3D object detection from arbitrary camera rigs

Ayush Rakesh Baid*
School of Interactive Computing
Georgia Institute of Technology
Atlanta, USA
abaid@gatech.edu

Nitish Rajnish Sontakke*
School of Interactive Computing
Georgia Institute of Technology
Atlanta, USA
nitishsontakke@gatech.edu

James Hays
School of Interactive Computing
Georgia Institute of Technology
Atlanta, USA
hays@gatech.edu

***Abstract*—Autonomous vehicles, or self-driving cars, as they are commonly referred to, are generally equipped with a wide array of sensors including, but not limited to cameras, RADAR, and Light detection and ranging (LiDAR). One of the main uses of data collected using these various sensor modalities is detecting other vehicles on the road, pedestrians, stop signs, obstacles and other objects of interest. These task comprise a small subset of the broad domain of 3D object detection. Multiple methods have been proposed to solve this task, which involve LiDAR-only and image-only approaches, as well as methods that involve fusing both. In the current paper, we address this particular task by exclusively making use of image data without any inherent depth information, obtained from cameras mounted at various locations on the vehicle. We develop an RGB-only network to perform 3D object detection, and then try to improve the performance by using LiDAR only during training.**

***Index Terms*—3D object detection, autonomous vehicles.**

I. Introduction

The need for autonomous vehicles (AVs) arises from concerns of safety and reliability [1]. It can also benefit individuals with disabilities that may preclude them from driving cars themselves, by providing them with mobility options that do not involve relying on other people while also being cost effective. It is also a compelling computer science problem as it will bring us one step closer to artificial general intelligence. It is an extremely complex task because of the inherently dynamic nature of self driving in addition to the wide variety of rules, regulations, traffic patterns and other environmental variables that affect driving decisions. There is also no margin for error. Accuracy is therefore paramount, and multiple forms of redundancies need to be incorporated in order to account for worst-case scenarios. Autonomous vehicles therefore employ a diverse set of sensors such as cameras, LiDAR, and RADAR [2].

One of the key components of the autonomous vehicle software stack involves object detection. As mentioned in the previous paragraph, there are various forms of input that are available to perform this task. There are methods that rely solely on LiDAR data, which is typically in the form of point clouds, methods that try to perform object detection using just RGB image data, and methods that try to combine data from both these sources in order to increase performance. In the next section, we will take a look at the LiDAR-only and camera-only methods that have been proposed for object detection.

* denotes equal contribution

II. Related work

A. LiDAR based methods

LiDAR has been the dominant sensor input for 3D object detections by a large margin, and boasts of top performance on various public benchmarks. Modern autonomous vehicles employ multiple LiDAR scanners, relying heavily on them because of the easy availability of depth information that is absent in case of camera images.

Zhou et al. [3] kickstarted the excellent performance in 3D detection using VoxelNet by using a learned feature encoder instead of hand-crafted transformations of LiDAR point clouds. They divide the 3D space into voxels, and use 3D convolutions to perform detection in an end-to-end fashion. There are multiple works which build upon the ideas introduces in VoxelNet.

PointPillars [4] identifies the bottleneck that 3D convolutions pose in methods like VoxelNet [3], and aims to improve upon that by instead discretizing the point cloud in only the x-y plane, enabling use of 2D convolutions. This allows the authors to achieve a significant improvement in speed over methods that use 3D convolutions. It discretizes point clouds in the x-y plane to form 'pillars' of points. The authors then use a simplified PointNet [5] to transform this data into a representation suitable for use with 2D Convolutional Neural Networks (CNNs), which they then use to extract higher-level features. These features are then used as input to a detection head to predict 3D bounding boxes for objects.

Qi et al. [6] introduce a very different processing philosophy. Instead of discretizing the input point cloud 3D space into voxels or pillars of points, they directly operate on 3D points by associating a vote for each LiDAR point and borrowing ideas from classical computer vision such as the Hough transform and voting.

