Graz Symposium Virtual Vehicle 2018

Data-driven Derivation of Requirements for a Lidar Sensor Model

MARTIN HOLDER, PHILIPP ROSENBERGER, FELIX BERT, AND HERMANN WINNER

Institute of Automotive Engineering (FZD), Technische Universität Darmstadt

{holder, rosenberger, winner}@fzd.tu-darmstadt.de ; felix.bert@yahoo.com

May 16, 2018

Abstract

Safety assurance in virtual driving simulation environments requires accurate sensor models. However, generally accepted quality criteria for sensor models do not yet exist. In this work, we investigate the model quality needed for a Lidar sensor model for virtual validation. We seek to answer the question, whether neglecting sensor effects in a simplified sensor model might lead to a measurable difference in performance of the sensor model compared to a real sensor. A data-driven approach has been chosen to identify relevant features for object classification in Lidar pointclouds which need to be accurately represented in simulations. The contribution of our work is two-fold: Firstly, we identify important features for object detection in point clouds from Lidar data. For this, we apply object classification algorithms to pointcloud segments, for which a variety of geometric, stochastic, and sensor-specific features have been calculated. Using filter models, principal component analysis (PCA), and embedded models, each feature is assessed and ranked on an individual basis. Secondly, we derive implications for Lidar sensor models based on our findings. We investigate variations in classification quality by succesively removing groups of features from our feature set. Our results show, that to make sensor models suitable for the validation of object detection algorithms, the accurate representation of simple geometric features in synthetic pointclouds is sufficient in many cases. Our method can also be used to support the derivation of requirements and validation criteria for sensor models.

Keywords: Lidar Sensor Model, Autonomous Driving, Virtual Validation

1 Introduction

The objective of the PEGASUS Project is to establish quality criteria, tools, and methods for the release of highly-automated driving functions [1]. Virtual testing methods have the potential to accelerate the release of automated driving functions [2]. Validation of automated driving functions in simulation, however, requires accurate models of environment perception sensors such as Radar, Lidar, and Camera. Until now, methods for the derivation of requirements for sensor models, eligible for validation in virtual environments, do not exist. In prior work, the dilemma between simulation efficiency and fidelity, depending on the complexity of the sensor models, has been shown [3]. Simulation-based validation tests of automated driving functions need to account for the same uncertainties a real-world driving test would entail. Object-detection by environment perception sensors is affected by uncertainty in three different domains [4]: state uncertainty, existence uncertainty, and class uncertainty. In this work, we focus on Lidar sensors and elaborate on questions regarding the required sensor model quality needed for validating simulations with respect to class uncertainty.

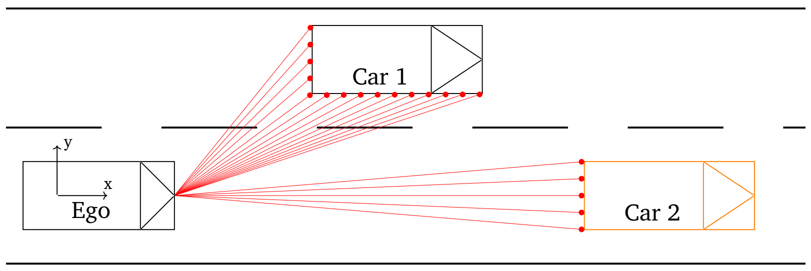


Figure 1: Schematic illustration of object detection with Lidar sensors.

Sensor models have been grouped into three categories, namely ideal models, phenomenological models, and physical models. However, no clear specification of requirements or validation criteria for sensor models are derived along with the models. Ideal models usually generate an object list perfectly perceived from a simulation environment, i.e. no sensor errors are modeled. Additional sensor characteristics such as field of view, limited resolution, and noise on measured quantities may be incorporated, leading to phenomenological models. Here, typical sensor behavior is incorporated in a probabilistic manner inferred from prior observations [5] and expert knowledge [6]. In contrast, physical sensor models aim to reproduce the sensor's raw data based on models of the underlying physical phenomena. This can be realized with ray tracing methods [7] but also deep learning methods have been reported [8]. For Lidar, a physical sensor model solves mathematical equations aiming to correctly model the transmission and perception of pulsed laser light and its reflection in the (virtual) environment.

