Leveraging Dynamic Occupancy Grids for 3D Object Detection in Point Clouds

David Sierra González, Anshul Paigwar, Özgür Erkent, Jilles Dibangoye, Christian Laugier

Leveraging Dynamic Occupancy Grids for 3D Object Detection in Point Clouds

David Sierra-González, Anshul Paigwar, Ozgur Erkent, Jilles Dibangoye, and Christian Laugier

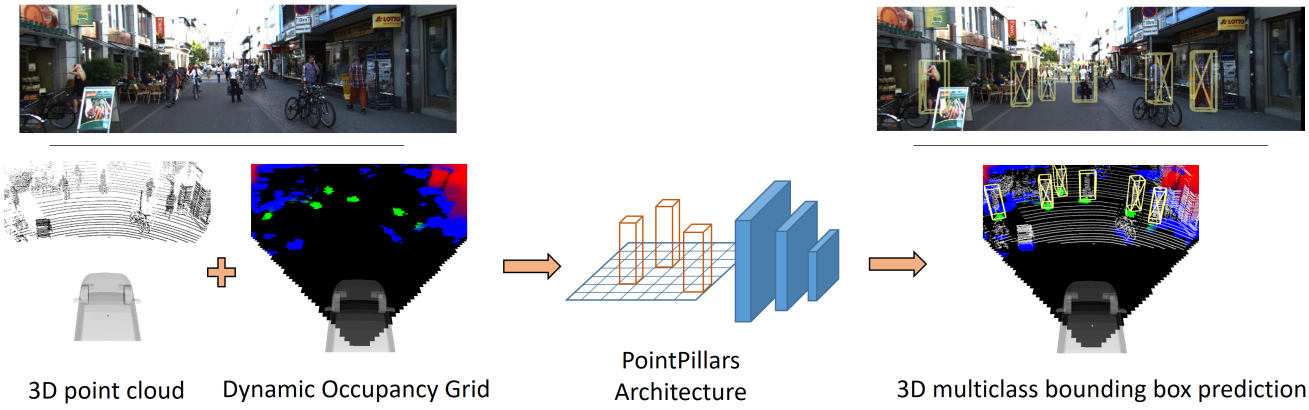


Fig. 1: 3D object detection in point clouds leveraging dynamic occupancy grid representations of the environment.

***Abstract*— Traditionally, point cloud-based 3D object detectors are trained on annotated, non-sequential samples taken from driving sequences (e.g. the KITTI dataset). However, by doing this, the developed algorithms renounce to exploit any dynamic information from the driving sequences. It is reasonable to think that this information, which is available at test time when deploying the models in the experimental vehicles, could have significant predictive potential for the object detection task. To study the advantages that this kind of information could provide, we construct a dataset of dynamic occupancy grid maps from the raw KITTI dataset and find the correspondence to each of the KITTI 3D object detection dataset samples. By training a Lidar-based state-of-the-art 3D object detector with and without the dynamic information we get insights into the predictive value of the dynamics. Our results show that having access to the environment dynamics improves by 27% the ability of the detection algorithm to predict the orientation of smaller obstacles such as pedestrians. Furthermore, the 3D and bird's eye view bounding box predictions for pedestrians in challenging cases also see a 7% improvement. Qualitatively speaking, the dynamics help with the detection of partially occluded and far-away obstacles. We illustrate this fact with numerous qualitative prediction results.**

I. INTRODUCTION AND RELATED WORK

Perceiving the environment in a reliable manner is one of the hardest challenges that needs to be solved before the commercial deployment of fully autonomous vehicles becomes a reality. Amongst other perception tasks, autonomous robots need to detect static and moving objects in the 3D world, under varying weather and lighting conditions. To perform this task, different sensing technologies are available, namely, monocular and stereo cameras, Lidar scanners and radar.

The avalanche of novel deep learning techniques developed over the past decade have brought the 2D object detection task very close to be considered a solved problem [1], [2], [3]. However, in robotics, and more particularly in the domain of autonomous driving, we are more interested in detecting objects in 3D; that is, we aim to find oriented 3D bounding boxes around the obstacles. Unfortunately, the progress in 3D obstacle detection methods is lagging behind their 2D counterparts.

