Siamese Networks for Online Map Validation in Autonomous Driving

Felix Drost[1] and Luca Parolini[2] and Sebastian Schneider[2]

***Abstract*— Many state-of-the-art autonomous driving systems require prior knowledge in the form of high definition map data, which provide information about the road and its environment. These data can be erroneous or outdated due to the frequency of updates and because of short- or long-term changes in the road. Any autonomous system that relies on such potentially invalid information needs to detect deviations from the map while driving, by the use of its own, or external sensors, a problem often called online map validation.**

This paper proposes a novel approach to online map validation, where map data are compared against the readings of onboard sensor measurements by a deep learning classifier. This classifier is based on the Siamese Network architecture, an architecture known from similarity learning. The classifier is trained on data from real world test drives and evaluated on both correct maps and incorrect maps as found in construction sites. Results show that the classifier reaches an F1 score of 89.1 %, whereby misclassified scenes mostly stem from the limited variability in the training data and the lacking evidence of construction sites in the input data.

I. Introduction

In most autonomous driving systems high definition (HD) map data are a critical component for building a reliable environment model [1]. Especially in urban scenarios the field of view (FOV) of sensors is often limited due to dense traffic or surrounding infrastructure. HD maps provide pre-generated information with a precision of 10 cm to 20 cm [2]. At the same time, they are not affected by dynamic environment conditions and provide a much larger FOV, compared to onboard sensors. These maps include geometric information about the road such as the lane structure, and the position and type of traffic signs, as well as semantic information such as the speed limit and right-of-way.

In an autonomous driving system, HD maps are used for localization, object detection and prediction as well as trajectory planning. Since standard Global Positioning System (GPS) devices do not provide sufficient accuracy, environment features such as landmarks, which are detected at run time by the sensors of the vehicle, are matched against the information of the map. This leads to a localization accuracy within 10 cm [3]. Object detection and prediction algorithms rely on lane and road data for initialization of new target tracks, as well as for following given lanes [4].

However, with the exception of Simultaneous Localization Mapping (SLAM) [5], map data are the result of data collection, processing and distribution, all of which add to a delayed availability of the map data. These delays can easily be in the order of days, weeks and even months, thereby increasing the likelihood that the map data and the real world have deviated. In 2012 alone, there have been 20 000 construction sites in the larger metropolitan area of Munich, Germany, some of which have altered the entire road geometry [6].

While SLAM-algorithms provide accurate and up-to-date maps of the environment, for use cases such as RoboTaxis the use of HD map data is usually favored, as they are the result of a fleet of vehicles. As such, they allow the data to be much more thoroughly processed, which is not bound to real-time constraints, and additionally, offers the possibility for the map data to be manually corrected, enriched or annotated [7], [3].

Consequently, there is a need for an autonomous vehicle (AV) to continuously assess the validity of the map data using the onboard or external sensors.

II. Related Work

Previous work focuses mainly on either the creation of HD maps for autonomous driving by SLAM-algorithms [5] or the generation of online road models [8]. However, the validation of existing maps is only sparsely represented in the research community. Map validation can be considered the process of determining the similarity between the HD map data and the environment depicted by the sensors of an AV. To the best of our knowledge, this has not yet been conducted for this specific purpose. Nonetheless, there exists a variety of work in image comparison with data driven approaches.

A. Map Validation

HEREMaps generates HD maps by using a fleet of industrial capture vehicles, which records their environment with a high-end sensor setup [9]. Since scanning large areas frequently in order to detect changes is not feasible, the map is validated by vehicles with a lower cost stereo or single-view cameras. Object detection algorithms predict features that are present in HD maps. A quality index for existence, position and type of these landmarks is assigned by fusing the results of multiple recordings. While erroneous map regions can be detected by such an approach, there is still a non-negligible delay until this information is available to the AVs in operation.

