Applied & Educational Psychology (2021)
Clausius Scientific Press, Canada

DOI: 10.23977/appep.2021.020113
ISSN 2523-5842 Vol. 2 Num. 1

End-to-End Multi-Sensor Fusion for 3d Object Detection in Lidar Point Clouds

Kanrun Huang
Nauto, Inc., California, 94306, United States, China

Keywords: Object detection, Deep learning, Sensor fusion

Abstract: In this paper, we propose a new method to fuse camera and lidar point clouds at the same time that outperforms the state-of-the-art network at long range and challenge conditions like poor weather and highly reflective surfaces. Point clouds from LiDAR tend to be sparse and highly variable, especially at long range and poor weather, which may cause 3D object detectors to fail to detect small but important objects (pedestrians, traffic signs, cones, etc.). Our neural network model does not need labels of objects in the 2D images. The neural network model we proposed is evaluated both on the Waymo Open Dataset and on the KITTI dataset [1] and we demonstrate that our model achieves state-of-the-art accuracy on 3D object detection.

1. Introduction

Understanding the 3D environment is one of the core capabilities required for autonomous driving. Lidar sensors usually are used for the 3D object detection task due to its accuracy in measuring the distance to the object while being robust to most lighting conditions. In addition to LiDAR, autonomous driving vehicles are also commonly equipped with high resolution cameras, which can also be used to generate 3D point cloud. At range, LiDAR measurements become increasingly sparse, so incorporating high resolution image data could improve performance on distant objects.

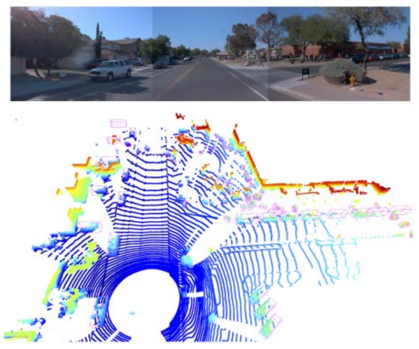

Fig.1 3d Object Detection Results from Our Proposed Method. Our Approach Utilizes Both 2d Images (Top) and 3d Lidar Points (Bottom).

Convolutional neural networks (CNNs) have produced state-of-the-art results on 3D object detection from LiDAR data. Typically, previous work discretizes the LiDAR points into 3D voxels and performs convolutions in the bird's eye view (BEV). Only a few methods utilize the native range view (RV) of the LiDAR sensor.

Both BEV view and RV have its own advantages.

BEV advantage: One benefit of operating in the BEV space is that it preserves the metric space, i.e., object sizes remain constant with respect to distance from the sensor. This allows models to leverage prior information about the size of objects during training. On the other hand, as the point cloud becomes sparser or as measurements get farther away from the sensor, the number of points available for each voxel embedding becomes more limited.

perspective range-image advantage:

The RV representation has been shown to perform well at longer ranges where the point cloud becomes very sparse, and especially on small objects. By operating on the "dense" range-image, this representation can also be very computationally efficient. Due to the perspective nature, object shapes are not distance-invariant and objects may overlap heavily with each other in a cluttered scene, which leads to difficulty at object segmentation. Moreover, we inferred perspective range-image from RGB images from camera to densify the pointcloud.

In this paper, we make two major contributions.(1) We present a multi-task learning framework to learn the probabilistic 3D object detector combing the RV image and BEV view images input at the same time in an end-to-end training fashion. (2) Our approach is more accurate because we also fuse camera images to generate point cloud.

2. Related Work.

2.1 2D Object Detection:

Starting from the R-CNN detector proposed by Girshick et al., researchers have developed many modern detector architectures based on Convolutional Neural Networks (CNN). Among them, there are two representative branches: two-stage detectors and single-stage detectors. The seminal Faster RCNN paper [10] proposes a two-stage detector system, consisting of a Region Proposal Network (RPN) that produces candidate object proposals and a second stage network, which processes these proposals to predict object classes and regress bounding boxes. On the single-stage detector front, SSD by Liu et al. simultaneously classifies which anchor boxes among a dense set contain objects of interest, and regresses their dimensions. Single-stage detectors are usually more efficient than two-stage detectors in terms of inference time, but they achieve slightly lower accuracy compared to their two-stage counterparts on the public benchmarks such as MSCOCO, especially on smaller objects. Recently Lin et al. demonstrated that using the focal loss function on a single-stage detector can lead to superior performance than two-stage methods, in terms of both accuracy and inference time.

