i = \frac{\sum_{j \in S_i} (1/\sigma_j^2) \boldsymbol{b}_j}{\sum_{j \in S_i} (1/\sigma_j^2)} \qquad \hat{\sigma}_i^2 = \left(\sum_{j \in S_i} \frac{1}{\sigma_j^2} \right)^{-1}$$

4 End-to-end Training

- For each point in the image, we use focal loss to learn the class probabilities.
- For each point on an object, we use the hindsight loss to learn the mixture model.
- The component that is closest to the ground-truth is updated by penalizing the negative log likelihood.
- The mixture weights are learned using the cross entropy loss.

Classification Loss

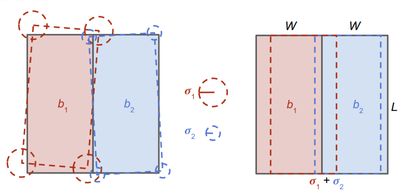
$$\mathcal{L}_{\text{cls}} = -\sum_{i=1} y_i (1 - p_i)^\gamma \log p_i \quad \} \text{ Focal Loss}$$

Regression Loss

$$\mathcal{L}_{\text{reg}} = \frac{1}{2\hat{\sigma}_{k^*}} \|\hat{\boldsymbol{b}}_{k^*} - \boldsymbol{b}_{\text{gt}}\|_1 + \frac{1}{2} \log \hat{\sigma}_{k^*} \quad \} \text{ Negative Log Likelihood}$$
$$k^* = \arg\min_k \|\hat{\boldsymbol{b}}_k - \boldsymbol{b}_{\text{gt}}\|$$
$$\mathcal{L}_{\text{mix}} = -\sum_{k=1} \mathbf{1}_{k=k^*} \log \alpha_i \quad \} \text{ Cross Entropy}$$

5 Adaptive Non-Maximum Suppression

- Non-maximum suppression is performed to remove boxes with an intersection over union (IoU) greater than a threshold.
- Due to the uncertainty in the predictions, some amount of overlap is expected.
- For each pair of boxes, we adapt the IoU threshold based on their predicted standard deviations.



$$t = \begin{cases} \frac{\sigma_1 + \sigma_2}{2w - \sigma_1 - \sigma_2} & \sigma_1 + \sigma_2 < w \\ 1 & \text{otherwise} \end{cases}$$