Bittel et al. develop a model in [10] to determine road parameters such as the number of lanes and the lateral distances to the lane boundaries during an autonomous drive. The model is based on a convolutional neural network (CNN)

[1]Felix Drost was a student at the Dept. of of Computer Science, Technical University of Munich, Arcisstraße 21, 80333 München, Germany felix.drost@tum.de

[2]Luca Parolini and Sebastian Schneider are with the Research and Development Department of BMW AG, Alfred-Nobel-Straße 3, 85716 Unterschleißheim, Germany luca.parolini@bmw.de, sebastian.sb.schneider@bmw.de

that directly predicts these parameters from the various environment representations of the AV. The results can be compared to data from the HD map in order to perform map validation. While this provides a first approach towards online map validation, it is limited to a predetermined number of road parameters. Therefore, invalid map data can only be detected if they can be represented by these parameters.

B. Similarity Learning

In the past years, deep learning has made significant advancements in many computer vision tasks such as image classification or object detection. One special topic in this field is the task of Similarity Learning: two images are compared whether they show the same content or not.

Bromley et al. introduced Siamese Networks first in [11] in order to verify signatures. Two input images are processed independently by two neural networks with the same architecture. The resulting features are joined only at the output layer of the networks, where the similarity is measured by a distance metric such as the cosine distance. The weights are constrained to be identical between both networks, also called weight sharing. Often Siamese Networks are trained with a contrastive loss function. However, using other loss functions, such as the triplet loss, can lead to better performance [12].

Mou et al. propose Pseudo-Siamese Networks in [13] for comparing high resolution optical imagery with satellite RaDAR scans in order to find corresponding patches. Due to the different modalities between the input domains, the constraint of weight sharing is lifted. Furthermore, a decision head consisting of fully connected layers is used instead of the distance metric for comparing the features of both images.

III. Dataset

In order to train a deep learning classifier to perform map validation a dataset is needed, which contains sensor measurements of the vehicle and the corresponding map data from this location. Therefore, data from test drives on two routes are collected and processed into an image format. In order to ensure independence between training and testing data, both routes do not overlap. The datasets from these routes consist of 21 509 and 5647 samples, respectively.

A. Data Representation

Each sample (see Fig. 1) contains the following representation of its environment in images with height and width of 256 px. Most of the input modalities are represented as grids from the bird's-eye perspective, which indicate whether a cell in space around the AV is occupied or is classified to a grid specific state [14].

A **map** region (Fig. 1(b)) around the AV is extracted from the HD map data based on its GPS position and plotted to a three channel image. While the map data contains additional information such as landmarks, here only the geometry of the road is used. In the image the road space is depicted in green and the lane boundaries in blue.

An intrinsically and extrinsically **calibrated camera** (Fig. 1(c)) facing towards the front of the AV is used to collect the visual information of the street environment. While this information is easy to interpret for a human observer, computationally extracting the road shape is a challenging task.

The **free space** (Fig. 1(d)) indicates the perceived driveable space of the AV. This space may be limited due to dynamic and static obstacles that occlude the sensor FOV. The driveable space correlates strongly to the lane space of the map.

The **space classification** (Fig. 1(e)) provides the occupancy state. States include free space and dynamic, static, and unknown occupancy. It combines the information of the other grids and, thus, potentially provides most knowledge of the road geometry in the scene.

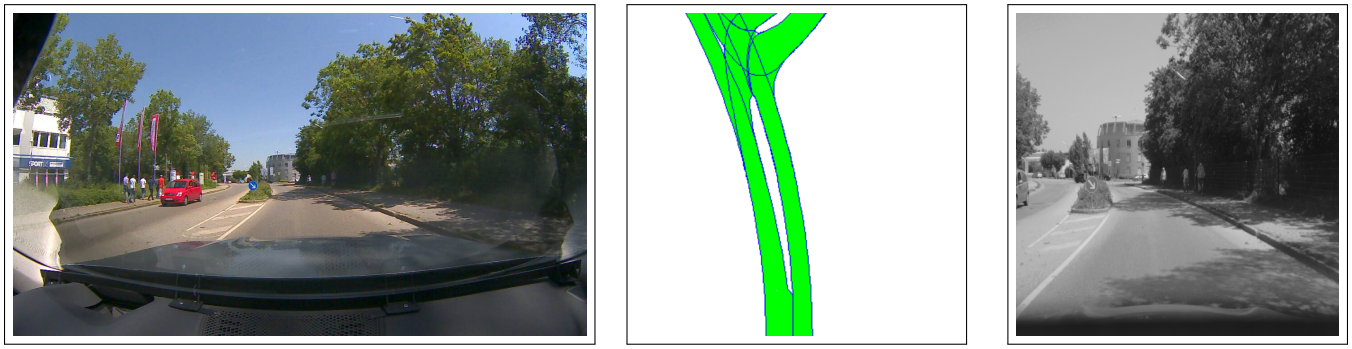
The **static evidence** (Fig. 1(f)) of obstacles near the road are collected by LiDAR and RaDAR sensors. This includes buildings, vegetation, and parked cars. The information of both sensors is fused into a grid representation, where the value of a cell indicates its occupancy. Static evidence shows a clear outline of the road geometry when compared to the map.