2.2 3D Object Detection

A popular paradigm for processing a point cloud produced by LiDAR is to project it in birds-eye view (BEV) and transform it into a multi-channel 2D pseudo image, which can then be processed by a 2D CNN architecture for both 2D and 3D object detection. The transformation process is usually hand-crafted, some representative works include Vote3D, Vote3Deep, 3DFCN, AVOD [3], PIXOR and Complex YOLO. VoxelNet by Zhou et al. [2] divides the point cloud into a 3D voxel grid (i.e. voxels) and uses a PointNet-like network to learn an embedding of the points inside each voxel. PointPillars [7] builds on the idea of VoxelNet to encode the points feature on pillars (i.e.

vertical columns). Shi et al. propose a PointRCNN model that utilizes a two-stage pipeline, in which the first stage produces 3D bounding box proposals and the second stage refines the canonical 3D boxes. Perspective view is another widely used representation for LiDAR. Along this line of research, some representative works are VeloFCN and LaserNet.

3. Proposed Method

In the following sections, we describe the architecture of our model to fuse RGB images, perspective images, and BEV images to jointly perform 3D object detection.

3.1 Input Representation

autonomous-driving vehicles usually utilize multi-modal sensors to collect data. The input to our model 3D data projected to 2D perspective image and 2D BEV image from a LiDAR, and 2D images from a RGB camera.

For each measurement, the sensor provides a range r, reflectance e, azimuth angle θ of the sensor, and elevation angle φ of the laser that generated the return. The 3D position of the measurement can be computed as follows: computed as follows:

$$\mathcal{p} = \begin{bmatrix} r\cos\emptyset\cos\theta \\ r\cos\emptyset\sin\theta \\ r\sin\theta \end{bmatrix} \quad (1)$$

where $\mathcal{p}$ is ordinarily referred to as a LiDAR point. As in [18], we form an RV image by mapping lasers to rows and discretizing the azimuth angle into columns. For each cell in the image that contains a measurement, we populate a set of channels with the LiDAR point's range r, height z, azimuth angle θ, and intensity e, as well as a flag indicating if the cell is occupied. The result is a five channel LiDAR image. The camera captures a RGB image which covers the front 90◦ horizontal and the full 30◦ vertical field of view of the LiDAR image. We crop both the RGB and LiDAR image to align with each other and input them to a depth completion backbone model.

BEV view Voxelization

Voxelization can be used to generate a BEV view from the point cloud. Voxelization divides a point cloud into an evenly spaced grid of voxels, then generates a many-to-one mapping between 3D points and their respective voxels. We take the same algorithm of voxelization as VoxelNet [5], hard voxelization as a two stage process: grouping and sampling. Given a point cloud P = { $\mathcal{p}_1, \ldots, \mathcal{p}_n$ }, the process assigns N points to a buffer with size K × T × F, where K is the maximum number of voxels, T is the maximum number of points in a voxel and F represents the feature dimension. In the grouping stage, points $\{\mathcal{p}_i\}$ are assigned to voxels $\{ v_j\}$ based on their spatial coordinates. Since a voxel may be assigned more points than its fixed point capacity T allows, the sampling stage sub-samples a fixed T number of points from each voxel. If the point cloud produces more voxels than the fixed voxel capacity K, the voxels are sub-sampled.

3.2 Point-Wise Feature Fusion:

We apply point-wise feature fusion between the convolutional feature maps of LiDAR and image streams. The fusion is directed from image steam to LiDAR steam to augment the lidar point cloud feature BEV features with information richness of image features.

To fuse multi-sensor convolutional feature maps, we need to find the pixel-wise correspondence between the two sensors. Inspired by Fast r-cnn, we use continuous fusion to establish dense and accurate correspondences between the image and BEV feature maps. For each pixel in the BEV feature map, we find its nearest LiDAR point and project the point onto the image feature map to retrieve the corresponding image feature. We compute the distance between the BEV pixel and LiDAR point as the geometric feature. Both image feature and geometric feature are passed as input into a Multi-Layer Perceptron (MLP) and the output is fused to BEV feature maps by element-wise addition.