2 Object Detection with Lidar

Lidar sensors are widely used in automated driving, for example in object detection [9], mapping [10], and localization [11] but also for inferring additional information about the environment, such as determining weather conditions [12]. The sensors emit pulsed infrared light [12] and the distance to hit points is measured by means of the time-of-flight principle. For each measurement cycle, every hit point for which the reflected energy exceeds a certain threshold gets registered and added to a point cloud (sometimes referred to as raw scan). In other words: Whether a point belonging to an object is detected by a Lidar sensor is mainly determined by the object's reflection properties and ultimately the energy of the laser beam reaching the receiver. Both are in turn dependent on incident and emergent angle of the beam as due to reflection properties of the materials involved [13]. Object detection with Lidar is illustrated schematically in Figure 1, where illuminated parts of the cars are registered as points in a point cloud. Scan points belonging to static objects such as guardrails or lane markings are excluded from the figure for clarity reasons.
A typical object detection mechanism with Lidar consists of point cloud segmentation, feature extraction, classification, and tracking. Segmentation algorithms aim to partition a raw scan into disjoint sets of points (segments), such that each object in the original scan is represented by exactly one segment [14]. Ideally, all points belonging to an object and no points belonging to a different object are included in a segment, thus making the ideal segmentation a perfect mapping between scan points and real-world objects. During tracking, a chronological data track of segments is derived and object detection algorithms perform classification of the tracked objects. A large variety of algorithms have been proposed to classify (tracked) objects with Lidar, e.g. Random Forest [15], Support Vector Machines [9], and deep learning methods such as Convolutional Neural Networks [16]. Good performances for the classification task have been reported for Random Forest and Adaboost, both machine learning algorithm, which motivates their use for our work.
Sensor uncertainties, as described above, are present during multiple stages of object detection: State

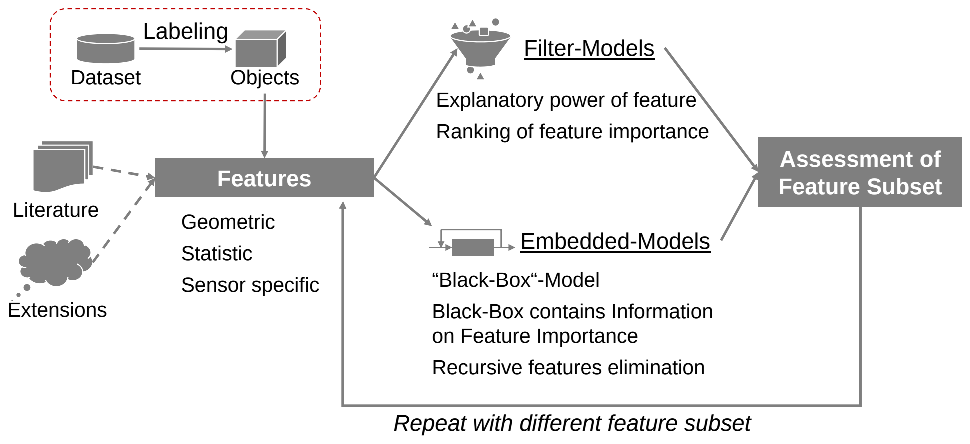
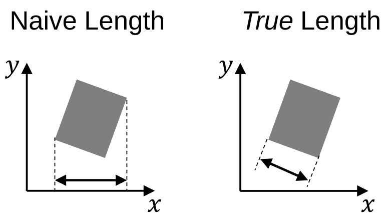

Labeling
Dataset
Objects
Literature
Extensions
Features
Geometric
Statistic
Sensor specific
Filter-Models
Explanatory power of feature
Ranking of feature importance
Embedded-Models
"Black-Box"-Model
Black-Box contains Information on Feature Importance
Recursive features elimination
Assessment of Feature Subset
Repeat with different feature subset

Figure 2: Overview of our proposed method for assessing feature importance: A dataset was recorded and objects have been labeled. Different categories of features are derived from both literature and extension. A set of features is chosen and assessed using filter models. Classifier-based embedded models are trained. Their performance is evaluated before the process is repeated with a different feature subset.