The **dynamic evidence** (Fig. 1(g)) of traffic participants is derived by the sensor fusion. Their speed and position are estimated over time to gather motion information. The behavior of other vehicles provides insight about the road structure. For example a vehicle that drives on a road boundary indicated in the map over an extended period of time can indicate an invalid section.

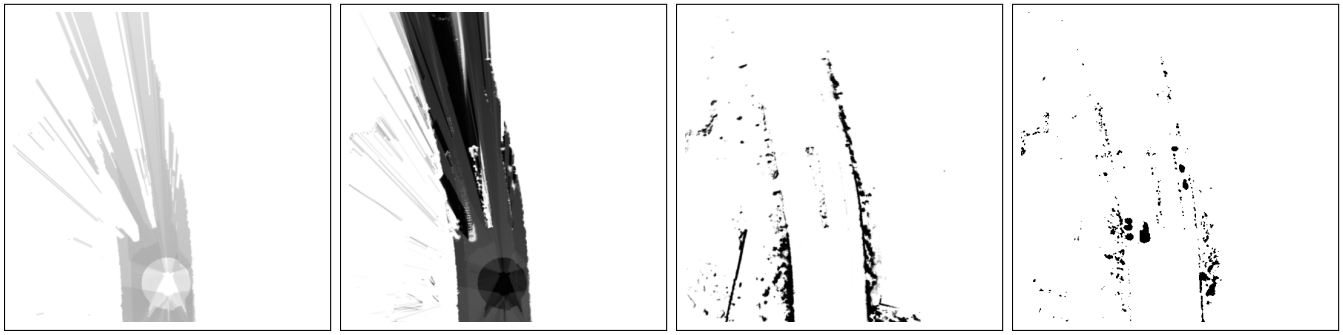
All environment grids result from measurements of multiple sensors, which are fused over time. For map and grid representation, the scale is chosen so that the images show 100 m in height and width, 85 m to the front and 15 m to the rear of the car. While only the validity of the road ahead is of interest during an autonomous drive, a previous map error can provide evidence as well. When driving at a velocity of $50\,\mathrm{km\,h^{-1}}$ in an urban environment, the head-on FOV is therefore approximately six seconds. The map and grid data are rotated in a way so that the driving direction points towards the top of the image. Due to this alignment between the grid and the map, no additional transformations have to be learned by the neural network.

B. Split and Labeling

In order to train supervised models for binary classification, positive and negative samples are needed. It is difficult to collect a large amount of negative samples, as, in most cases, the map data fits the environment data of the same time stamp. Therefore, negative training samples are created artificially by matching map data from different time stamps and GPS positions than the sensor data. In Fig. 2 it can be seen that the map is invalid with respect to the environment. In this fashion, the training and validation datasets are created purely from shuffled samples of the first route with a ratio of 80% and 20%.



(a) Dash-board camera (b) Map data (c) Calibrated camera



(d) Free space (e) Space classification (f) Static evidence (g) Dynamic evidence

Fig. 1. The images of the map and environment data are used for training the Siamese Network. A dash-board images is included as a reference image.