Autonomous vehicles, however, generate data that is inherently dynamic in nature. We need to be able to handle not just static point clouds, but sequences of them. FlowNet3D [8] addresses this very problem of scene flow, which tries to predict the motion of points in 3D space. The authors propose a novel end-to-end trainable architecture that looks at point clouds at consecutive time steps to predict flow vectors. While FlowNet3D works by considering consecutive frames of point clouds and estimating flow vectors between them, MeteorNet [9] generalizes to longer sequences of point cloud data.

Fig. 1. Deep3DBox [7], one of the methods that performs 3D object detection using a single RGB image.

B. Camera-based methods

3D object detection on RGB images is a challenging task owing to the lack of depth information. There have been improvements in single-image based methods but they still fall behind single-image 2D object detection and LiDAR based 3D detection. Mono3D [10] uses ground plane prior and constructs proposals in 3D which are projected to 2D images. Deep3DBox [7] first estimates the object orientation and dimensions, and then estimates the 3D bounding box by optimizing in the constraint provided by 2D bounding boxes.

Pseudo-LiDAR [11] is a seminal work where the authors argue that the huge performance gap between object detection from camera images and LiDAR scans is not the data quality but the data representation. They argue that advances in monocular and stereo depth estimation has made depth maps pretty accurate, so lack of depth cues in 2D images is not the main source of performance degradation. They argue that sequence of 2D convolutions is a bad choice for performing 3D detection because the neighborhoods in 2D images can have wildly different depth images. As a result, they transform the 2D images into pseudo-LiDAR point clouds using depth estimation, and use 3D detection techniques on the transformed representation. The performance jump from 22% to 74% is a huge one and has influenced a lot of subsequent work.

Roddick et al. [12] use a similar reasoning about the flaws of perspective image-based representation. However, their choice of the intermediate representation is different: they directly map the input images to a birds-eye-view based feature space. In contrast to [11], they do not rely on explicit depth recovery but argue that their network directly learns to map image pixels to appropriate gird locations in the BEV frame.

Philion and Fidler [13] propose Lift-Splat-Shoot, which is a segmentation and motion planning network which works on arbitrary camera rigs. But it follows similar philosophy of intermediate representation in BEV. The first half of their architecture proposes features and a probabilistic depth vector for each pixel in the input image. This is in contrast with OFT [12], and the features are then smeared over the BEV frame according to the probabilistic depth. Although they use the BEV representation for a different end goal, the similarities with OFT [12] provides an alternate feature encoder which can work well for 3D detection.

More recently proposed methods such as MonoDIS [14] aim to tackle the problem of monocular 3D object detection by balancing the contribution of various parameters in the loss functions, which typically vary significantly in magnitudes and require careful weighting, while also precluding stage-wise training. They achieve this through a novel transformation that 'disentangles' these dependencies between the parameters, allowing them to train their network end-to-end, while also retaining the overall nature of the loss function. They further propose a novel signed Intersection-over-Union (IoU) based metric that they use to demonstrate improved 2D detection performance. They also use self-supervision to learn detection confidence score prediction for 3D bounding boxes.

Another recent method, CenterNet [15], represents objects using center points of their bounding boxes. The authors use a keypoint estimator to detect the center point of an object bounding box, and then regress to the object's properties such as size, 3D location, orientation, and pose. Their method does not require non-maximal suppresion and is end-to-end differentiable.

C. LiDAR + Camera Sensor Fusion Methods

Fusing LiDAR and RGB data has also been a common approach to achieve cutting-edge performance. MV-3D [16] uses BEV representations produced from LiDAR scans to generate 3D object proposals and projects them to other sensor data. AVOD [17] generates proposals on the fused sensor data. These methods are more accurate than methods that rely solely on a single sensor modality as they have strictly more information than the latter.

The LiDARs used in autonomous vehicles, however, tend to be extremely expensive [2]. Therefore, it is worthwhile considering purely camera-based methods as well, since cameras tend to be significantly cheaper and more easily available.