To improve the situation, different sensing modalities are being explored. For example, some approaches aim to detect the 3D obstacles relying exclusively in monocular or stereo images [4]. However, it seems that the characteristics of Lidar sensors are more suitable for this perception task, as evidenced by the ranking of the KITTI 3D object detection benchmark [5]. Lidars can directly provide range information and are unaffected by lighting conditions. Nonetheless, the point cloud data these sensors provide is sparse, and it becomes increasingly so at larger distances.

Different detection approaches building upon point cloud data have been introduced in recent years. Since the point cloud is sparse, a common technique is to voxelize the 3D space to create a tensor on which to perform convolutional operations and transfer techniques from the 2D detection field. Li proposes to encode each voxel with the occupancy value of the point cloud and then apply 3D convolutions [6]. Other authors opt for a top-view representation of the point cloud, in which the height corresponds to hand-engineered features such as the height map of the points, the point density and the reflectance at the maximum height [7], [8].

Instead of relying on hand-crafted features, other au-

This work was supported by Toyota Motor Europe.

[1]The authors are with Univ. Grenoble-Alpes, Inria, 38000 Grenoble, France - {david.sierra-gonzalez, anshul.paigwar, ozgur.erkent, jilles.dibangoye, christian.laugier}@inria.fr

thors use the PointNet architecture to automatically learn discriminative point cloud features relevant for the object detection task [9], [10], [11]. The VoxelNet architecture voxelizes the 3D space and encodes the points inside each voxel with features learned with PointNet; this point cloud representation is followed by 3D convolutions, a backbone network, and a single-stage detection head [10]. The PointPillars architecture follows a similar approach but avoids the expensive 3D convolution by representing the point cloud from the top view, where the height dimension corresponds to the PointNet features.

One of the common characteristics of all the cited approaches is that none of them exploit the sequential nature of the sensor measurements, i.e. with each new sample they forget everything they saw before. One explanation for this is that the KITTI dataset, the gold standard of perception datasets for robotics and autonomous driving, only provides non-sequential samples in the detection benchmarks. If no sequential information is provided, then the dynamics of the environment cannot be leveraged for the detection task.

To the best of our knowledge, the only approach that considers successive point cloud measurements with a detection objective is Fast and Furious [12]. This approach jointly addresses the object detection, tracking, and prediction tasks. Its key idea is to stack together voxelized representations of a given number of successive past point clouds and perform 3D convolutions over space and time to predict bounding boxes for future frames. The authors claim that this approach leads to improved detection of occluded and far-away obstacles.

Although not directly detecting on point clouds, Hoermann et al. propose to perform bird's eye view (BEV) bounding box predictions using dynamic occupancy grid environment representations of each sample [13]. Their architecture consists of an encoder-decoder with single-stage detection head. Their results show that they can successfully predict BEV bounding boxes for cars and pedestrians on a dataset collected from a stationary platform. It is unclear whether this approach would scale to datasets with a more diverse collection of traffic scenes.

In this paper, we perform an in-depth study on how leveraging the dynamics of a driving scene can affect the 3D object detection capabilities of a state-of-art detection architecture on the KITTI dataset. We hypothesize, in line with previous work [12], that this information has significant predictive potential for the object detection task. To perform our study, we extend the KITTI 3D object detection dataset with a dynamic occupancy grid for each sample [5].

The rest of the document is organized as follows: Section II describes the proposed experiment and provides basic information about dynamic occupancy grids and about the 3D detection architecture that will be tested; Section III describes the dataset that has been prepared to provide dynamic information of the environment in the KITTI 3D object detection dataset. This section also presents and discusses the different qualitative and quantitative experimental results. Lastly, Section IV concludes.

II. 3D OBJECT DETECTION IN POINT CLOUDS WITH DYNAMIC INFORMATION

To determine if Lidar-based 3D object detection algorithms can benefit from having access to the dynamic information of a scene, we extend the 3D KITTI object detection dataset with the dynamic occupancy grid representation of the environment for each of the samples. We describe the constructed dataset in subsection III-A. In this section, we provide details about the tracker used to generate the grid representations and about how we can combine these representations with the features that encode the point cloud data.

A. Bayesian occupancy grid filtering

Bayesian occupancy grid filtering is a technique for situational awareness in which the environment is represented as a grid of cells and the occupancy state of each cell is maintained over time [14], [15]. By processing measurements over time, this technique can estimate the velocity of each cell in the grid, modeling thus the dynamic environment.