For evaluating the classifier, a test set containing valid and invalid data is needed. The recordings from a second test route are used to generate 2362 valid samples. While pseudo-labels can be used for training, evaluation conducted on them does not ensure correct behavior of the classifier during employment in an AV. Therefore, the invalid test set consists of samples that include construction sites. While they are not the only source of invalid map data, construction sites provide a large amount of scenes, where the environment and the map data do not match. In Germany, construction sites are often indicated by yellow lane markings or beacons. Therefore, object detection is performed on the front camera image in order to filter each sample, which includes these features. Hereby, a total of 3285 samples showing construction sites is collected. The final evaluation is performed on 80% of this dataset. The remaining 20% is used to detect convergence and to determine an optimal decision threshold. The classifier should therefore detect invalid map data caused by a construction site, even though it has never seen one during training.

IV. Methodology

A classifier based on the Siamese Network architecture is trained on this dataset. Its architecture can be seen in Fig. 3. A convolutional body, which consists of two independent CNNs with the same architectural design, processes two images into a feature space. Due to the different modalities in both input images, the networks do not share weights following [13]. Subsequently, the features of both input images are compared in a decision head, which predicts a binary value according to whether the map corresponds to the environment.

TABLE I

Architectural Desing of the CNN.

layer	filters	output size	parameters
1	5x5x64	256x256	4800 / 1600*
2	5x5x64	128x128	102 400
3	5x5x32	64x64	51 200
4	5x5x32	32x32	25 600
5	5x5x16	16x16	12 800
6	5x5x16	8x8	6400

* The parameter count differs between single- and three-channel input.

A. Body

The feature extractors consists purely of convolutional layers following Springenberg in [15]. In order to reduce the size of the feature space, a stride of two is used instead of maxpooling layers. As shown in Table I, each feature extractor consists of 6 convolutional layers with a mask width and height of 5 px. In order to reduce the amount of features, which are compared by the distance metric, the amount of filters is divided in halves after every second convolutional layer. Thus, the amount of trainable parameters remains comparatively low.

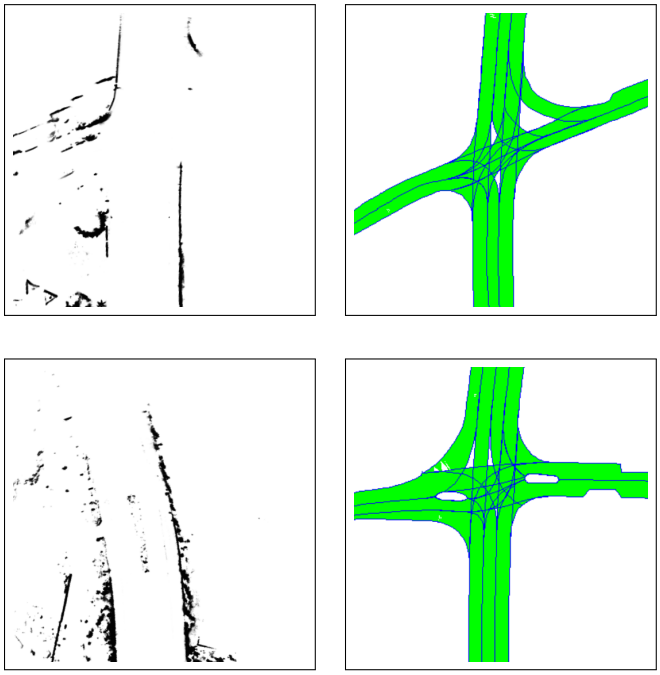


Fig. 2. The top row shows a valid sample consisting of environment data (static evidence) and its corresponding map. In the bottom row map data from an other position is assigned in order to generate an invalid pairing.

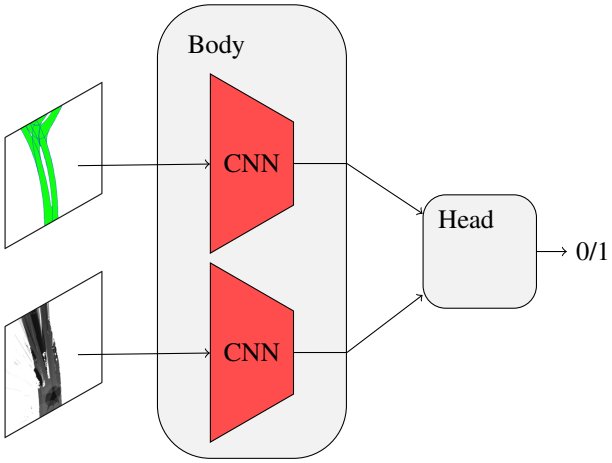


Fig. 3. The Siamese Network takes map and environment images as input data in order to perform binary classification. The architecture consists of a convolutional body and a decision head.