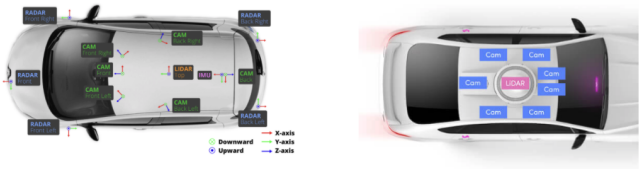

Fig. 2. Different camera rig configurations. The left part of the image depicts a car with 6 cameras which was used to create the nuScenes dataset [18], whereas the camera rig for the car in the right half of the image contains 7 cameras, and was used to create the Lyft level 5 database [19]

III. Proposed Method

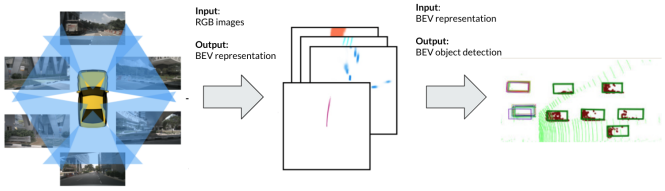

Fig. 3. Overview of our proposed method. Source: [13]

The main challenge posed by monocular 3D object detection is the inherent lack of depth information. Even for autonomous vehicles that have camera rigs, it is difficult to obtain depth information using multi-view stereo owing to the lack of significant overlap between the camera frustums. Different cars also have different camera rig configurations. We build our monocular-image based network on the Lift-Splat-Shoot [13] architecture. This model performs the geometry transform as suggested by pseudo-LiDAR [11] and confirmed by Ma et al. [20]. We call our model Lift-Splat-Detect (LSD).

An overview of the proposed method is provided in Figure 3. The 'Lift' component of the network outputs the categorical distribution over depth and a context vector to transform each input image from its local 2D coordinate system to a shared 3D coordinate system. The 'Splat' component performs the geometry transformation using camera extrinsics and intrinsics to map each per-image frustum-shaped point cloud generated by the 'Lift' component onto the BEV plane. The 'Detect' network works on this intermediate representation to perform object detection. We think this will be more powerful than working on single-3D images and will not require explicit depth information like Pseudo-LiDAR [11]. Using this approach has the added benefit of robustness, as it works even in cases of camera dropout and can be generalized to arbitrary camera rigs. The network architecture for our proposed monocular-only network is presented in Figure 4.

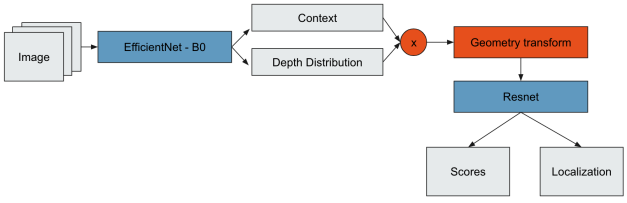

Fig. 4. Our proposed pipeline, Lift-Splat-Detect (LSD).

Recent methods for image-based object detection using stereo for depth estimation show performance at-par with common LiDAR based methods [21]. This suggest that accurate depth information is highly correlated with the detection performance. To understand the effect of oracle depth on our network architecture, we propose a sensor-fusion between RGB images and LiDAR. The depth (z-coordinate) from LiDAR is encoded as a sparse single-channel range image. This new network is called LSD-LiDAR, and the network architecture is presented in Figure 6.

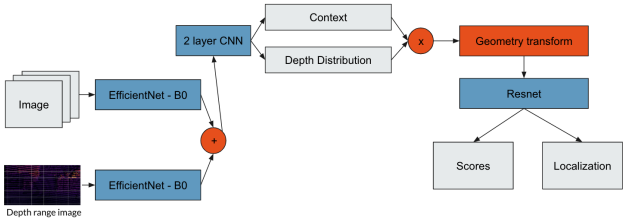

Fig. 5. LSD-LiDAR: Lift-Splat-Detect, with access to LiDAR depth information.

Finally, we plan to leverage the LiDAR sensor data as *privileged information (PI)*. Learning under privileged information (LUPI) setup, proposed by Vapnik et al. [22], has privileged information available only during training. The network has to learn the correlation between features encoded from the regular inputs and privileged information, so as to perform well during inference time when the privileged information is absent. We follow the heteroscedastic dropout approach introduced by Lambert et al. [23] and corroborated by Kamienny et al. [24]. We hope to see an improvement in the training sample efficiency or final performance using privileged information. During inference time, the PI branch and the dropout can be safely ignored and hence the model still remains an image-only model.

