Map-aided annotation for pole base detection

Benjamin Missaoui[1] Maxime Noizet[1] Philippe Xu[1]

***Abstract*—For autonomous navigation, high definition maps are a widely used source of information. Pole-like features encoded in HD maps such as traffic signs, traffic lights or street lights can be used as landmarks for localization. For this purpose, they first need to be detected by the vehicle using its embedded sensors. While geometric models can be used to process 3D point clouds retrieved by lidar sensors, modern image-based approaches rely on deep neural network and therefore heavily depend on annotated training data. In this paper, a 2D HD map is used to automatically annotate pole-like features in images. In the absence of height information, the map features are represented as pole bases at the ground level. We show how an additional lidar sensor can be used to filter out occluded features and refine the ground projection. We also demonstrate how an object detector can be trained to detect a pole base. To evaluate our methodology, it is first validated with data manually annotated from semantic segmentation and then compared to our own automatically generated annotated data recorded in the city of Compiègne, France.**

I. Introduction

High definition (HD) maps provide a rich prior knowledge to automated navigation systems. Many different types of information are encoded in HD maps, often organized in layers. The road topology layer helps decision-making and motion planning while the features layer can be used for localization. For the latter, there are various kinds of HD maps.

One type of maps consists in low-level features such as point clouds or image-based key points/frames. These types of maps are often constructed from a SLAM perspective and their use may be restricted to the use of sensors similar to the ones employed during the mapping phase. Another constraint comes from the scalability of such maps as they are often heavy in terms of storage. However, one main advantage is that the mapped features being essentially geometric, the map can be constructed with a low level of human intervention.

Another type of maps consists in so-called vector map, which encodes features at a higher semantic level. The features are typically road infrastructure such as traffic signs, traffic lights, road markings or sometimes building footprints. The geometry of these features are often vectorized in 2D with geometric primitives such as points and lines. This enables these maps to be relatively light and can be easily scaled up at a city level. Contrary to the previous case, these maps often require a higher amount of intervention from a human operator and could be harder to update automatically.

In this work, we consider only vector maps as they are sensor agnostic and could be provided in generic frameworks such as OpenStreetMap. The information encoded by road features such as traffic signs and traffic lights can be used for navigation but the knowledge of their spatial position can also be used for localization purposes. More generally, all georeferenced features can serve as landmarks to estimate the vehicle pose. To do so, the vehicle needs to be able to perceive these features from its own embedded sensors. Figure 1 shows an HD map of the city of Compiègne, France, containing several types of point features such as traffic signs or traffic lights among others.

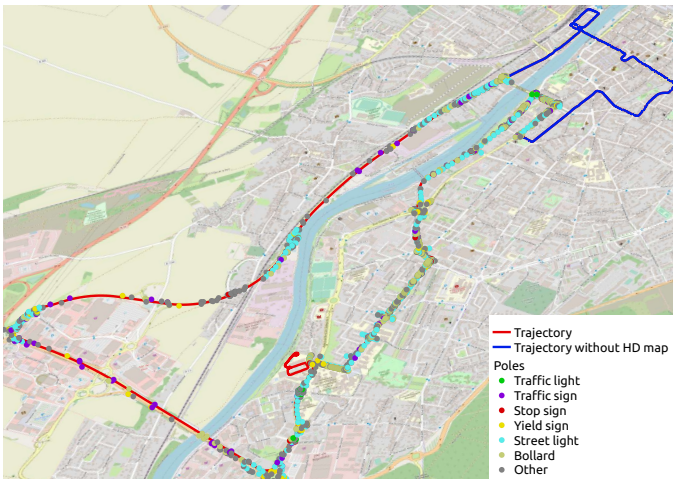

Fig. 1: HD map of the city of Compiègne, France, containing generic pole-like elements such as traffic signs, bollards or street lights.