To produce the dynamic occupancy grids (also known as dynamic occupancy grid maps, or DOGMas) for the KITTI dataset we use the Conditional Monte Carlo Dense Occupancy Tracker (CMCDOT) [16]. This approach uses the Lidar measurements to maintain two distinct discrete probability distributions for each cell, one over its occupancy state and the other over the velocity of the cell's occupancy. The possible occupancy states for a cell are: "occupied by a static object", "occupied by a dynamic object", "empty" and "unknown". Figs. 1-3 show visualizations of some of the DOGMas of the KITTI validation split. The color scheme is the following: the red channel is for the unknown probability, green is for the dynamic occupancy probability, and blue is for the static occupancy probability. Thus, cells colored in black are estimated to be empty. The cells outside the FOV of the camera are displayed in white.

B. Leveraging DOGMAs for 3D object detection

Fig. 2 shows an overview of the proposed architecture used to study the impact of dynamic data in the 3D object detection performance. After receiving a Lidar point cloud measurement, we can obtain the DOGMa environment representation with the occupancy tracker. Using the FeatureNet component of the PointPillars architecture (in essence, a simplified version of PointNet [9]), we can also produce a tensor known as pseudo-image that characterizes the distribution of Lidar points above each cell [11]. We then concatenate both environment representations and pass them through the backbone and SSD detection head to produce the 3D bounding boxes and the class predictions [2].

The DOGMa calculation on GPU is about 50ms for the selected grid size and tracker parameters. However, this calculation can be executed in parallel with the point cloud preprocessing and the pseudo-image generation, which take up roughly 60% of the PointPillars total runtime. In any case, the DOGMa computation time could be significantly reduced by trading-off grid accuracy.

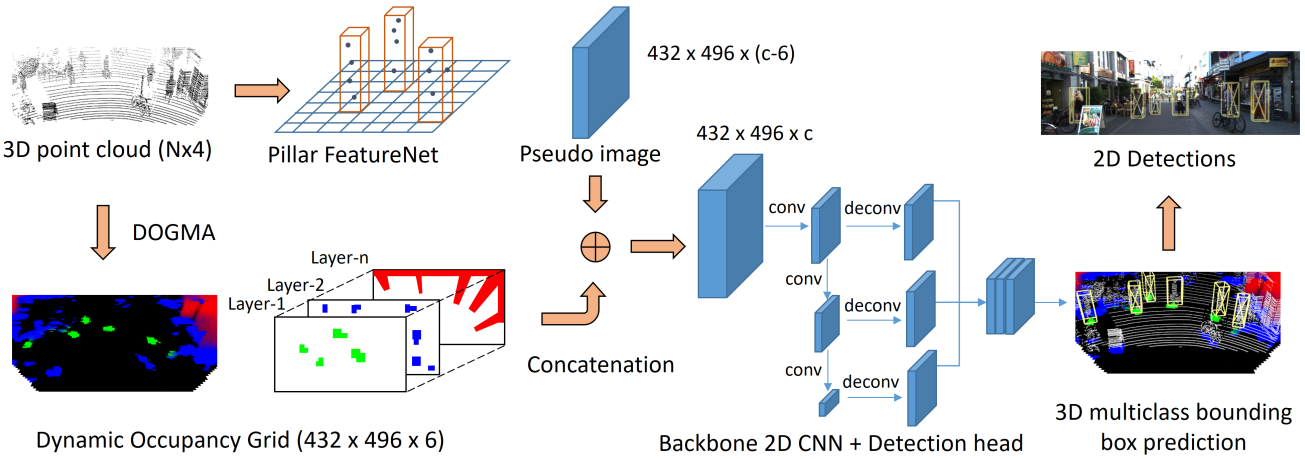


Fig. 2: Overview of the proposed experimental architecture. Each point cloud from the KITTI 3D object detection dataset has an associated DOGMa obtained from the KITTI raw dataset. Using PointPillar's point cloud feature extractor, a tensor that matches the size of the DOGMa is obtained and concatenated with the DOGMa features. The resulting tensor is passed through the backbone network and the SSD detector head produces the 3D bounding box predictions

III. EXPERIMENTAL EVALUATION

In this section, we first detail the characteristics of the constructed DOGMa dataset and the settings selected for the PointPillars architecture and for the experimental evaluation. Next, we present and discuss the quantitative results obtained. Finally, we present several qualitative results to help illustrate the main points of the paper.