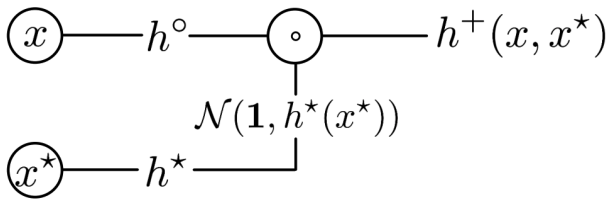

Fig. 6. Using PI for Heteroscedastic Dropout: x is the RGB image, and x^{*} is the LiDAR PI. The samples from Gaussian distribution serves as dropout for the main tower of our network. Source: [23]

We propose to use the LiDAR information to improve the depth distribution prediction component of the LSD model. This model is called LSD-PI. This choice is guided by the limited compute resource and evidence improvements in depth prediction translating to excellent detection performance [20]. The network architecture is illustrated in Figure 7. In this setup, there is an additional regularization component on the predicted standard-deviations from PI, weighted by α.

IV. Experiments

We plan to use the nuScenes dataset [18], performing our experiments on the full dataset. As we are constrained by the compute resources, we plan to train our 3 models till 50 epochs.

We initially planned to use OFT [12] as our baseline, owing to the similarities mentioned in the previous section,

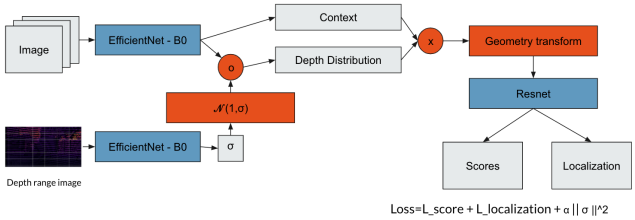

Fig. 7. LSD-PI: Lift-Splat-Detect, using LiDAR as privileged information.

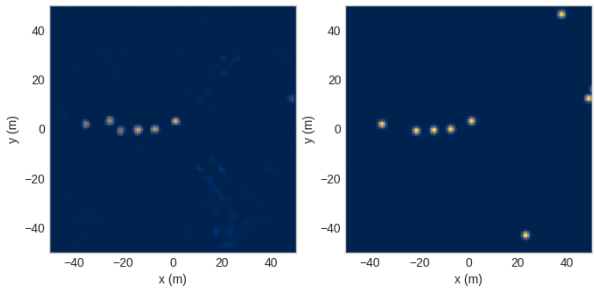

Fig. 8. A snapshot of our preliminary results for the modified Lift-Splat architecture. The yellow regions represent Gaussians centered around objects. The left image is the prediction made by our model, whereas the right image is the ground truth data.

and retrain it for the same number of epochs: 50. However, while training on nuScenes, we observe poor results and on debugging found a bug in the author's code. The object mask was incorrect with the loss being calculated across the entire image instead of just the ground truth bounding boxes, which we subsequently corrected.

We therefore decided to first reproduce the authors' results on the KITTI benchmark as reported in their paper as a sanity check before retrying on the full nuScense dataset [18]. However, even when using the corrected code provided by the authors, using their default parameters, we were unable to reproduce their results. Given the lack of time and our limited computational resources, we decided to use CenterNet [15] and MonoDIS [14] as our baselines, reported on the nuScenes leaderboard, since these were the top two performing camera-only methods at the time that we were conducting run our experiments.

We next trained the LSD model, for 3D object detection instead of segmentation. For the score component output by the Resnet, we experimented with Huber loss, Focal loss (as used in CenterPoint [25]), as well re-weighting negative samples for the Huber loss (similar to OFT [12]). We obtained the best results when using the latter. Localization involves angel, dimension, and position loss terms, for which we used Huber loss evaluated on just the anchors that had an overlap with the ground truth. We only train our model to detect cars and report results on the validation split. The preliminary qualitative results are presented in Figure 8. A small video snippet is available at the following link: modified Lift-Splat on nuScenes mini dataset. While the results seemed promising, we noticed that our model suffered from overfitting and failed on larger distances.