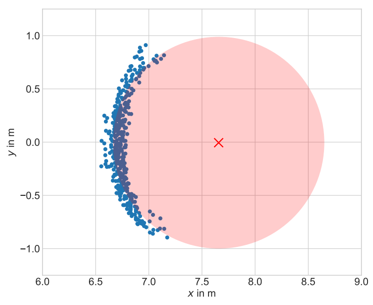
uncertainties are caused by limited resolution and measurement accuracy of the Lidar. Typical values for an ibeo LUX Lidar are distance resolutions of 4 cm with a repeat accuracy of 10 cm in a 1 σ interval. The horizontal working range can be chosen between 85 ° and 110 ° with a resolution of up to 0.125 ° [17]. Existence uncertainties are caused by spurious reflections from terrain, multipath reflections, and other measurement errors such as outliners and false-negative detections occurring if the reflected energy remains under the detection threshold. In contrast, class uncertainties are not directly caused by hardware limitations but rather by negative synergies between the sensor hardware (i.e. produced point cloud) and object classification algorithms. In this work, our main focus lies on the impact the consideration of different segment-level features has on class uncertainties of some exemplary classifiers.

3 Feature Selection

The Lidar processing pipeline described above is well established [18] and shows analogies to data mining approaches [19], which are exploited in this work: Data collection, data preprocessing, and analytical processing steps are considered to correspond to sensing, segmentation, tracking, and classification respectively. This motivates the use of data mining techniques for feature subset selection and for the evaluation of feature importance for the classification task. Our method for the data-driven derivation of sensor model requirements has been applied to data recorded by four-layer Lidar sensors. The process of selecting features and assessing the impact of selecting different subsets of features on classifier performance is illustrated in Figure 2. A shortcoming of this method is that we limit our dataset to reflections above the detection threshold, i.e. the collected data is exposed to a true-negative filter by the sensor as any reflections below the detection threshold are removed.
As such, our focus lies on features that were proposed for a similar type of sensor. Lidar sensors with as many as 128 scan layers have become available recently [20]. Raw scans recorded by this new class of Lidar sensor contain a large amount of scan points and provide a detailed picture of an object's volumetric boundaries in three dimensions. Consequently, new segmentation algorithms and features have been proposed to exploit the additional spatial information encoded in the scans of such high-resolution sensors even better. In comparison, scans recorded with four-layer Lidar sensors are much more sparse, and for most objects, the three-dimensional shape is only captured partially. Thus, recently



(a) Naïve vs. true length along x-axis.



(b) Circularity measures the quality of a fitted circle.

Figure 3: Illustration of geometric features

proposed features might not be suitable in every case. Because of the described differences in sensors, our findings cannot be directly applied to high-resolution Lidar. Nevertheless, our proposed method can be applied to requirements for high-resolution Lidar and other sensor models by expanding the list of features investigated.

3.1 Identification of suitable features

89 features have been included in our analysis, which comprise self-devised features, features found in literature, and features expanding on these. Several of the features originally proposed in literature have been extended to the multi-layer scanning device case to better utilize the additional information. The features can be grouped into three categories:

3.1.1 Geometric Features

For vehicle classification, geometric primitives such as object length, width, and height appear to be a natural choice for discrimination. However, accurately determining these dimensions for a given object is hard, as only the visible (i.e. non-occluded) part of an object is scanned (see Figure 1) and object orientation is generally unknown and hence needs to be estimated as well. To reduce associated errors introduced by (partial) occlusion and object rotation, several approaches have been proposed in literature, many of which rely on the fitting of predefined geometric shapes (e.g. L-shapes [21]) to the object point clouds. In addition, we consider naïve features despite this shortcoming: We use cardinality, or maximum span width of segments, in each dimension as an indicator for object size, width, and height respectively. These naïve quantities also allow us to calculate naïve area (naïve length by naïve width), and naïve volume (naïve area by naïve height). Including these features in our analysis allows for a reasoning about the need for their more sophisticated but computationally more expensive counterparts. Figure 3a depicts our definition of naïve dimensions. On the left, the naïve length for an object is indicated. The error made due to the object's rotation causes noise in the feature space. On the right, the true length of the object is indicated. We approximate the true width and length of an object by a rectangular shape fitted to the object in the least-squares sense (cf. [21]). Another indication for volume expansion is the radius of a least-squares fitted circle, see Figure 3b. Here, the bird's eye view shows a typical point cloud obtained from a vehicle. A second feature based on the fitted circle, the circularity, is obtained by calculating the mean squared error for the obtained parametrization. The circularity is a measure of goodness-of-fit and thus corresponds to the *roundness* of an object. Boundary length, like radius and circularity, was first proposed by [22] and represents the length of a polyline corresponding to the segment as the sum of its adjacent points. We extended boundary length to the case of multiple scan layers.

3.1.2 Statistical Features

Quantitative information about large amounts of data is often encoded as statistical features. Statistical measures such as the mean or standard deviation of a set of observations can be interpreted as a low-dimensional representation of shape-information when based on coordinates. A higher skewness, for example, indicates a higher concentration of scan points on one side of an object and thus might indicate a higher expansion of the object in other dimensions on this side. Similarly, many other statistical features can be motivated by the geometric information they encode.

3.1.3 Sensor Specific Features

The Ibeo LUX Lidar measures up to three reflections in each beam direction. In addition to the coordinates of detected points, the echo-pulse width of each reflection is stored [17]. These sensor capabilities and other specifics of the sensor model allow for the composition of sensor-specific features. Sensor-specific features derived from this sensor include the total number of scan points on an object (i.e. cardinality of the point cloud of an object), number of scan points by distance and characteristics of echo-pulse width (range, histogram, profile) over an object.

3.2 Filter Models

To assess features, filter models rate them by means of mathematical expressions and eliminate low-ranked features against a lower barrier or a fixed number of features [19]. This allows reasoning about the explanatory power of each feature and allows for a ranking of feature importance for classification. Two metrics were selected for use in our filter models: Firstly, Mutal Information (MI) was chosen as the metric for feature importance. For two discrete random variables, MI is generally defined as [23, 24]

$$I(X,Y) = \sum_{i,j} p_{ij} \log \frac{p_{ij}}{p_i(X)p_j(Y)} \tag{1}$$

where $p_i(X) = \Pr(X = x_i)$, and $p_j(Y) = \Pr(Y = y_j)$ denote the marginal distributions of X and Y, and $p_{ij} = \Pr(X = x_i, Y = y_j)$ denotes their joint distribution. For the classification setting, an adaption to the case of one discrete and one continuous random variable is necessary [25]. MI is preferred over other metrics such as the correlation coefficient, as it also renders non-linear relations (e.g higher moments), and over p-values, as MI is independent of sample size [25].
Secondly, Principal Component Analysis (PCA) was performed to derive an additional metric for feature importance. In PCA, the matrix of explanatory variables is transformed, such that the first component corresponds to the direction of maximum variance in feature space (referred to as the first principal component, PC). Consecutive PCs are perpendicular to all prior PCs, and are sorted by decreasing variance. By leaving out PCs with a low contribution to variance in feature space, PCA allows for a (visual) evaluation of class-separability in a lower-dimensional space. Furthermore, the contribution of each feature to the first PC (referred to as *load*) can be interpreted as a proxy for feature relevance [26].