Lidars and cameras are the most common sensors used in the field of intelligent vehicles. Image-based methods for object detection or semantic segmentation have seen a tremendous improvement in past years thanks to deep neural networks. A key element for these methods to work is to have a large amount of annotated data.

We propose in this paper a method to make use of HD maps for automatic annotation of images. In particular, we show how the concept of pole base can be defined from road features and how they can be detected from an object detection point-of-view using state-of-the-art deep learning methods. The contributions of this paper are as follows:

- definition of a pole base class from HD map pole-like features
- projection, refinement and filtering of image annotations by the use of a lidar sensor
- bounding box annotations for object-based detection

[1]The authors are with the Université de technologie de Compiègne, CNRS, Heudiasyc, France. name.surname@hds.utc.fr

- experimental validation using both manually annotated data from semantic segmentation datasets and from real data and an HD map in the city of Compiègne, France

The rest of the paper is organized as follows. First, some related works are presented in section II. In section III, we introduce how HD map features can be projected onto images for automatic annotations and why additional refinement and filtering using a lidar are needed. To compare with manually annotated data, we show in section IV how semantic segmentation data can be used to achieve similar annotations. Next, in section V, we introduce the use of a bounding-box based object detector for pointwise detection. Finally, experimental results are detailed in section VI, first using data from semantic segmentation datasets and then from real unannotated data along with an HD map acquired in the city of Compiègne.

II. Related works

The use of pole-like features in localization context is fairly common [1]–[3]. The most common sensors used in intelligent vehicles context are cameras and lidars. For lidars, geometric assumptions can be used in order to build pole-like object detectors [4]–[6]. The lidar point intensity can also be used to detect reflective traffic signs [7]. Contrary to lidars, modern camera-based detectors mostly rely on machine learning techniques especially through the use of deep neural networks. Therefore, they heavily rely on available annotated data. The detection of road features such as traffic signs or traffic lights can be made using object detection methods requiring bounding box annotations. Many generic object detectors are trained from datasets such as the Microsoft COCO [8] or more specialized one such as KITTI [9]. The detection of pole-like features, however, such as street lights is often considered from semantic segmentation point-of-view. This requires pixel-level annotations which are very costly. Several large scale semantic segmentation datasets are nevertheless available such SemanticKITTI [10], Cityspaces [11] or BDD100K [12].

In order to be less reliant on manually annotated data, Dong et al. [13] proposed to use range images constructed from a lidar in order to detect pole-like features which then served as pseudo-labels to train a deep neural network. Sun et al. [14] used an HD map to generate image-level labels, e.g., number of lanes in an image. Lee et al. [15] proposed a semi-automatic traffic landmark annotation using 3D road features from an HD map. Their goal was to provide initial annotations to accelerate a human annotation process. In this paper, we aim at using 2D HD maps to automatically annotate pole-like features in images while using an object detection approach.

III. HD map features projection and filtering

Throughout this paper, the map features are considered to be encoded as georeferenced two-dimensional points without height information. In order to apply geometric transformation to these point features, their geodetic coordinates, i.e., longitude and latitude, are first converted into an East-North-Up (ENU) coordinates frame with its origin being in the vicinity of the driving sequences. Note that the "up" coordinate can be ignored as the altitude is set arbitrarily. A map feature is therefore represented as a 2D point ${}^{\mathrm{M}}P$ with coordinates $\left[{}^{\mathrm{M}}x, {}^{\mathrm{M}}y\right]^\top$ expressed in the map frame denoted by the superscript M. The map is therefore a set of n 2D points ${}^{\mathrm{M}}\mathcal{M} = \left\{{}^{\mathrm{M}}P_i\right\}_{i=1,\dots,n}$.