Our best set of hyperparameters obtained after tuning are: learning rate of $1e^{-7}$ without any scheduling, score weight of 2.5 reweighting negative samples using a value of 0.1, and 1 for the localization terms, and a weight decay of $1e^{-7}$. It takes our model 30 hours to train for 50 epochs. To report our results, we use the following metrics:

- **mAP**: Mean average precision,
- **ATE**: Average translation error, obtained by computing the 2D Euclidean distance between bounding box centers in meters,
- **ASE**: Average scale error, obtained by aligning bounding box centers and orientations and computing (1 - IoU),
- **AOE**: Average orientation error, obtained by calculating the smallest difference in yaw angles between ground truth and prediction, in radians.

In all the tables below, the arrows next to the metrics that point downward indicate that lower values are better, whereas arrows pointing upwards are used to signify that higher values indicate better results. Table I describes our results after training solely on monocular images for 50 epochs.

Method	**mAP** (↑)	**ATE** (↓)	**ASE** (↓)	**AOE** (↓)
LSD*	0.22	0.52	0.16	0.15
CenterNet [15]	0.54	0.47	0.14	0.09
MonoDIS [14]	0.48	0.61	0.15	0.07

TABLE I
COMPARISON OF LSD (OUR METHOD) WITH OUR TWO BASELINES.

As can be observed, the results we obtained were significantly worse than our baselines. On inspecting further, we found that this was due to overfitting, particularly in the object dimensions. We trained the model upto 300 epochs, and the overfitting becomes more severe, with lowest loss around 50 epochs.

Even though these results were sub-optimal, we decided to continue and experiment with LSD and LSD-PI for this project. This decision was taken in interest of time and resource constraints. We first establish a ceiling for LSD-PI using LSD-LiDAR, as LSD-LiDAR as LiDAR sensor information available during both training and inference. We report these results in Table II. We note a significant performance improvement in the mAP and ATE metrics, which were expected because of the additional depth information available.

Method	**mAP** (↑)	**ATE** (↓)	**ASE** (↓)	**AOE** (↓)
LSD	0.22	0.52	0.16	0.15
LSD-LiDAR	0.37	0.44	0.15	0.15

TABLE II
RESULTS OF SENSOR FUSION ON VALIDATION SPLIT, TRAINED FOR 50 EPOCHS.

We next train our third model, LSD-PI, on the full dataset. We noticed a significant improvement in performance com-

pared to our first model, LSD, when trained for 5 epochs. These results can be found in Table III. We report results for 3 values of the regularization of the standard deviations predicted by the PI: high values of α means that PI won't be used by the network, and low values of α will mean that we suffer during inference as we rely on the PI heavily.

Method	**mAP** (↑)	**ATE** (↓)	**ASE** (↓)	**AOE** (↓)
LSD	0.134	0.547	0.153	0.189
LSD-LiDAR	0.180	0.492	0.156	0.203
LSD-PI ($\alpha = 1e^{-1}$)	0.135	0.544	0.150	0.198
LSD-PI ($\alpha = 1e^{-3}$)	0.148	0.551	0.154	0.177
LSD-PI ($\alpha = 1e^{-4}$)	0.137	0.530	0.152	0.181

TABLE III
COMPARISON OF LSD, LSD-LIDAR, AND LSD-PI, **5 EPOCHS**

Method	**mAP** (↑)	**ATE** (↓)	**ASE** (↓)	**AOE** (↓)
LSD	0.191	0.541	0.153	0.150
LSD-LiDAR	0.244	0.474	0.153	0.138
LSD-PI ($\alpha = 1e^{-1}$)	0.173	0.516	0.157	0.177
LSD-PI ($\alpha = 1e^{-3}$)	0.177	0.529	0.156	0.141
LSD-PI ($\alpha = 1e^{-4}$)	0.172	0.527	0.150	0.160

TABLE IV
COMPARISON OF LSD, LSD-LIDAR, AND LSD-PI, **10 EPOCHS**

Method	**mAP** (↑)	**ATE** (↓)	**ASE** (↓)	**AOE** (↓)
LSD	0.221	0.522	0.159	0.147
LSD-LiDAR	0.372	0.450	0.160	0.146
LSD-PI ($\alpha = 1e^{-1}$)	0.217	0.537	0.161	0.156
LSD-PI ($\alpha = 1e^{-3}$)	0.219	0.533	0.158	0.153
LSD-PI ($\alpha = 1e^{-4}$)	0.205	0.520	0.161	0.160

TABLE V
COMPARISON OF LSD, LSD-LIDAR, AND LSD-PI, **50 EPOCHS**

We report our results for 10 epochs and 50 epochs in Tables IV and V respectively. We lose the performance gains in mAP on both these results. Improvements in ATE still hold till 10 epochs, but disappear at 50 epochs.