3.3 Network Architecture

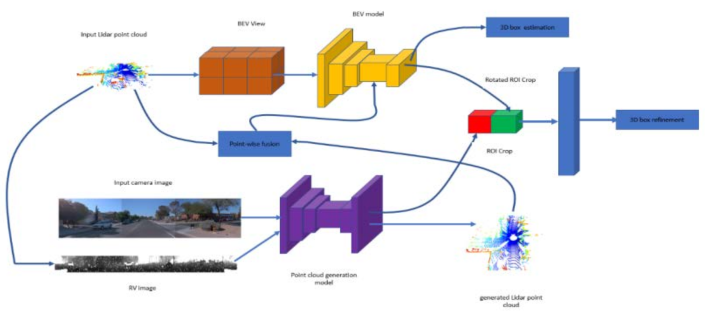

Fig.2 3d Object Detection Model.

Image:

for depth perspective image and RGB image, we use 2D fully convolutional residual network as feature extractor. we use a ResNet-18 architecture until the fourth residual block. Each block contains 2 residual layers with the number of feature maps increasing from 64 to 512 linearly.

Lidar:

For the LiDAR stream, we use a customized residual network which is deeper and thinner than ResNet-18 for a better trade-off between speed and accuracy.

3.4 Predictions and Training

In the previous work Focal loss for dense object detection, the model is trained to predict a set of class probabilities and a set of bounding boxes for each point in the LiDAR image. Since the model classifies LiDAR points as vehicle, bike, pedestrian, or background, it already performs 3D semantic segmentation to some extent. To provide more information to downstream components in a self-driving system, we increase the number of classes to distinguish between background and road as well as bicycles and motorcycles. The training procedure is mostly unchanged from Focal loss for dense object detection. We simply add the additional classes to the classification loss, and we do not modify the regression loss. Although the loss is applied at each point in the LiDAR image, the parameters of the auxiliary network can be updated by backpropagating the loss through the projected image features. It is important to note that the image feature extractor requires no additional supervision; therefore, no supplemental 2D image labels are necessary.

4. Experiments

To investigate the effectiveness of our proposed network, we evaluate it on the KITTI 3D object detection benchmarks and Waymo Open Dataset benchmarks.

4.1 Object Detection on Kitti

Dataset and metric: KITTI’s object detection dataset has 7,481 training frames and 7,518 frames for testing, where we split it to 3770 training samples and 3711 validation samples . To conduct evaluation on the test set with the KITTI official test server, the model is trained with 80% of all available training and validation data and the remaining 20% data is used for validation.

We evaluate our approach on vehicles class. All results are evaluated by the mean average precision with a rotated IoU threshold 0.7 for cars and 0.5 for pedestrian and cyclists. The mean average precisions on the test set are calculated with 40 recall positions on the official KITTI test server. The results on the validation set in Table 2 are calculated with 11 recall positions to compare with the results by the previous works. For multi-task training, we also leverage the dense depth labels from the intersection of KITTI’s depth completion and object detection datasets. KITTI’s detection metric is defined as Average Precision (AP) averaged over 11 points on the Precision-Recall (PR) curve.

4.2 Object Detection on Waymo Open Dataset

Dataset and metric: To futher prove the effectiveness of our proposed model, we evaluate it on the newly released Waymo Open Dataset. Waymo Open Dataset is a large-scale dataset recently released for benchmarking object detection algorithms at industrial production level. The dataset provides information collected from a set of sensors on an autonomous vehicle, including multiple LiDARs and cameras. It captures multiple major cities in the U.S., under a variety of weather conditions and across different times of the day. The dataset provides a total number of 1000 sequences. Specifically, the training split consists of 798 sequences of 20s duration each, sampled at 10Hz, containing 4.81M vehicle and 2.22M pedestrian boxes. The validation split consists of 202 sequences with the same duration and sampling frequency, containing 1.25M vehicle and 539K pedestrian boxes. The effective annotation radius is 75m for all object classes. For our experiments, we evaluate both 3D and BEV object detection metrics for vehicles.

Evaluation Metrics. We evaluate models on the standard average precision (AP) metric for both 7-degree-of-freedom(DOF) 3D boxes and 5-DOF BEV boxes, using intersection over union (IoU) thresholds of 0.7 for vehicles and 0.5 for pedestrians, as recommended on the dataset official website.