A. Projection of pole base onto image frames

In order to project the map features onto the image frame, they are first expressed in the vehicle frame which is defined as the body frame of the vehicle IMU placed above the center of the rear axle. A map feature point ${}^{\mathrm{V}}P$ in the vehicle frame is related to ${}^{\mathrm{M}}P$ as follows

$$\begin{bmatrix} {}^{\mathrm{V}}x \\ {}^{\mathrm{V}}y \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} {}^{\mathrm{M}}x - x \\ {}^{\mathrm{M}}y - y \end{bmatrix}, \tag{1}$$

where $[x, y]^\top$ is the coordinates vector of the vehicle and θ its heading. Note that the accuracy of the coordinates of ${}^{\mathrm{V}}P$ gets more sensitive with respect to the heading estimate θ when the map features ${}^{\mathrm{M}}P$ gets further away from the vehicle position.

Because there is no height information available, it is not possible to project the map features directly onto an image. Instead, we decided to consider all the map features at ground level. That is to say, regardless of the true height of the map feature, we will only consider its projection onto the ground. For pole-like features such as bollards or street lights, it will correspond to their base, i.e., the contact point with the ground. For traffic signs, most of them are actually fixed on top of poles although these poles may not be mapped explicitly. Therefore, all the map features will be considered as represented in the image frame by their respective pole base point.

By assuming that the x-y plane of the vehicle frame is parallel to the ground and at fixed known height h, a map feature can be set at the ground level by setting

$${}^{\mathrm{V}}z := -h. \tag{2}$$

These points can then be projected onto the camera image by making use of the camera intrinsic calibration parameters as well as its extrinsic ones with respect to the vehicle frame.

Figure 2 illustrates the projection of some map features onto an image. The map feature points are supposed to represent the ground level projection of the map features. The green point on the right side pictures is a well-projected map feature. The red ones represent ill-projected map features due to the parallel ground hypothesis being wrong. The black points depict projected map features that are actually not visible from the camera due to occlusions. The yellow points are distant features that are both occluded but also wrongly projected as they are all the more sensitive to the ground plane estimation. Finally, the blue circle shows an example of an unmapped feature. This shows that the projection of map features onto an image is not straightforward and needs further filtering and refinement.

B. Lidar-based refinement and filtering

In order to deal with the issues pictured in Fig. 2, we propose to use an additional lidar sensor to refine and filter the map feature projection. The lidar is used for two purposes.

Fig. 2: Naive projection of map features onto an image frame. Only the green point on the right-hand side can be considered as a correct annotation.

Fig. 3: Projection of map features onto an image frame using Patchwork++.

The first is to better estimate the ground surface on which to project the map features and the second is to assess whether or not a feature is visible in the image frame.

For this purpose, we make use of a Hesai Pandora sensor, which provides a 3D lidar sensor placed on top of five cameras: four grayscale wide-angle cameras to cover a 360° field-of-view and an additional front color camera. This setup allows to have almost the same field-of-view for both the lidar and the cameras. That is to say that if a feature is occluded from the camera perspective, it is also the case from the lidar one.

Although the two modalities share the same field-of-view, the lidar sensor is limited in terms of range. In our case, we consider a threshold of 150 meters above which we assume that the lidar cannot be used. Therefore, a first step was to remove the map features outside a radius of 150 meters around the vehicle. Note that these features could nevertheless still be visible in the image frame. This first filtering allows to remove the yellow crosses pictured in Fig. 2.

Next, the 3D point cloud $\mathcal{C} = \left\{ p_i = [x_i, y_i, z_i]^\top \right\}_{i=1,\dots,m}$ from the lidar is used to have a better estimate of the ground surface. In this work, we use the Patchwork++ ground segmentation method proposed in [16]. This algorithm separates all the lidar points into two groups, a ground points one $\mathcal{G}$ and a non-ground one $\overline{\mathcal{G}}$. The group composed of ground points is used to estimate the height of the map features. To do so, the map feature point ${}^{\mathrm{V}}P$ in the vehicle frame is transformed into a point ${}^{\mathrm{L}}P$ in the lidar frame using a similar equation as in (1) with the extrinsic calibration parameters encoding the relative position of the lidar w.r.t. the vehicle frame. Similarly to the vehicle localization, the calibration quality of the lidar, especially in terms of angle, has a higher impact of map features far away from the vehicle. Once a map feature has been projected in the lidar frame from a top down 2D space, we use a nearest neighbor approach to select the closest lidar point, categorized as belonging to the ground, and use its z-coordinate as a height estimate instead of the one from (2):