A. Dataset construction

The KITTI 3D object detection dataset consists of 7481 training images and 7518 test images, as well as their corresponding point clouds, with a total of 80.256 labeled objects. The training set is typically split into 3712 training samples and 3769 validation samples [17], [11]. To find the DOGMas corresponding to each of the training and validation samples, we run the CMCDOT dynamic occupancy grid filter on all the driving sequences of the KITTI raw dataset. The mapping from each training and validation sample to the corresponding driving sequence sample is provided. For the test samples the mapping is not provided, so we will not be able to benchmark our approach on the test set.

The selected DOGMa size was $(432, 496)$ cells, with a resolution of 0.16m. This corresponds to a distance of 69.12m along the x axis (front of the vehicle) and 39.68m to each side in the y axis. Each cell has six associated floating point values, namely, the probabilities of the four possible states of the cell (occupied static or dynamic, free, unknown), and the estimated velocities for the cell's occupancy along the x and y axes. The KITTI dataset provides no labels for objects outside the field of view (FOV) of the camera. All cells lying outside the FOV have their values set to 0.

As the CMCDOT filtering requires a few steps before convergence, the DOGMas of some of the training and validation samples that correspond to the beginning of the driving sequences could not be obtained. Also, the odometry of one of the driving sequences of the raw dataset appeared to be corrupted, so the DOGMa could not be calculated for any of the samples taken from that sequence. All in all, the final number of training and validation samples is reduced to 3705 and 3570, respectively. The complete DOGMa dataset will be released upon publication[1].

B. 3D detection architecture and experimental settings

The settings selected for the PointPillars architecture correspond in general to the default values of the PyTorch open-source code released by the authors of the paper[2]. The cell resolution is also set to 0.16m and the point cloud limits are selected so that the size of the pseudo-image matches the size of the generated DOGMas. Some of the more relevant parameters are the maximum number of voxels, which is set to 12.000; the maximum number of points per voxel, set to 100; and the number of features automatically learned per cell ($c - 6$ in Fig. 2), which we set to 64.

A model is trained for the car class and another for the pedestrian/cyclist classes. We train each model for 160 epochs. The evaluation is performed on the validation set. Following previous work, we perform object and global augmentation in each sample during training [18], [10], [11]. Object augmentation refers to inserting artificial instances of the target object class in each sample. The artificial instances are sampled from a database constructed using the original dataset. In our case, each artificial instance consists of the object's point cloud as well as the DOGMa cells below the points. Note that we do not insert the trail of dynamic occupancy that lags behind objects moving at medium to large speeds.

C. Quantitative results

To evaluate the impact of the DOGMa features in the obstacle detection task, we consider three different metrics. First, we look at the Average Precision (AP) of the 3D

[1] https://github.com/d451gon/kitti_dogma_dataset
[2] https://github.com/nutonomy/second.pytorch

Features		mAP		Car			Pedestrian			Cyclist		
Pointnet	Grid	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
✗	✓	13.44	13.44	20.74	16.45	15.48	17.75	15.46	14.99	11.57	08.41	08.21
✓	✗	**64.92**	**59.33**	**84.06**	**75.13**	**69.43**	**62.57**	57.52	51.17	**81.96**	**62.11**	**57.39**
✓	✓	58.50	56.56	80.45	67.57	66.30	61.64	**57.89**	**55.78**	69.01	50.06	47.60

TABLE I: Average precision results on the validation split of the KITTI 3D detection dataset

Features		mAP		Car			Pedestrian			Cyclist		
Pointnet	Grid	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
✗	✓	29.75	28.11	56.32	43.26	39.93	26.33	23.43	22.60	30.35	22.56	21.80
✓	✗	**72.63**	**69.01**	**89.99**	**87.13**	**85.15**	70.54	65.70	60.18	**85.07**	**65.06**	**61.72**
✓	✓	68.00	66.44	89.54	81.91	81.83	**72.64**	**67.59**	**65.92**	72.55	54.51	51.58

TABLE II: Average precision results on the validation split of the KITTI Bird's Eye View detection dataset

