Transforming LiDAR Point Cloud Characteristics between different Datasets using Image-to-Image Translation

Felix Berens[1,2], Yannick Knapp[1], Markus Reischl[2], Stefan Elser[1]

[1]Institute for Artificial Intelligence
Ravensburg-Weingarten University of Applied Sciences, Weingarten, Germany
E-Mail: Felix.Berens@rwu.de

[2]Institute for Automation and Applied Informatics
Karlsruhe Institute of Technology, Karlsruhe, Germany
E-Mail: Felix.Berens@kit.edu

Abstract

In recent years several new LiDAR datasets for object detection were published. All these datasets were recorded with different LiDAR setups and at different locations. KITTI, for example, has 64 channels and was recorded in Germany, whereas Lyft (Level 5) has only 40 channels and was recorded in the USA. This leads to different characteristics of the LiDAR point clouds. In this paper, we present and evaluate a way to transform KITTI BEV maps such that they look like Lyft BEV maps. For this transformation we use the state-of-the-art image-to-image translator CycleGAN. The transformation is evaluated by two strategies: Firstly we test if the translated KITTI BEV maps work better for an object detector, which is trained on Lyft. Secondly we test if the characteristic structure of the Lyft dataset (number of channels, location of points) is adopted from the translated point cloud. The conducted experiments showed that after the translation the KITTI BEV maps are more similar to Lyft BEV maps, but the detection got worse.

1 Introduction

Especially in autonomous driving, any method used to evaluate the 3D driving environment must be fail safe. LiDAR, RADAR and ultrasonic sensors are used to obtain 3D information about the surrounding area. While an ultrasonic sensor only works in short range and the 3D information of a RADAR is very sparse, the LiDAR sensor can produce accurate information of the immediate surroundings such as pedestrians or other vehicles.

With the KITTI dataset [1], a dataset for autonomous driving containing LiDAR point clouds is publicly available since 2012 and is still used as a standard dataset. In recent years, new datasets for autonomous driving containing LiDAR point clouds became publicly available e.g. 2018 nuScenes [2], 2019 Lyft Level 5 [3], 2019 Audi A2D2 [4] or 2020 Ford AV [5]. With Astyx HiRes [6] the first dataset for autonomous driving was published, which also contains point clouds from an high resolution RADAR in addition to LiDAR point clouds. As Table 1 shows, all datasets use different LiDAR setups. Significant differences between the datasets are the number of channels and the vertical resolution, which depends on the type of LiDAR that was used. While for most datasets one LiDAR is mounted on the top of the roof, for the Audi dataset multiple LiDARs were used: one at the front of the roof and 4 at the corners of the roof, with a slightly tilt. Due to the different locations of the LiDARs the point clouds of A2D2 look different than point clouds from a single LiDAR. The point clouds in A2D2 have a grid pattern, and not a circular pattern as in the other datasets with one top LiDAR. Also the different LiDAR setups for the datasets lead to a different appearance of the LiDAR point clouds (Fig. 1). Both KITTI and Lyft have a LiDAR mounted on top of the roof and generate a 360 degree vision of the surrounding area by multiple laser channels. Because in KITTI many laser channels scan the close area of the ego vehicle, a point cloud from the KITTI dataset contains much more points near the ego vehicle than a point cloud from Lyft.

[1] While working on this paper only the data of the top LiDAR were available.

Table 1: Comparison of the LiDAR setups of different datasets. Specs marked with * are taken from the datasheet of the LiDAR manufacturer and not from the source of the dataset. T = Top; B=Bumper; F=Front; C=Corner.

	KITTI	Lyft	Audi A2D2	Astyx
Number of LiDARs	1	1 + 2	5	1
Range [m]	120	-	100	100
Channels	64	40/64 [1]	16	16
Azimuthal FOV [°]	360	360	360	360
Vertical FOV [°]	26.8	-	30	30
Azimuth resol. [°]	0.08 - 0.35*	0.2	0.1-0.4 *	0.1-0.4 *
Vertical resol. [°]	0.4*	-	2	2*
Rate [Hz]	10	10	10	10
Position	T	T + 2 B [1]	F + 4 C	T
Intensity	✓	×	✓	✓
Type	HDL-64E	-	VLP-16	VLP-16

These LiDAR datasets are used to train and test methods e.g. for 2D and 3D object detection, segmentation, SLAM or optical flow. Wang et al. [7] showed that an object detector trained on one dataset (source) performs worse on another dataset (target). They concluded that a possible reason is the size of cars in different regions of the world. They applied different strategies that focus on the size of the cars: enlarge or shrink the bounding box and the corresponding point clouds in the training scenes (SN), continue training with some ground truth point clouds from the target dataset (FS) or enlarge or shrink the predicted boxes of the detector (OT). These three methods proved to be effective on the performance of the detector. All these methods base on changing the detection method and not changing the target dataset.