We adopt the official released evaluation tools for evaluating our method, where the mean average precision (mAP) and the mean average precision weighted by heading (mAPH) are used for evaluation. The rotated IoU threshold is set as 0.7 for vehicle. The test data are split in two ways. The first way is based on objects’ different distances to the sensor: $0 - 30$m, $30 - 50$m and > 50m. The second way is to split the data into two difficulty levels, where the LEVEL 1 denotes the ground-truth objects with more than 5 inside points while the LEVEL 2 denotes the ground-truth objects with at least 1 inside points.

Implementation details: We train the model on a 4 GPU machine with a total batch size of 16 frames. We set the initial learning rate to 0.001 for Adam optimizer and decay it after 30 and 45 epochs respectively. The training ends after 50 epochs.

Evaluation results:

Table 1 Evaluation results on the testing set of KITTI 3D object detection benchmark (car). We compare with previously published detectors on the leaderboard ranked by Average Precision (AP) in the moderate setting.

Group	*Input Data*		*3D AP (%)*		
	Lidar	*Image*	*easy*	*mod.*	*hard*
MV3D[2]	*X*	*X*	*71.09*	*62.35*	*55.12*
AVOD[3]	*X*	*X*	*73.59*	*65.78*	*58.38*
ContFuse[4]	*X*	*X*	*82.54*	*66.22*	*64.04*

MV3D[2]	*X*		*66.77*	*52.73*	*51.31*
VoxelNet[5]	*X*		*77.49*	*65.11*	*57.73*
SECOND[6]	*X*		*83.13*	*73.66*	*66.20*
Ours	*X*	*X*	*86.81*	*76.75*	*68.41*

Table 2 Evaluation results on the testing set of Waymo Open Dataset 3D object detection benchmark (car). We compare with previously published detectors on the leaderboard ranked by Average Precision (AP) in the moderate setting.

Group	*Input Data*		*3D AP (%)*		
	Lidar	*Image*	*easy*	*mod.*	*hard*
MV3D[2]	*X*	*X*	*74.97*	*63.63*	*54.00*
AVOD[3]	*X*	*X*	*83.07*	*71.76*	*65.73*
ContFuse[4]	*X*	*X*	*83.68*	*68.78*	*61.67*
MV3D[2]	*X*	*X*	*66.77*	*52.73*	*51.31*
PointPillars[7]	*X*		*82.58*	*74.31*	*68.99*
SECOND[6]	*X*		*84.65*	*75.96*	*68.71*
Ours	*X*	*X*	*89.23*	*77.78*	*69.54*

We compare our approach with previously published state-of-the-art detectors in Table 1 and Table 2, and show that our approach outperforms competitors by a large margin in all 3D detection tasks. In the most challenging 3D detection task (as it requires 0.7 3D IoU), we show a large gain over competitors. We surpass the best detector SECOND and outperform the previously best multi-sensor detector AVOD.

5. Conclusion

We have proposed a multi-task multi-sensor detection model that reasons about 3D object detection and depth completion. Point-wise is applied to achieve full multi-sensor fusion, while multi-task learning provides additional map prior and geometric cues enabling better representation learning and denser feature fusion. We validate the proposed method on KITTI [8] and Waymo Open Dataset[29] benchmarks, and surpass the state-of-the-art in all detection tasks by a large margin.

References

[1] Geiger, Andreas, Philip Lenz, Raquel Urtasun(2012). Are we ready for autonomous driving? the kitti vision benchmark suite. 2012 IEEE conference on computer vision and pattern recognition, pp. 3354-3361.

[2] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, Tian Xia(2017). Multi-view 3d object detection network for autonomous driving. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 1907-1915.

[3] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L. Waslander(2018). Joint 3D proposal generation and object detection from view aggregation. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1-8.

[4] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun(2018). Deep continuous fusion for multi-sensor 3D object detection. In Proceedings of the European Conference on Computer Vision (ECCV), pp. 641-656.

[5] Yin Zhou, Oncel Tuzel(2018). Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4490-4499.

[6] Yan Yan, Yuxing Mao, Bo Li((2018)). "Second: Sparsely embedded convolutional detection." Sensors 18, no. 10 : pp. 3337.

[7] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, Oscar Beijbom(2019). Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12697-12705.