This might be caused due to overfitting, but we need to perform thorough investigation. PI looks promising but we need to investigate the cause for dissipation of the gains. We report the qualitative results of our final experiments in Figure 9.

V. FUTURE WORK

Based on the results we have obtained so far, we intend to first tackle the issue of overfitting in our RGB-only model. We aim to include more aggressive data augmentation. In the current set of experiments, we have trained our model to only detect cars. We want to transition to multi-class training since we believe this will also enable us to mitigate the issue of overfitting. We next want to examine the use of privileged information in more detail, digging deeper into the statistics of the dropout values, as well as probe regions where the dropout values have high variance to obtain more clarity about the training process. In addition to these experiments, we also want to observe the effect of adding a depth prediction loss to our current framework, and establish another baseline for training with privileged information by using it in conjunction with an established 3D detection algorithm.

VI. ACKNOWLEDGMENTS

We would like to thank Sean Foley (course TA) and John Lambert for their advice and suggestions for our experiments.

REFERENCES

[1] S. Singh, "Critical reasons for crashes investigated in the national motor vehicle crash causation survey," tech. rep., 2015.

[2] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A survey on 3d object detection methods for autonomous driving applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3782–3795, 2019.

[3] Y. Zhou and O. Tuzel, "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, 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*, pp. 12697–12705, 2019.

[5] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

[6] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3d object detection in point clouds," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9277–9286, 2019.

[7] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7074–7082, 2017.

[8] X. Liu, C. R. Qi, and L. J. Guibas, "Flownet3d: Learning scene flow in 3d point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 529–537, 2019.

[9] X. Liu, M. Yan, and J. Bohg, "Meteornet: Deep learning on dynamic 3d point cloud sequences," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9246–9255, 2019.

[10] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3d object detection for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2147–2156, 2016.

[11] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8445–8453, 2019.

[12] T. Roddick, A. Kendall, and R. Cipolla, "Orthographic feature transform for monocular 3d object detection," *arXiv preprint arXiv:1811.08188*, 2018.

[13] J. Philion and S. Fidler, "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d," *arXiv preprint arXiv:2008.05711*, 2020.

[14] A. Simonelli, S. R. Bulo, L. Porzi, M. López-Antequera, and P. Kontschieder, "Disentangling monocular 3d object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1991–1999, 2019.

[15] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," in *arXiv preprint arXiv:1904.07850*, 2019.

[16] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1907–1915, 2017.

[17] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–8, IEEE, 2018.