B. Decision Heads

Three decision heads with varying loss functions are implemented and evaluated. Following [11], one variant of the decision head is trained on the contrastive loss function. In order to compare the content of images X_1 and X_2 features $D(X)$ the comparison is conducted by using the Euclidean distance. The label y^* indicates a positive or negative match by the value of one or zero respectively. The contrastive loss function $\mathcal{L}_{cont}$ is given by

$$\begin{aligned} \mathcal{L}_{pos} &= ||D(X_1) - D(X_2)||^2 \\ \mathcal{L}_{neg} &= \max\left(0, m^2 - ||D(X_1) - D(X_2)||^2\right) \\ \mathcal{L}_{cont} &= y^* \mathcal{L}_{pos} + (1 - y^*)\mathcal{L}_{neg}, \end{aligned} \tag{1}$$

where the positive loss $\mathcal{L}_{pos}$ minimizes the distance between similar images in an embedding space and the negative loss $\mathcal{L}_{neg}$ maximizes the distance between different images up to a margin m.

The triplet loss function is implemented as another variant. It uses an anchor image X_a, a matching image X_p, and a negative image X_n. While the Euclidean distance performs better for the contrastive loss, the Cosine similarity

$$\text{Sim}_{\cos}(x, y) = \frac{\mathbf{x} \cdot \mathbf{y}}{||\mathbf{x}|| * ||\mathbf{y}||} \tag{2}$$

is used for the triplet loss function, following Nguyen and Bai in [16], which measures the angle between two input vectors x and y. The loss function

$$\begin{aligned} L(X_1, X_2) &= \text{Sim}_{\cos}\left(D(X_1), D(X_2)\right) \\ \mathcal{L}_{trip} &= \max\left(0, L(X_a, X_p) - L(X_a, X_n) + m\right). \end{aligned} \tag{3}$$

is pushing corresponding images closer in the embedding space, while the distance between the anchor and the negative image at the same time.

For training on the contrastive and triplet loss functions, the features of each input modality are subsequently fed into two fully connected layers of sizes 256 and 32.

Following [13], the last decision head variant does not compute a distance metric, but the features of the two convolutional streams are combined directly using three fully connected layers with sizes 512, 256, and 1. Further, it is trained on the binary cross-entropy (BCE) loss

$$\mathcal{L}_{BCE} = -y^* * log(y) - (1 - y^*) * log(1 - y), \tag{4}$$

where y^* indicates the correspondence to one of two classes [17].

C. Training Details

For this work, no pre-trained models are utilized. While using pre-trained models in general results in a decrease of training time, they cannot be applied in this case, as no pre-trained models have been found with input images similar to the grid representation that is used as one of the inputs in this paper. Therefore, each variant of the Siamese models are trained fully from scratch in an end-to-end fashion using the Adam Optimization Algorithm [18]. Training is conducted until convergence on the real-world validation set.

As described in section III-B, the task during training and testing differs due to the use of pseudo-labels. Regularization techniques, which are commonly used to combat overfitting, show good results in decreasing the performance gap between predicting samples with pseudo-labels and samples from the real-world test set. Normal data augmentation such as image rotations, flips, and scaling is largely not applicable to this case. The input images are from calibrated sensors. Therefore, such disturbances do not occur in a real-life

TABLE II

EFFECTS OF DIFFERENT DECISION HEAD VARIANTS ON THE PERFORMANCE OF THE SIAMESE NETWORK TRAINED ON FREE SPACE CLASSIFICATION GRIDS. ALL VALUES ARE GIVEN AS PERCENTAGE.

loss	validation	accuracy	precision	recall	F1-Score
Contrastive	93.9	61.3	**95.5**	35.2	51.4
Triplet	95.6	57.9	73.9	42.6	54.1
BCE	**97.1**	**67.9**	75.3	**66.8**	**70.8**