For multiple methods, or other tasks than object detection, it could be difficult and time consuming to modify all methods and retrain them. It would be faster if we could transform the point clouds of the target dataset so that they look like the source dataset on which we trained our methods. So in this paper we will focus on the different LiDAR setups used in the datasets. Such a type of problem is called domain adaption (Sec. 2.2). One part of domain adaptation is unsupervised image to image translation. Methods like CycleGAN [8] or UNIT [9] have shown promising results in the translation between images of

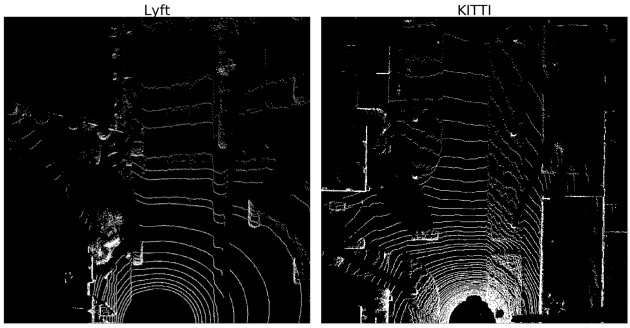

Figure 1: Sample of Bird's-Eye-View for Lyft and KITTI dataset. Both BEV maps are on information level **Position**.

different domains e.g. night to day, summer to winter or simulated to real data. While these methods work well for 2D image domains as seen in [8, 9], they are not designed for the application on 3D point clouds, like the LiDAR point cloud. Approaches for 3D object detection like Simon et al. [10] convert the 3D point clouds to Birds-Eye-View RGB-maps (BEV) and train a 2D object detector on this converted data. They encoded additional information, like height, density and intensity of the point clouds into the RGB color channels of the BEV map.

Sallab et al. [11] and Saleh et al. [12] proposed frameworks to translate synthetically BEV LiDAR maps to real BEV LiDAR maps. The simulated point clouds were generated with the CARLA simulator [13]. Without additional post-processing, the simulated point clouds are too smooth and miss artifacts that are common in real LiDAR point clouds. Sallabs and Salehs approaches used both KITTI as a real world dataset. Sallab did not state which LiDAR setup their simulated LiDAR used and encoded the height information of the 3D point clouds into the BEV maps. But Sallab did not use any additional information about intensity or density of the LiDAR points. The simulated BEV maps were translated with CycleGAN and got used together with real BEV maps from KITTI to train an object detector. Sallab improved the object detector from 65.3 % mAP, when using KITTI point clouds with 100.000 raw simulated point clouds to 71.5 %, when using KITTI point clouds and 100.000 translated simulated point clouds. Saleh used a simulated Velodyne HDL-64E

LiDAR for the simulated point clouds, so they have the same type of LiDAR as for the KITTI dataset (Tab. 1). Saleh mapped the 3D point clouds into BEV without any additional information and used only the location of the points. Saleh trained an object detector with several dataset combinations and tested it on the same split from the KITTI dataset. For KITTI alone they had an AP of 57.26 %, this was improved with the addition of 6.000 simulated point clouds to 59.16 % and with the addition of 6.000 simulated and translated point clouds to 64.29 %. Using only the 6.000 simulated point clouds for training resulted in an AP of 29.93 %, which could also be improved to 34.78 % by translating point clouds. However, this also shows, that the translation alone could not produce perfect real looking data.

We aim to make the BEV maps of two real world LiDAR datasets look more similar. For this we will:

- carry out a translation using CycleGAN between two real world LiDAR datasets for the first time
- use different information levels of BEV maps for the translation (**Position**, **Height**, **Height+Density**, **Height+Density+Intensity**)
- evaluate the quality of this translation with two different strategies. **Usability**: test if the translated target BEV maps work better for an object detector, which is trained on the source. **Structure**: test if the characteristic structure of the Lyft dataset (number of channels, location of points) is adopted from the translated BEV map.