$$ {}^{\mathrm{L}}z := z_k, \text{ with } k = \underset{i,\ p_i \in \mathcal{G}}{\arg\min} \left\| {}^{\mathrm{L}}P - p_i \right\|_{2D}, \tag{3} $$

where the $\|\cdot\|_{2D}$ operator is the Euclidean distance but using only the x-y coordinates.

Figure 3 shows the lidar point cloud projected onto the image. The green dots correspond to point segmented as ground while the others are in red. Using this new height estimation, the red crosses on the left side in Fig. 2 are now correctly projected at the ground level in Fig. 3.

The final step is to remove the map features that are occluded by some obstacles. To do so, we first project all the lidar points onto the image as well as the map features. For each map feature, we estimate the average depth using the lidar points that surround it in the image frame within a given radius. This depth is then compared with the true distance separating the map feature and the camera frame. The idea is that if a map feature is occluded by the obstacle, such as the bus in Fig. 3, then the estimated depth will be shorter than the true distance. By setting a threshold on the distance difference, the occluded features can be removed. The black crosses in Fig. 3 are correctly classified as occluded and can be filtered.

These different steps enable to have a cleaner projection of the map features. However, we can see in Fig. 3 that some errors still remain. Because the ground segmentation can also be erroneous some wrong projection can still occur. We can also see the projection of the lidar onto the image is not perfect despite using the calibration parameters provided by the sensor manufacturer. And finally, the unmapped features remain unannotated.

IV. Point Annotation from Semantic Segmentation

The automatic annotation presented in the previous section is still imperfect due to various noises from localization, calibration, map errors or imperfect ground estimation. In order to assess the performance of a deep learning method for the detection of pole base features, we also study the use of manually labeled data.

To our knowledge, there exist no dataset with such annotations. Fortunately, pole-like features can be found in datasets dedicated to semantic segmentation. In this study, we consider the BDD100K dataset [12]. In this dataset, we merge the three categories “pole”, “traffic sign” and “traffic light” to represent

(a) Point annotation (b) Semantic segmentation

Fig. 4: Point annotations from semantic segmentation.

our map features. The annotations are provided at the pixel level, while in our case, we wish to mimic the point-wise annotations retrieved from an HD map. Therefore, for each cluster of annotated map feature pixels, we need to compute the position of the pole base.

Like in the previous section, the base of a pole may not be visible due to occlusions. Thanks to the pixel-wise annotations, we can define a pole base as the lower part of a map feature cluster that lies on top of a ground pixels. The ground is defined by merging the following classes: "road", "sidewalk" and "terrain". In addition to filtering out occluded poles, we also added a minimal width to the pixel clusters so as to filter out far away features. Figure 4 illustrates some results on the BDD100K datasets. The crosses represent the pole bases lying on the ground.

V. Pole base detector

In this study, we propose to formalize the detection of the pole base as an object detection problem. Therefore, bounding boxes are used to represent the pole base. At first glance, using a box to encode a point may not be relevant, especially for pole-like features that have thin appearances. In our case, the boxes are not "bounding" an object as in classical object detection approaches. Here, the boxes are used to encode the context surrounding the pole base while the centers of the boxes represent the points we aim at detecting. The size and shape of the boxes are meta-parameters studied in the experimental process. Using a bounding box formalism also allows to make use of highly efficient object detectors such as YOLO. In this paper, the YOLOv7 [17] algorithm is used.