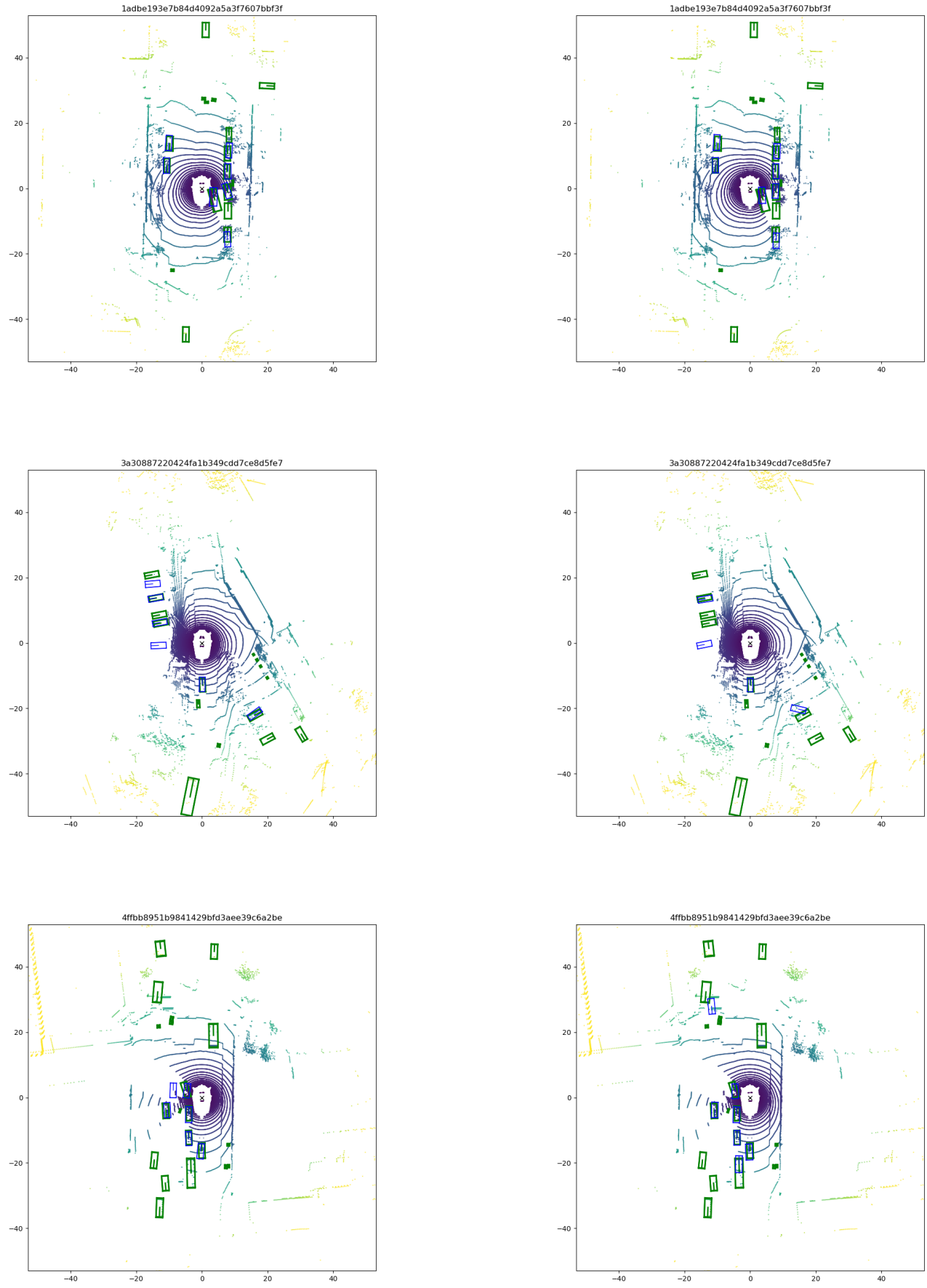

Fig. 9. Detection from LSD (left column) and LSD-PI (right column).

[18] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11621–11631, 2020.

[19] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, *et al.*, "Lyft level 5 av dataset 2019," *urlhttps://level5. lyft. com/dataset*, 2019.

[20] X. Ma, S. Liu, Z. Xia, H. Zhang, X. Zeng, and W. Ouyang, "Rethinking pseudo-lidar representation," in *European Conference on Computer Vision*, pp. 311–327, Springer, 2020.

[21] Y. Chen, S. Liu, X. Shen, and J. Jia, "Dsgn: Deep stereo geometry network for 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12536–12545, 2020.

[22] V. Vapnik and A. Vashist, "A new learning paradigm: Learning using privileged information," *Neural networks*, vol. 22, no. 5-6, pp. 544–557, 2009.

[23] J. Lambert, O. Sener, and S. Savarese, "Deep learning under privileged information using heteroscedastic dropout," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8886–8895, 2018.

[24] P.-A. Kamienny, K. Arulkumaran, F. Behbahani, W. Boehmer, and S. Whiteson, "Privileged information dropout in reinforcement learning," *arXiv preprint arXiv:2005.09220*, 2020.

[25] T. Yin, X. Zhou, and P. Krähenbühl, "Center-based 3d object detection and tracking," *arXiv preprint arXiv:2006.11275*, 2020.