2 Method

2.1 Datasets

We focus on the translation from KITTI to Lyft, because Lyft has a similar LiDAR setup as KITTI, but not totally similar. Both KITTI and Lyft have LiDAR mounted on the roof and generate a 360 degree vision of the surrounding area by multiple laser channels. KITTI has just one top LiDAR and we will also only use the LiDAR point clouds of Lyft's top LiDAR. The circular pattern of

both LiDAR point clouds can be seen in Figure 1. A point cloud from KITTI has much more points near the ego vehicle than a point cloud from Lyft. This can be seen in Figure 1, as more white pixels are located around the ego vehicle of KITTI in the bottom center than in the BEV map of Lyft. Furthermore, in Figure 1 the effect of the different number of LiDAR channels of KITTI and Lyft can be seen. It can be seen that the LiDAR of Lyft produces less white circles than that of KITTI in the BEV map. This is because the LiDAR in KITTI has more laser channels to scan the close area of the ego vehicle, while in Lyft the lasers are such that they focus the scan on the midrange. For the BEV maps the most significant differences between the two datasets can be seen on parts that belong to the street, while in both examples of Figure 1 the cars have a similar L-shape. The number of channels is 64 for KITTI's and 40 for Lyft's top LiDAR (Tab. 1). Hence a translation from KITTI to Lyft should delete some of these wave lines, without destroying the L-shape of a vehicle.

2.2 Domain Adaptation

The problem of training a method on one dataset (*source domain*) and applying it to another dataset (*target domain*), is called domain adaptation. Domain adaptation is a type of transfer learning and aims to bridge the gap between different domains of data [14]. A specific type of domain adaption is the image-to-image translation. In this type an input image from the source domain can be transformed to an image that is similar to the distribution of the target domain. The transformation is observable as the source image adapts the "style" of the target domain. Such a transformation can be studied in two settings: the supervised and the unsupervised setting. In the supervised translation one, every target image has a corresponding source image. In the unsupervised one, no target image has a corresponding source image. As Lyft and KITTI are recorded at different locations and different times, we do not have paired data.

Within the scope of this work, we will use the Cycle-Consistent Adversarial Network (CylceGAN) [8]. By taking pairs of images out of different domains CycleGAN learns how to apply the characteristics of one domain to the images of the other. CycleGAN consists of two generators and two discriminators:

one generator takes images of the first domain and output images of the second domain, the second generator vice versa. The demonstrators determine how plausible the output of the generators are for the domains. In addition, CycleGAN applies the cycle-consistency loss. So in CycleGAN adversarial losses are combined with cycle consistency loss, i.e. evaluating also a possible back-projection of generated data from the target domain into the source domain.

Because CycleGAN[2] is designed to translate 2D data, we will convert the 3D point clouds to 609×609 Birds-Eye-View RGB-maps. We can encode different levels of information into the pixel color values of the BEV. The simplest way of a BEV map (**Position**) uses only the location of the points and ignores the height, such that the pixels are 255 iff there is a LiDAR point at this position and 0 if not. See Figure 1 as an example. In the next level of information (**Height**), we use the height of the points and encode this information into the greyscale value of the pixel. If more than one point is at the same position, we use the maximum height. In the third level of information (**Height+Density**) we map the density of points that lie over each other in addition to the height in the first two color channels of the pixel. In the last level of information (**Height+Density+Intensity**), we also encode the intensity information, which describes how good the laser beam is reflected, into the third color channel of the pixel. If more than one point gets mapped to the same pixel, then the maximum intensity is used.

2.3 Evaluation

We choose two ways to evaluate how good the translation of the point clouds provided by KITTI into the structure provided by Lyft is:

- **Usability**: How good can be the transformed BEV maps be used for task of autonomous driving, when the method is trained on the source dataset.
- **Structure**: How well does the translated BEV maps represent the structure of the LiDAR setup from the source dataset.

[2] Our CycleGAN implementation based on the implementation of Ming-Yu Liu., which can be found on GitHub https://github.com/junyanz/CycleGAN, Jan. 2020.

2.3.1 Usability

To have a quantitative criterion how good the translation fulfills the criterion **Usability** we will use a similar strategy as [11, 12] and test the translation on the detector ComplexYOLO [10]. Instead of retraining with additional BEV maps, we train ComplexYOLO on Lyft and test how the detector performed on the translated KITTI BEV map compared to the original KITTI BEV map.