To evaluate the performance of the detector, we consider two types of metrics. First, the traditional object detection metrics are used (mAP, precision and recall), in order to evaluate the ability of the model to predict the right boxes. Note that we will be using the notation mAP 0.5:0.95 throughout this article, which refers to the mAP at different IoU thresholds, a metric commonly used in object detection as less saturated. Then, to have a finer evaluation of the point-wise accuracy of the box centers, we compute the horizontal distance with respect to the point annotations. This choice is motivated by the fact that within a monocular-camera localization context, the horizontal coordinate of the detection points in the image frame can be used for bearing-only localization. Therefore, the accuracy along the horizontal axis becomes more important than along the vertical one. Nevertheless, a Euclidean 2D distance could also be used instead to compute the point-wise accuracy.

TABLE I: BDD100K dataset information

	Number of images	Number of poles
Training	7628	9357
Validation	372	909

VI. Experimental results

Our experiments were divided into two parts. First, we validated the YOLO-based pole base detector using data from the BDD100K dataset. These two cases represented the ideal context where the pole base features could be assumed to be perfectly annotated manually. Then in a second step, we used an HD map to automatically annotate images. This was done using an experimental Renault ZOE car equipped with a Pandora sensor and driving in the city of Compiègne, France.

A. BDD100K

Before trying to learn a pole base detector using our HD map aided automatic annotations, that is known to be partially imperfect, we first validated it using data from the BDD100K dataset. The annotation procedure described in Sec. IV was applied to all the images from each dataset. In addition, for the validation set, the annotations were manually reviewed in order to remove some remaining erroneous annotations generated from the semantic segmentation ground truth. Table I gives the number of training and validation examples for each dataset.

As stated in the previous section, we used an object detection approach with bounding boxes. The pole base features were encoded by a box centered on it and with fixed size. The choice of the box width and height then arose. We decided to first experiment on BDD100K with squared boxes of various sizes ranging from 60 to 400 pixels. We used the official implementation of YOLOv7, with all default hyperparameters. The model was initialized with the weights from MS COCO. The training took 18h roughly on a single Tesla V100 32G GPU for 300 epochs. Most of the training converged after 100 epochs.

The detection metrics for all of these models are detailed in Table II. When boxes were too small, typically less than 100 px, the model performs poorly. The performance of the detector gets better and stable starting from 100 px. One plausible explanation for this could be that small boxes do not contain enough local context. When gradually increasing their size, the boxes aggregate more and more information about the surroundings of the pole until these bits of information become enough for the model to recognize and discriminate the pole. However, increasing the size further has little impact, since the box already contains enough context. The results with large enough boxes reached almost 80% precision and more than 60% recall.

The results provided in Tab. II quantifies the detection of the pole base at the box level. Next, we studied the accuracy of coordinates of the center of the boxes w.r.t. the pole base annotations. As stated previously, we only focused on the horizontal accuracy along the x-coordinate in the image

TABLE II: Detection metrics after 100 epochs of training with different box sizes on the validation set of BDD100K.

Box size	mAP 0.5:0.95	Precision	Recall
60x60	22.4%	52.0%	48.1%
80x80	35.8%	59.5%	56.4%
100x100	54.9%	75.8%	63.7%
200x200	59.9%	79.2%	**63.8**%
300x300	57.0%	74.6%	63.2%
400x400	**63.8**%	**79.6**%	62.5%

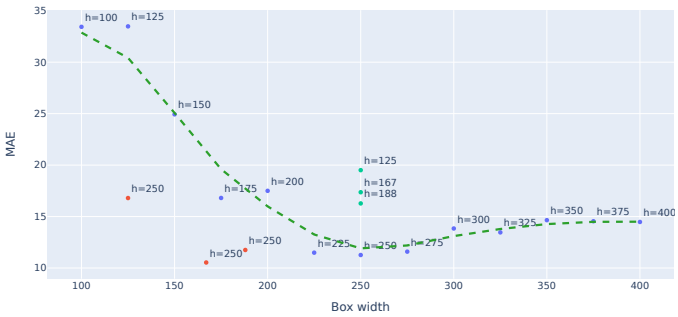

Fig. 5: MAE for different box sizes and ratios. The blue points represent squared boxes. The red points are non-squared boxes with a fixed height of 250px. The green points are non-squared boxes with a fixed width of 250px.

frame. Using all the true positive predicted bounding boxes, we computed the mean absolute error (MAE) between the predicted x-coordinate and the labeled one. The blue points in Fig. 5 represent the results obtained for different box sizes. The green dashed line was obtained by sliding a 1D Gaussian kernel over these data points.