Features		mAP		Car			Pedestrian			Cyclist		
Pointnet	Grid	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
✗	✓	21.47	21.27	40.87	31.08	29.69	14.04	12.47	13.01	28.26	20.87	21.13
✓	✗	**65.29**	**63.33**	**90.58**	**88.25**	**86.29**	42.51	41.98	40.47	**84.67**	**65.65**	**63.25**
✓	✓	64.70	62.12	90.01	85.64	81.86	**55.71**	**53.15**	**50.55**	75.19	55.32	53.96

TABLE III: Average orientation similarity results on the validation split of the KITTI 3D detection dataset

bounding box prediction task; these results are shown in Tab. I. The performance evaluation is separated by classes and also by difficulty. The difficulty depends on different aspects of the obstacle, such as its distance to the ego-vehicle, its level of occlusion, and how truncated it is out of the FOV of the camera.

In the top row of Tab. I we show the results obtained when the only features passed to the backbone network are those of the DOGMa. As we can see, it is not possible to detect 3D objects solely from occupancy grid information.

The second and third rows in Tab. I show the results obtained by the model with point cloud features but without DOGMa features, and the model with both point cloud and DOGMa features, respectively. We can observe that the DOGMa features provide no advantage for the 3D bounding box prediction task in the car and cyclist classes.

In contrast, these features provide an edge for the pedestrian class in the moderate and, especially, in the hard categories. Roughly 25% of the pedestrians in the KITTI dataset are occluded to some degree, while the number climbs to roughly 40% for cars [5]. Inspecting the dataset we see that the difference between both classes is that most pedestrians appear to be moving, while there is a large number of cars that are static. This leads us to hypothesize that having access to the DOGMa features, and in particular to the dynamic-related ones, helps in the detection of partially occluded, moving obstacles. This makes sense, as a partially occluded pedestrian should be able to be detected from a reasonably-sized dynamic occupancy and a number of Lidar hits consistent with the average class size. This intuition carries over to obstacles located far-away from the ego-vehicle. It is not clear why the performance improvement does not apply also to cyclists. The number of cyclists in the dataset is about one fourth the number of pedestrians, but there is no information about their occlusion statistics. A possibility is that the object augmentation procedure, in which the dynamic occupancy trail behind fast moving objects was not inserted, might have harmed the cyclist model performance.

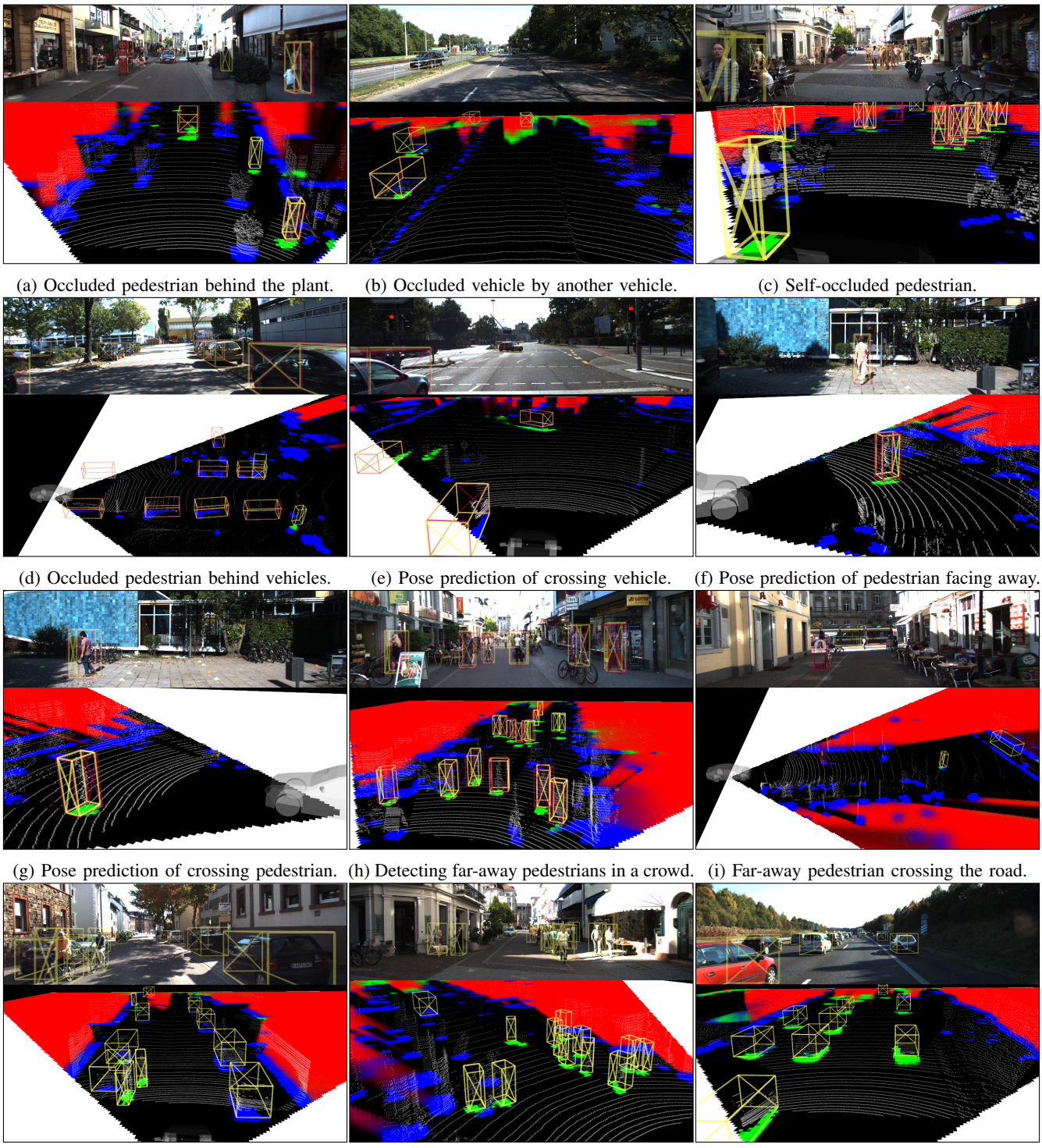
Table II provides the AP results for the BEV detection task. The results are consistent with the 3D metrics discussed above.