3.3 Embedded Models

In embedded models, classification algorithms are used that inherently allow to rank features based on their contribution on the decision of the final classifier. For example, for ensemble methods utilizing trees as a base learner, the average reduction of the Gini index over all trees can be calculated for splits over a given feature [27]. In contrast to filter models, embedded models assess feature importance relative to other features involved, and not on an individual basis. By iteratively removing features with a small contribution to our strong learner and retraining the model (*recursive features elimination*, RFE),

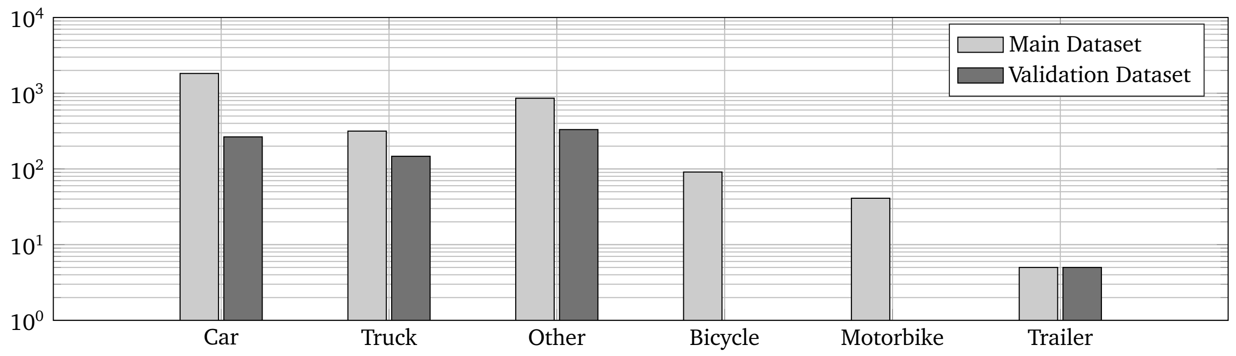


Figure 4: Absolute frequency on a logarithmic scale of class-occurrences in the two datasets. The datasets consists of 3136 and 748 objects, respectively, with an uneven distribution among classes.

we obtain a subset of the best performing features. The further removal of features was stopped when classification accuracy dropped noticeably. Subsequently, results using the Random Forest algorithm are reported. Decision trees were fully developed to reduce bias, as the averaging of trees reduces variance and thus overfitting to the data [28]. Differences in results obtained using the AdaBoost and Random Forest algorithms were negligible for our reasoning.

4 Derivation and Preconditioning of Dataset

In this study, two datasets have been used: Data was collected with a vehicle equipped with three four-layer Ibeo LUX Lidar Scanners in a highway scenario and in an urban traffic scenario (main dataset). Additionally, a validation dataset was created based on a highway test-drive with a different route. The total distribution of classes is shown in Figure 4.

4.1 Segmentation and Labeling

The point clouds have been segmented and labeled according to object classes as standardized by Open Simulation Interface [29]. Rather than segmenting point clouds algorithmically and then labeling the segments, we manually assign points belonging to the same object to segments before labeling. This approach ensures that we are working with a close-to-ideal segmentation. By this, we can exclude additional uncertainties, possibly introduced by the choice of the segmentation algorithm. For the purpose of feature calculation, object point clouds have been completely extracted from raw scans based on labeling information. Again, this choice was made to decouple results from the chosen segmentation algorithm.

5 Evaluation

Feature importance in the main and validation dataset has been analyzed using the models described above. The results from both datasets were consistent with each other to a large extent. Differences arose in the relative importance of layer-specific features compared to each other, as the sensor-alignment has been adjusted between the recording of the main and validation dataset. Repeated RFE with increasing target numbers of remaining features showed that classification accuracy stabilizes with as little as 15 remaining features, tested on the validation dataset. Thus, the ranked feature subset of a Random Forest classifier obtained by RFE with a target number of 15 features is reported. The results are displayed in Table 1, where the 4 scan layers are numbered from 0 to 3, where 0 is the