Although the performances got relatively stable in terms of mAP, precision and recall when the box sizes got larger than 100 px, the resulting MAE exhibited more variations.The 250px model did best while more than halving the error compared to the 100px model. Like in the previous case, we observed an improvement of the performance when increasing the box sizes up to a point (here 250px) after which the performance became stable with even a slight decrease.

Finally, we studied the use of non-squared boxes. We started from a squared box model with $(h, w) = (250, 250)$, the best performing one so far. Then, we tried shrinking either the height or the width of the box to see the effect on the performances. We tried with the following width:height aspect ratios: $\{(1{:}1), (1{:}2), (2{:}3), (3{:}4), (4{:}3), (3{:}2), (2{:}1)\}$.

In Fig. 5, the green points represent non-squared boxes with height smaller than 250px. We can see that MAE got worse when the height decreased. On the contrary, when reducing the width but keeping the height at 250px, represented by the red points in Fig. 5, we see that MAE remained stable until the width became less than 150px. These results show that the vertical context was much more important than the horizontal one. This made sense as poles are typically vertical structures.

Fig. 6: Experimental Renault ZOE vehicle equipped with a NovAtel SPAN-CPT GNSS/IMU and a Hesai Pandora sensor.

B. City of Compiègne

1) Data collection: An experimental Renault ZOE vehicle was equipped with the following sensors:

- NovAtel SPAN-CPT GNSS/IMU with post-processed PPK computations for centimeter-level accuracy localization (50 Hz).
- Hesai Pandora 40-layer LiDAR with a horizontal angular resolution of $0.2°$ and integrating five synchronized cameras (10 Hz) (a front color camera, and four wide-angle mono cameras). For this work, only the color camera was used.

The extrinsic calibration between the Pandora lidar and the SPAN was obtained using a high-accuracy FARO Vantage laser tracker. For intrinsic calibration of cameras and the lidar, factory settings were used. Figure 6 shows the experimental platform along with the different working frames.

The recorded sequence illustrated in Fig. 1 covered the entire city of Compiègne, France, with diverse road contexts. It lasted 40 minutes for a cumulative distance of 13.5 km. The HD map used in this experiment was acquired in 2019. In the meantime, some new poles were added and others were removed. Additionally to annotation errors due to localization, calibration or projection, some other errors were also due to erroneous HD map data.

2) Training sets: In the driving sequence illustrated in Fig. 1, the blue part of the trajectory was driven in an area where the environment was not mapped. Therefore, we used this part of the sequence as the validation set and manually annotated the images in order not to bias the evaluation part. For this part, only one image per second was kept. The rest of the sequence was used as our training set and the HD map was used to automatically annotate the images. In zones where the vehicle was stationary, the images were subsampled in order not to have too many almost duplicated examples.

The sets obtained are described in Table III. It is important to note that although the number of images seems larger compared to BDD100K, there is a lack of variability as they correspond to a unique driving sequence. Similarly, the 6724

TABLE III: Heudiasyc dataset information

	Number of images	Number of poles
Train	15724	6724
Validation	605	2148

TABLE IV: Map-aided annotation parameters

Maximum distance between a pole and the vehicle	150m
Image search radius for pole lidar-based filtering	20px
Depth difference for pole lidar-based filtering	5m

TABLE V: Detection metrics after 100 epochs of training on the validation set of Compiègne.

Model	mAP 0.5:0.95	Precision	Recall
BDD100K	49.3%	66.9%	61.4%
Compiègne	39.5%	62.4%	58.8%

annotated poles actually correspond to a subset of 1600 unique poles in the HD map.