ComplexYOLO is a real-time 3D object detection network operating on LiDAR BEV maps. It is based on the YOLO framework with the addition of a specific Euler-Region-Proposal approach, that estimates the orientation of objects. The determination of the orientation of an object is necessary in BEV, since an object gets detected from above instead of the common front view and therefore can be rotated.[3]

To improve the detection of objects important structures as the shape of a car should not be destroyed after the translation, while the characteristics of the Lyft BEV maps should be adopted from the translated KITTI BEV maps.

As metrics we will use the well known intersection over union (IoU) with a threshold of 0.5 and the average precision (AP).

2.3.2 Structure

A criterion, which does not mostly depend on geometric properties of a scene, and distinguishes between different LiDAR setups, is required to measure how well the LiDAR BEV map of KITTI is translated to the characteristics of Lyft BEV maps. For this we map the BEV map back to 3D coordinates. In the case **Position** we do not have any height information. So all the height values are set to 0 for all points. These 3D coordinates are transformed to a spherical coordinate system. In a spherical coordinate system every point consists of radial distance, polar angle, and azimuthal angle (r, θ, ϕ). In the case that

[3] Our ComplexYOLO implementation based on the unofficial implementation from Deepak Ghimire, PhD, which can be found on GitHub https://github.com/ghimiredhikura/Complex-YOLOv3, Jan. 2020.

we only have one LiDAR without any tilt and the position of the LiDAR is the origin of the spherical coordinate system the angles can be interpreted as the angles of the laser beam during the scanning of the scene and r as the distance between the object and the LiDAR. So the distribution of ϕ and θ depends on the LiDAR setup. Because the azimuthal LiDAR setup in KITTI and Lyft is similar, the distribution should be similar and only differ mostly due to geometric properties of the scene. Because θ describes the polar angle it depends on the vertical resolution and vertical FOV, besides the scene, the distribution of θ can be used to distinguish between the LiDAR setups. The distribution of r depends both on the scene and also the LiDAR setup. Since in KITTI there are more points around the ego vehicle than in Lyft (see Fig. 1), the distribution has a higher peak by lower r.

After the translation of KITTI to Lyft, the distribution of θ should be more similar to Lyft than to the original KITTI.

As metric to measure the similarity of two distributions we use the well known Wasserstein distance also known as earth mover's distance [15, 16]. The Wasserstein distance comes from the transportation theory and for histograms it can be intuitively interpreted as cost to transform on histogram into the other.

3 Results

We train CycleGAN on a split of 7000 BEV maps for KITTI and 7000 BEV maps for Lyft. The BEV maps are created for the four level of information **Position**, **Height**, **Height+Density** and **Height+Density+Intensity**.

Figure 2 shows qualitative results of the translation for **Position**. It can be seen that in the translated KITTI BEV maps the number of white pixels close to the ego vehicle is decreased, like in Lyft BEV maps. Also some circle lines in the translated BEV maps are deleted and the number of circle lines match the number from Lyft. Further, most of the shapes are preserved for most cars, but for some cars the number of white pixels decrease. So optically these examples show promising results in the translation of the characteristics of the LiDAR BEV maps.

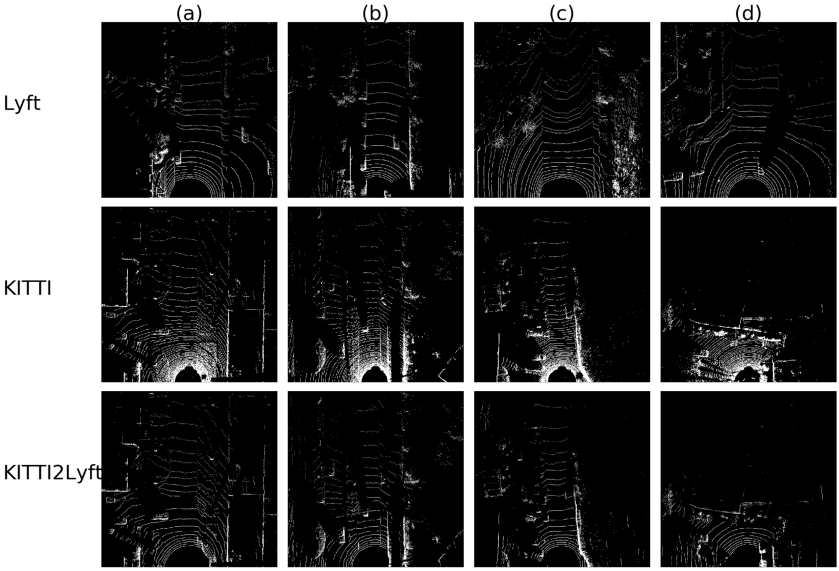

Figure 2: Qualitative results for the translation from KITTI to Lyft, for the information level **Position**.