Table 1: Feature importance ranking for each model

Rank	**Mutal Information**	**Principal Components**	**Random Forest**
1	standard deviation, y	points by distance	mean value, y
2	mean value, y	total number of points	naïve width
3	naïve width	true length	standard deviation, y
4	naïve length	naïve length	echo-pulse width, max
5	true length	true width	boundary length, layer 3
6	points by distance	mean deviation from median	mean deviation from median
7	boundary length, layer 3	standard deviation, x	circularity
8	radius	boundary length, layer 1	naïve length
9	mean deviation from median	boundary length, layer 0	Skewness, x
10	circularity	points per distance	points by distance

topmost layer. The 10 most descriptive features in each model category are predominantly simple features with an easy geometric interpretation, from which the following conclusions are drawn:

- Features that require considerable computation power or appear to be sophisticated do not necessarily outperform simple features.
- The *true* length and width, intuitively a highly descriptive feature of objects, performs worse than naïve dimensions, in particular for Random Forest.
- Features related to Lidar-specific quantities (e.g. echo-pulse width) appear not to play a major role for classification.

To test our assumptions, features have been grouped corresponding to these conclusions, and classification accuracy was 10-fold cross-validated. The results are shown in Table 2 where σ denotes the standard deviation of the cross validation. As a baseline performance, we choose a classifier trained on a dataset comprising all 89 features, leading to 94 % accuracy. For the second group, we excluded features based on echo-pulse width. By this, classification accuracy drops by only 0.5 %. Taking only simple features into account (including echo-pulse width), we observed 93.7 % accuracy, while additionally excluding echo-pulse width lets the performance drop to 92 %. Classification accuracy drops by just 2 % between the full set of features and simple features without echo-pulse width, a difference within 60 % of a standard deviation. These results support our conclusion: Including seemingly sophisticated features and features based on echo-pulse width only leads to a marginal improvement of classification results, even though the number of features in this baseline model is considerably higher than in the model excluding the features.

Table 2: Overall results of classification with different numbers of features. σ denotes standard deviation.

Group (Number of Features)	Accruacy	σ
All features (89)	94	3.1
w/o echo-pulse width (34)	93.5	4.2
simple features (20)	93.7	2.9
simple features w/o (16)	92	3.4

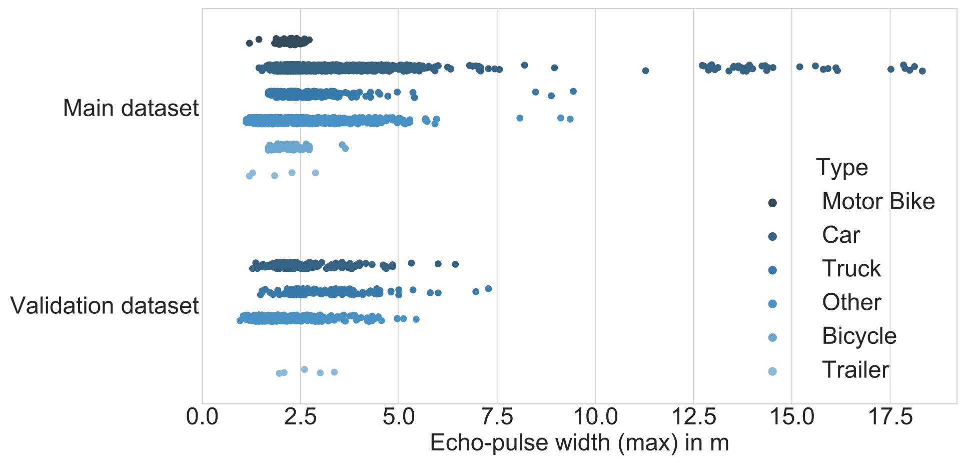


Figure 5: The distribution of echo-pulse width in our datasets indicates a high noise level across the different classes.