Finally, Tab. III presents the Average Orientation Similarity metric, which measures how well the heading of the predictions matches the ground truth. Having access to the dynamics of a moving obstacle should facilitate the task of predicting its orientation. In our case, we observe a significantly superior performance for the pedestrian class, which is typically the class whose orientation is harder to predict based only on point cloud data. However, the performance drops in the hard category of the car class (probably connected with noise present in the DOGMa in distant regions) and in all categories of the cyclist class, which did not converge (jointly with the pedestrian class) to a good model.

D. Qualitative results

Figure 3 shows several instances of the predictions obtained in the validation set. In Figs. 3a-3i we show the predictions for the model with point cloud and DOGMa features (yellow) and the one without DOGMa features (purple). More particularly, Figs. 3a-3d show scenes in which at least one of the obstacles is occluded. In Fig. 3a, a pedestrian is largely occluded behind a plant, receiving Lidar hits only in the head and upper torso. However, this number of Lidar hits is enough to track the dynamic occupancy by the CMCDOT tracker, leading to a positive detection with the right orientation. The model without DOGMa features fails to detect this pedestrian. Similar situations occur in Figs. 3b through 3d.

The difficulty of predicting the orientation of an obstacle from point cloud data without any access to dynamics is highlighted in Figs. 3e-3g. Of particular interest is the case of Fig. 3f; predicting the orientation of a pedestrian facing away from the ego-vehicle solely from point cloud data does



(a) Occluded pedestrian behind the plant. (b) Occluded vehicle by another vehicle. (c) Self-occluded pedestrian.

(d) Occluded pedestrian behind vehicles. (e) Pose prediction of crossing vehicle. (f) Pose prediction of pedestrian facing away.

(g) Pose prediction of crossing pedestrian. (h) Detecting far-away pedestrians in a crowd. (i) Far-away pedestrian crossing the road.

(j) Exemplary prediction on a residential road. (k) Exemplary prediction on a city street. (l) Exemplary detection on a highway setting.

Fig. 3: Qualitative results obtained in different scenes from the validation split of the KITTI 3D object detection dataset. The ground colors represent the state of each cell of the DOGMa (blue: occupied static, green: occupied dynamic, black: free, red: unknown). Figures (a-i) show the bounding box predictions of the models with DOGMa features (yellow) and without (purple). Figures (j-l) show exemplary predictions of the model with DOGMa features. The first 9 figures aim to show the advantage given by the extra features in the detection of: occluded obstacles (a-d), the orientation of moving obstacles (e-g) and far-away obstacles (h-i).

not seem like an easy task, especially as the distance with the obstacle increases.