For quantitative results we use the described evaluation strategies **Usability** and **Structure** (Sec. 2.3).

For **Usability** ComplexYOLO was trained on a Lyft split of 7000 BEV maps for the four different BEVs information levels to detect cars. Table 2 shows the average precision of the detector on test splits from Lyft, KITTI and the same KITTI scenes translated. For real Lyft BEV maps the detector has a good average precision over 0.95. For KITTI BEV maps it drops to an average precision between 0.53 - 0.6. After translation the average precision of the object detector gets a bit worse, probably because some shapes of cars were destroyed. This happened, for example, when translating the BEV map (c) in Figure 2. In the original KITTI BEV map the two cars in Figure 2 (c) are clearly visible, in the corresponding translated map most of the car is missing and could not be detected. Another potential problem is that after the translation pixel with small color values occur around the LiDAR points. These low color artifacts are barely visible to a human. In the first translated

Table 2: Results of the evaluation **Usability**. Average precision (IoU=0.5) for the class car of an object detector that was trained on Lyft Level5 dataset and tested on Lyft Level5, KITTI and KITTI2Lyft translated with CycleGAN.
P = **Position**; H = **Height**; D = **Density**; I=**Intensity**.

	P	H	H+D	H+D+I
Lyft	0.95	0.97	0.97	0.97
KITTI	0.53	0.6	0.54	0.53
KITTI2Lyft	0.52	0.54	0.51	0.47

Table 3: Results of the evaluation **Structure**. Mean Wasserstein distance between different distribution of the spherical coordinates from the datasets.
P = **Position**; H = **Height**; D = **Density**; I=**Intensity**.

	P	H	HD	HDI
$\theta_K \leftrightarrow \theta_L$	0.082	0.12	0.12	0.12
$\theta_{K2L} \leftrightarrow \theta_K$	0.12	0.17	0.17	0.17
$\theta_{K2L} \leftrightarrow \theta_L$	0.06	0.096	0.096	0.093
$\phi_K \leftrightarrow \phi_L$	0.18	0.18	0.18	0.18
$\phi_{K2L} \leftrightarrow \phi_K$	0.20	0.21	0.21	0.20
$\phi_{K2L} \leftrightarrow \phi_L$	0.20	0.21	0.21	0.20
$r_K \leftrightarrow r_L$	4.44	5.35	5.35	5.35
$r_{K2L} \leftrightarrow r_K$	7.18	5.60	6.41	5.25
$r_{K2L} \leftrightarrow r_L$	4.07	3.60	3.88	3.42

map of Figure 2 25454 pixel have a color value between 1 and 10 and 25415 pixel a color value greater than 10. This number of pixels containing small values decreases if we encode more information in the color channels, like in **Height**, **Height+Density** or **Height+Density+Intensity**, but the performance of the detector does not increase as Tab. 2 shows. Also if we set all pixels with a color value smaller than 50 to 0, the AP only growth only a little to 0.53 in the case **Position**. So the influence of these small values can be neglected.

For the criterion **Structure** we translate the coordinate system such that the LiDAR is nearly the origin of the coordinate system and transform the coordinate system to spherical coordinates. We compare the distribution of 100 KITTI BEV maps, the corresponding translated BEV maps and Lyft BEV maps. The

BEV maps are reprojected and transformed to a spherical coordinate system and calculate the mean Wasserstein distance between the distribution of the spherical coordinate values (Tab. 3). According to the results of Table 3 the distribution of r for the translated KITTI BEV maps is always more similar to Lyft than to KITTI and also closer to Lyft, than Lyft to KITTI. The distribution of ϕ does not significantly change after the translation. Interestingly the addition of the information density and intensity has only little influence of the performance.