6 Implications for a Sensor Model

Our results show that above all, geometric features are the most important type of features for classification of segmented point clouds. In a sensor simulation model, the accurate representation of these features is achieved if a correct representation of the sensor's field of view is implemented. This includes preservation of the assignment of points to a scan device and scan layer as we calculated features depending on this information. Furthermore, given the sensor's tolerances, modelling sensor noise in a sophisticated manner is also required. By this, a correct representation of the features' distance-dependency is guaranteed. In our investigations, we only saw little improvement in classification accuracy when echo-pulse width as a Lidar-specific property was considered to derive features. We argue from Figure 5 that features based on echo-pulse width are not consistent enough for the robust classification of vehicles compared to simple geometric features. Still, echo-pulse width may be seen as a provider of additional information about a scan point: It can only be measured if the point itself is detected. However, a set of detected points allows for the calculation of simple geometric features by itself, which showed to be more consistent. We conclude from this that an explicit model of echo-pulse width is not necessary for high sensor model fidelity. Neglecting echo-pulse width also reduces sensor model validation efforts, as credible models of echo-pulse width require detailed knowledge about detection thresholds, which is often not accessible. Lidar sensor models with ray tracing offer the potential for seamless integration of all features mentioned above. For increased sensor model fidelity, we also recommend a paradigm-shift in modelling the detection mechanism: Rather than assuming all points are detected, ray tracing calculations should be used in a manner of collecting evidence that points do actually exist. This approach renders existence uncertainty in a realistic manner.

7 Conclusion

In our work, we used data mining methods to derive requirements for a Lidar sensor model from datasets. At the example of the Ibeo LUX, we showed a method for deriving feature importance and ranking of feature influence for classification. Our results imply that simple geometric features calculated from segmented Lidar point clouds need to be preserved by a sensor model to obtain high classification accuracy. Our results are in strong correspondence with previously reported work on feature importance for classification with airborne Lidar [30]. Here, it was also found that above all, *simple* features such as the height difference to the ground convey the highest feature importance for classification. We conclude that credible sensor models for virtual validation need to maintain the cardinality in terms of geometric properties of the point cloud rather. This could possibly allow the use of *simple* simulation models of Lidar, which can, due to their simplicity, potentially meet real-time requirements. In future work, we will investigate the transferability of our findings to other sensor types, larger datasets, and to recently reported methods based on deep learning for point cloud segmentation (e.g. [31]).

Acknowledgment

This paper is part of the work in the project PEGASUS funded by the German Federal Ministry for Economic Affairs and Energy based on a decision of the Deutsche Bundestag. The authors would like to thank Lukas Stäcker for his support with data preparation.

References

1. German Aerospace Center. PEGASUS Research Project: Securing Automated Driving efficiently, 2017.
2. Sandeep Sovani. Simulation Accelerates Development of Autonomous Driving. *ATZ worldwide*, 119(9):24–29, 2017.
3. Peng Cao, Walther Wachenfeld, and Hermann Winner. Perception sensor modeling for virtual validation of automated driving. *it - Information Technology*, 57(4):243–251, 2015.
4. Klaus Dietmayer. Predicting of Machine Perception for Automated Driving. In Markus Maurer, J. Christian Gerdes, Barbara Lenz, and Hermann Winner, editors, *Autonomous Driving*, pages 407–424. Springer, Berlin, Heidelberg, 2016.
5. N. Hirsenkorn, T. Hanke, A. Rauch, B. Dehlink, R. Rasshofer, and E. Biebl. A non-parametric approach for modeling sensor behavior. In Hermann Rohling, editor, *2015 16th International Radar Symposium (IRS)*, pages 131–136, Dresden, 2015.
6. Stefan Bernsteiner, Zoltan Magosi, Daniel Lindvai-Soos, et al. Radar Sensor Model for the Virtual Development Process. *ATZelektronik worldwide*, 10(2):46–52, 2015.
7. Michael Gschwandtner, Roland Kwitt, Andreas Uhl, and Wolfgang Pree. BlenSor: Blender Sensor Simulation Toolbox. In *Advances in Visual Computing*, volume 6939, pages 199–208. Berlin, Heidelberg, 2011.
8. Tim A. Wheeler, Martin Holder, Hermann Winner, and Mykel J. Kochenderfer. Deep stochastic radar models. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 47–53. IEEE, 2017.
9. Zeng Wang. *Laser-based detection and tracking of dynamic objects*. Dissertation, University of Oxford, 2014.
10. Matthias Roland Schmid. *Umgebungserfassung für Fahrerassistenzsysteme mit hierarchischen Belegungskarten*. Dissertation, Universitätsbibliothek der Universität der Bundeswehr, 2013.
11. Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. Intelligent robotics and autonomous agents series. MIT Press, Cambridge, Massachusetts and London, England, 2006.
12. Heinrich Gotzig and Georg Geduld. Automotive LIDAR. In Hermann Winner, Stephan Hakuli, Felix Lotz, and Christina Singer, editors, *Handbook of Driver Assistance Systems*, pages 405–430. Springer, Cham, 2016.
13. Kay Ch Fürstenberg and Klaus C. J. Dietmayer. Fahrzeugumfelderfassung mit mehrzeiligen Laserscannern (The Use of Multi-Layer Laser Scanners for Monitoring the Environment of Driving Vehicles). *tm–Technisches Messen/Sensoren, Geräte, Systeme*, 71(3/2004):164–172, 2004.
14. Anh Nguyen and Bac Le. 3D point cloud segmentation: A survey. In *2013 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 225–230. IEEE, 2013.