TABLE III

EFFECTS OF DIFFERENT ENVIRONMENT REPRESENTATIONS ON THE PERFORMANCE OF THE SIAMESE NETWORK TRAINED ON THE BCE LOSS FUNCTION. ALL VALUES ARE GIVEN AS PERCENTAGE.

modality	validation	accuracy	precision	recall	F1-Score
Camera	95.7	65.3	68.0	76.2	71.9
Free Space	97.1	67.9	75.3	66.8	70.8
Classification	**97.2**	73.0	76.6	77.2	76.9
Static Grid	95.7	**87.3**	**89.1**	**89.0**	**89.1**
Dynamic Grid	50.3	53.0	60.9	53.9	57.2

scenario. Gaussian noise is added randomly to the input images during training in order to simulate sensor noise. Furthermore, dropout is used in between the fully connected layers of the decision heads with a rate of 0.5. Following the original publication [19], dropout is also used between the convolutional layers with a lower rate of 0.1.

V. RESULTS

The approaches are evaluated on the pseudo-label and the real-world dataset. The performance is measured based on accuracy, precision, recall value and the F1-score. In the following text, a sample that is truthfully predicted as invalid map data is considered as true positive. In the first stage of experiments, the different decision heads are evaluated. The results can be seen in Table II. All three models show good performance on the pseudo-label dataset, which shows that Siamese Networks can be used to differentiate between matching and non-matching content of the map and the environment. Surprisingly, only the Siamese Network trained on the BCE loss performs well with regard to the real-world test set, while the other variants predict only slightly better than random guessing. During the experiments, Siamese Networks trained on the contrastive and triplet loss function converged considerably faster than when training on the BCE loss. Therefore, the Siamese Networks specializes faster on the training task, which differs from the testing task.

In the second stage of experiments, the different input modalities are tested on a Siamese Network that is trained on the BCE loss. The results are provided in Table III. The results indicate that the camera image, free space, space classification, and static evidence grid can be used to some degree for the purpose of map validation. On the contrary, the dynamic evidence does not seem to provide sufficient information about the geometry of the streets. Particularly in situations without dense traffic, dynamic evidence is sparse and thus cannot be used for map validation. While camera images are the easiest input for a human observer to interpret, the complexity of the domain of natural images cannot be efficiently trained on this dataset. It remains a topic of further research as to whether this input modality can be used when training on a larger training set. Out of the remaining input modalities, the static occupancy grid performs best with a F1-score of 89.1 %. The evidence in this input source strongly resembles the lane boundaries of the streets. Therefore, we can assume that the correlation between map and static evidence can be learned efficiently by the Siamese Network.

In order to gain insights in the shortcomings of the Siamese Network the images from the dash-board camera are inspected for misclassified samples. Interestingly, the network does not misclassify single images, but rather scenes of consecutive samples. Therefore, typical error sources can be identified. For false positives, three scenarios are commonly miss-classified. The first case consists of parked cars on the roadside as they block the FOV of the sensors. In the second scenario, the car is located at large intersections where there is a lack of static evidence. The last case is when there are cases when there is a grass strip adjacent to the road. The occurrence of these errors can be explained by the absence of such scenes in the dataset. Naturally, the Siamese Network is not able to predict well in scenes that do not frequently appear in the training set. This suggests that the performance of the network can be further improved by adding a larger variety of different scenes and environment conditions to the dataset.

False negatives can be classified into three scenarios as well. First, the Siamese Networks have a low sensitivity when leaving a construction site. Second, scenarios where the width of the road is only reduced slightly are not detected. Finally, the performance of the Siamese Network is not reliable in intersections, where static evidence is rare. All these problems can be solved by using higher level abstractions of visual clues, such as using detected lane markings as an additional input modality.

VI. CONCLUSION

In this work, we have tackled the problem of Online Map Validation. We created a dataset containing sensor and map data based on records from test drives, which is used in order to train deep learning classifiers. The Siamese Network architecture is adapted for comparing the environment of an AV to the corresponding map data. The models are evaluated against samples of valid and invalid maps. The best model detects outdated map data at construction sites with an accuracy of approximately 90 % without being shown one during training. Developers at the research and development department of BMW AG are continuing the efforts in the development of an online map validation component based on the ideas discussed in this paper.