4 Conclusions and Outlook

In this paper we analyzed the possibility of unsupervised domain adaptation between two LiDAR datasets with different LiDAR setups. The demonstrated method of translating LiDAR BEV maps from KITTI to the structure of the Lyft LiDAR setup using CycleGAN showed optical promising results (Fig. 2). With our evaluation strategy **Structure** we could show that the KITTI point clouds really adopts the LiDAR setup characteristics of Lyft. Nevertheless **Usability** showed that the performance of our tested car detector could not be improved, even got a little bit worse, because some shapes of cars were damaged during the translation process. This confirms the assumption of Wang et al. [7] that the size of cars plays a more important role for the performance of object detectors, than the setup of the LiDAR. To improve the performance one has to improve the preserving of shapes that are interesting for the task, e.g. cars L-shape. But without the additional information to enlarge or shrink the size of cars, the performance of an object detector will likely not get much better. To analyze the influence of different LiDAR setups on object detection further study is needed. The CARLA [13] simulation environment can be used for the study of the setup influence, as we can choose in CARLA the number of channels and thus can record the same scene, with different LiDAR setups. So the object detector can be tested on the same scenes and only the LiDAR setup will change, hence if the setup has no influence the object detector should perform roughly as well, whether the same LiDAR setup is selected as in the training or a different LiDAR setup.

Next to the application of the translation between two datasets with different LiDAR setups, the method could also be used to transfer between different types of sensors. With Astyx HiRes a dataset with high resolution RADAR data is published. But since this is the only existent high resolution RADAR dataset at the moment and contains only a few scenes, training methods like object detection is challenging. With the demonstrated framework of translating the BEV maps, one could try to translate between the RADAR and LiDAR sensors. Because Astyx also provides LiDAR point clouds in addition to RADAR point clouds, one could also use supervised domain adaption methods.

References

[1] A. Geiger, P. Lenz, C. Stiller et al. “Vision meets robotics: The kitti dataset”. In: *The International Journal of Robotics Research* 32.11. S. 1231-1237. 2013.

[2] H. Caesar, V. Bankiti, A. H. Lang et al. “nuscenes: A multimodal dataset for autonomous driving”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* S. 11618-11628. 2020.

[3] R. Kesten, M. Usman, J. Houston et al. “Lyft Level 5 Perception Dataset 2020”. URL: https://level5.lyft.com/dataset/ 2020.

[4] J. Geyer, Y. Kassahun, M. Mahmudi et al. “A2D2: Audi Autonomous Driving Dataset”. In: *arXiv preprint arXiv:2004.06320*. 2020.

[5] S. Agarwal, A. Vora, G. Pandey et al. “Ford Multi-AV Seasonal Dataset”. In: *arXiv preprint arXiv:2003.07969*. 2020

[6] M. Meyer and G. Kuschk. “Automotive radar dataset for deep learning based 3d object detection”. In: *2019 16th European Radar Conference (EuRAD)* S. 129-132. 2019

[7] Y. Wang, X. Chen, Y. You et al. “Train in Germany, Test in the USA: Making 3D Object Detectors Generalize”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* S. 11710-11720. 2020.

[8] J. Zhu, T. Park, P. Isola et al. “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks”. In: *2017 IEEE International Conference on Computer Vision (ICCV)* S. 2242-2251. 2017.

[9] M. Y. Liu, T. Breuel and J. Kautz. “Unsupervised image-to-image translation networks”. In: *Advances in neural information processing systems*. S. 700-708. 2017.

[10] M. Simon, S. Milz, K. Amende et al. “Complex-YOLO: An Euler-Region-Proposal for Real-Time 3D Object Detection on Point Clouds”. In: *Computer Vision – ECCV 2018 Workshops* S. 197-209. 2019.

[11] A. E. Sallab, I. Sobh, M. Zahran et al. “Unsupervised Neural Sensor Models for Synthetic LiDAR Data Augmentation”. In: *arXiv preprint arXiv:1911.10575* 2019.

[12] K. Saleh, A. Abobakr, M. Attia et al. “Domain Adaptation for Vehicle Detection from Bird’s Eye View LiDAR Point Cloud Data”. In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)* S. 3235-3242. 2019.

[13] A. Dosovitskiy, G. Ros, F. Codevilla et al. “CARLA: An Open Urban Driving Simulator”. In: *Conference on Robot Learning* S. 1-16. 2017.

[14] Y. Ganin, E. Ustinova, H. Ajakan et al. “Domain-adversarial training of neural networks”. In: *The Journal of Machine Learning Research* 17.1 S. 2096-2030. 2016.

[15] E. Levina and P. Bickel “The Earth Mover’s distance is the Mallows distance: some insights from statistics”. In: *Proceedings Eighth IEEE International Conference on Computer Vision (ICCV)* 2 S. 251-256. 2001.

[16] Y. Rubner, C. Tomasi and L. J. Guibas “A metric for distributions with applications to image databases”. In: *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)* S. 59-66. 1998.