15. Feihu Zhang, Daniel Clarke, and Alois Knoll. Vehicle detection based on LiDAR and camera fusion. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1620–1625. IEEE, 2014.

16. Bo Li. 3D Fully Convolutional Network for Vehicle Detection in Point Cloud. `http://arxiv.org/pdf/1611.08069v2/`.

17. Ibeo Automotive Systems GmbH. Operating Manual ibeo LUX 2010 Laserscanner, 2012.

18. Shashank Deshpande, Wiktor Muron, and Yang Cai. Vehicle Classification. In Robert P. Loce, Raja Bala, and Mohan Trivedi, editors, *Computer Vision and Imaging in Intelligent Transportation Systems*, pages 47–79. John Wiley & Sons, Ltd, Chichester, UK, 2017.

19. Charu C. Aggarwal. *Data mining: The textbook*. Springer, Cham, 2015.

20. Philip E. Ross. Velodyne Unveils Monster Lidar With 128 Laser Beams. `https://spectrum.ieee.org/cars-that-think/transportation/sensors/velodyne-unveils-monster-lidar-with-128-laser-beams`, 2017. [Online; accessed 28-Nov-2017].

21. Xiao Zhang, Wenda Xu, Chiyu Dong, and John M. Dolan. Efficient L-shape fitting for vehicle detection using laser scanners. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 54–59. IEEE, 2017.

22. Kai O. Arras, Oscar Martinez Mozos, and Wolfram Burgard. Using Boosted Features for the Detection of People in 2D Range Data. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3402–3407. IEEE, 2007.

23. A. Kraskov, H. Stögbauer, R. G. Andrzejak, and P. Grassberger. Hierarchical clustering using mutual information. *Europhysics Letters (EPL)*, 70(2):278–284, 2005.

24. Isabelle Guyon and André Elisseeff. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

25. Brian C. Ross. Mutual information between discrete and continuous data sets. *PloS one*, 9(2), 2014.

26. Carsten F. Dormann. *Parametrische Statistik*. Springer, Berlin, Heidelberg, 2017.

27. Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*, volume 103. Springer, New York, NY, 2013.

28. Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: Data mining, inference, and prediction*. Springer eBook collection Mathematics and statistics. Springer, New York, NY, second edition edition, 2009.

29. Timo Hanke, Nils Hirsenkorn, Carlo Van-Driesten, Pilar Gracia-Ramos, Mark Schiementz, and Sebastian Schneider. Open Simulation Interface: A generic interface for the environment perception of automated driving functions in virtual scenarios: Research Report, 2017.

30. Nesrine Chehata, Li Guo, and Clément Mallet. Airborne lidar feature selection for urban classification using random forests. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38(Part 3):W8, 2009.

31. Jing Huang and Suya You. Point cloud labeling using 3D Convolutional Neural Network. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2670–2675. IEEE, 2016.