Figures 3h and 3i illustrate detections of far-away pedestrians, which are facilitated by the same reasons discussed in

the case of occluded obstacles. Finally, Figs. 3j-3l present some further exemplary predictions of the model with DOGMa features in different road-network settings.

A video showing the predictions produced by the proposed model in different sequences of the KITTI raw dataset is available at: https://youtu.be/hbQgpRuvuUo.

IV. CONCLUSIONS

In this paper, we studied the impact of combining a dynamic representation of the environment along with point cloud feature descriptors as inputs to a 3D object detection architecture. To achieve this, we created a dataset of DOGMas for the KITTI 3D object detection dataset by running an occupancy tracker on the KITTI raw dataset and using the provided correspondence between both datasets.

With the KITTI dataset enhanced by the dynamic representation of the environment, we trained the PointPillars 3D object detection architecture with different combinations of input features. In light of the detection results, we reached several conclusions. Firstly, using only the occupancy grid features as the input to the detector is not sufficient to produce acceptable 3D or even BEV bounding box predictions for a dataset with diverse traffic scenarios such as KITTI.

Secondly, including dynamic information as input data leads to a significant improvement in the orientation predictions for small obstacles with less points such as pedestrians.

Finally, the 3D and BEV AP detection metrics for the pedestrian class also benefit from including the dynamic information as an input, seeing a 9% performance boost in the hard category for both metrics. We hypothesize that this is caused by an improved ability to detect partially occluded and far-away obstacles. In other words, if an obstacle does not have many lidar hits but we have information that it is moving, the detection can still happen. On the other hand, the detection performance for the car and cyclist classes did not benefit in general from the inclusion of the dynamic features in the dataset. This might be due to several reasons: a) these classes typically have more Lidar hits than the pedestrian class, and the extra information in the shape of occupancy data might constitute noise for the detector; b) noise in the dynamic data present towards the edges of the grid; and c) the object augmentation procedure in which the trail behind fast-moving obstacles was not artificially inserted during training.

Following the conclusions of this paper, in following work we aim to study the effect of leveraging DOGMa features for the 3D obstacle detection task in sparser point clouds, such as those obtained from Lidars with 32 or 16 layers.

Additionally, the evaluation in this paper has only been performed on the validation split of the KITTI dataset, as the test set does not provide the mapping to the raw KITTI dataset. In future work, we will perform the analysis also on the nuScenes dataset [19], which provides the preceding point clouds leading to each detection sample.

ACKNOWLEDGMENT

We would like to thank Gabriel Othmezouri from TME for the helpful discussions during the preparation of this work. Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

REFERENCES

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems 28*, 2015, pp. 91–99.

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," *CoRR*, vol. abs/1512.02325, 2015.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[4] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," *CoRR*, vol. abs/1612.00496, 2016.

[5] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[6] B. Li, "3d fully convolutional network for vehicle detection in point cloud," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 1513–1518.

[7] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6526–6534.

[8] 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)*, 2018, pp. 1–8.

[9] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *CoRR*, vol. abs/1612.00593, 2016.

[10] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4490–4499.

[11] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 12 689–12 697.

[12] W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3569–3577.

[13] S. Hoermann, P. Henzler, M. Bach, and K. Dietmayer, "Object detection on dynamic occupancy grid maps using deep learning and automatic label generation," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 826–833.

[14] A. Petrovskaya, M. Perrollaz, L. Oliveira, L. Spinello, R. Triebel, A. Makris, J.-D. Yoder, C. Laugier, U. Nunes, and P. Bessiere, *Awareness of Road Scene Participants for Autonomous Driving*. London: Springer London, 2012, pp. 1383–1432.

[15] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière, "Bayesian occupancy filtering for multitarget tracking: An automotive application," *The International Journal of Robotics Research*, vol. 25, no. 1, pp. 19–30, 2006.

[16] L. Rummelhard, A. Nègre, and C. Laugier, "Conditional monte carlo dense occupancy tracker," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 2015, pp. 2485–2490.

[17] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," in *Advances in Neural Information Processing Systems 28*, 2015, pp. 424–432.

[18] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, 2018.

[19] 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," *arXiv preprint arXiv:1903.11027*, 2019.