While this result shows that deep learning classifiers, particularly Siamese Networks, can be used for performing map validation, the performance is not yet sufficient in order to be employed in an AV. An analysis of typical error

sources indicates that misclassifications can be explained by shortcomings in the dataset. Further, the training on pseudo-labels does not reflect the task, which the model performs during the evaluation.

Although better results can be made by a more elaborate architecture search, future work should focus on the creation of a more diverse dataset that contains a broader variety of road scenarios. Further, the classifier can be further refined by training on invalid samples that correspond more accurately to actual invalid map data. Another critical point that should be addressed by future research work is the training of the classifier on a dataset created from a variety of sensors, which could differ from those used at run time. This would allow car manufacturers to manage a single data set for map validation and to provide online map validation functions on multiple vehicles each having a different sensor setup.

REFERENCES

[1] Wolfgang Kühn, Michael Müller, and Tom Höppner. Road data as prior knowledge for highly automated driving. *Transportation Research Procedia*, 27:222–229, 01 2017.

[2] Kichun Jo, Chansoo Kim, and Myoungho Sunwoo. Simultaneous localization and map change update for the high definition map-based autonomous driving car. *Sensors*, 18:3145, 09 2018.

[3] Heiko Seif and Xiaolong Hu. Autonomous Driving in the iCity—HD Maps as a Key Challenge of the Automotive Industry. *Engineering*, 2:159–162, 06 2016.

[4] Ba-Ngu Vo and Wing-Kin Ma. The gaussian mixture probability hypothesis density filter. *IEEE Transactions on Signal Processing*, 54(11):4091–4104, 2006.

[5] H. Lategahn, A. Geiger, and B. Kitt. Visual slam for autonomous ground vehicles. In *2011 IEEE International Conference on Robotics and Automation*, pages 1732–1737, May 2011.

[6] Baureferat der Landeshauptstadt München. Jahresbericht 2012 2013. Technical report, 2013.

[7] Alex Zolotovitski, Yakov Keselman, James Lynch, and Scott Williamson. Analysis of potential to improve maps using car probe data. In *Proceedings of the 10th ACM SIGSPATIAL Workshop on Computational Transportation Science*, pages 24–29, 2017.

[8] E. Casapietra, T. H. Weisswange, C. Goerick, F. Kummert, and J. Fritsch. Building a probabilistic grid-based road representation from direct and indirect visual cues. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 273–279, June 2015.

[9] Stephen O'Hara. In-vehicle change detection for self-healing hd maps. http://on-demand.gputechconf.com/gtc/2018/presentation/s8834-in-vehicle-change-detection-closing-loop-car.pdf. HERE Technologies.

[10] S. Bittel, T. Rehfeld, M. Weber, and J. M. Zöllner. Estimating high definition map parameters with convolutional neural networks. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 52–56, Oct 2017.

[11] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 737–744. Morgan-Kaufmann, 1994.

[12] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823. IEEE Computer Society, 06 2015.

[13] Lichao Mou, M. Schmitt, Yuanyuan Wang, and Xiao Xiang Zhu. A cnn for the identification of corresponding patches in sar and optical imagery of urban scenes. In *2017 Joint Urban Remote Sensing Event (JURSE)*, pages 1–4, 03 2017.

[14] G. Tanzmeister and D. Wollherr. Evidential grid-based tracking and mapping. *IEEE Transactions on Intelligent Transportation Systems*, 18(6):1454–1467, 06 2017.

[15] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*, 05 2015.

[16] Hieu V. Nguyen and Li Bai. Cosine similarity metric learning for face verification. In *Proceedings of the 10th Asian Conference on Computer Vision - Volume Part II*, ACCV'10, pages 709–720, Berlin, Heidelberg, 2011. Springer-Verlag.

[17] Katarzyna Janocha and Wojciech Czarnecki. On loss functions for deep neural networks in classification. *Schedae Informaticae*, 25, 02 2017.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.