3) Map-aided annotation: To annotate our images using our HD map and the lidar, we set the parameters of the filtering described in Sec. III to the values detailed in Table IV.

Figure 7 illustrates examples of map-aided annotated images from our dataset in different situations. The first row corresponds to a naive projection of the HD map features with a simple filtering of far away features. The second row adds a lidar-based filtering to remove the occluded features. The last row adds an additional ground segmentation to refine the height estimation. In average, adding each of these additional filtering improve the annotations but occasionally some noises can also be introduced due to wrongly segmented ground or bad depth estimation. In addition, the unmapped features shown in Fig. 7e can not be dealt with our current approach.

Using these automatically annotated training data, we trained a model in the same way as we did on the BDD100K dataset. Table V shows the detection performances on the Compiègne validation set comparing the model trained on BDD100K to the one trained in Compiègne. We can see that the performance of the model trained on the Compiègne dataset performed worse than the one trained on BDD100K with a difference of almost 10% in terms of mAP. Figure 8 illustrates a detection example using both models on a Compiègne validation image.

The main reason is likely to come from the imperfect annotations despite the lidar-based refinement and filtering. A second reason may come from the low variability of the training data which was extracted from a single driving sequence. However, the automatic annotation framework introduced in this work enables to easily add supplementary training data from new driving sequences without additional cost in terms of data labeling.

VII. Conclusion

In this paper, we introduced a framework to use an HD map to automatically annotate images. The map features were encoded as pole bases and projected onto images while a lidar was used to filter occluded features and to better estimate the ground surface. Although a pole base was annotated by a single point, we demonstrated that it could be detected using a traditional object detector using bounding boxes. This approach was first validated from annotations generated from semantic segmentation on the BDD100K. We showed that using a sufficiently large box, especially in terms of height, is necessary for the object detector to perform properly. The method was then tested using annotations generated from an HD map of the city of Compiègne.

This preliminary work opens room to further improvements. The lidar-based filtering and refinement could be improved by tuning better the different parameters, but also with more sophisticated geometric consideration. The automated annotations, which are still imperfect, can also serve as an initial annotation either for a human operator or as an input for a second stage refinement. The ability to have a low-cost annotation pipeline could be exploited in many contexts such as generating training data in different driving weather or lighting conditions without having the need to label the data from scratch. The HD map used in this work also contained other features such as lane markings. This approach could also be extended to annotate data to train a lane marking detector. Finally, the use of the proposed pole base detector will be studied within a localization context in future work.

Acknowledgment

This work has been funded by the European project ERASMO (GSA/GRANT/03/2018) in the framework of the SIVALab laboratory between Renault and Heudiasyc. The authors would like to thank Rémy Huet and Vincent Brebion for their technical support in this work.

References

[1] L. Li, M. Yang, L. Weng, and C. Wang, "Robust localization for intelligent vehicles based on pole-like features using the point cloud," *IEEE Transactions on Automation Science and Engineering*, pp. 1–14, 2021.

[2] M. Sefati, M. Daum, B. Sondermann, K. D. Kreiskother, and A. Kampker, "Improving vehicle localization using semantic and pole-like landmarks," in *IEEE Intelligent Vehicles Symposium*, Los Angeles, CA, USA, June 2017, pp. 13–19.

[3] R. Spangenberg, D. Goehring, and R. Rojas, "Pole-based localization for autonomous vehicles in urban scenarios," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Daejeon, South Korea, Oct. 2016, pp. 2161–2166.

[4] M. Gouda, A. Shalkamy, X. Li, and K. El-Basyouny, "Fully automated algorithm for light pole detection and mapping in rural highway environment using mobile light detection and ranging point clouds," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2676, no. 7, pp. 617–629, July 2022.

[5] M. Lehtomki, A. Jaakkola, J. Hyypp, A. Kukko, and H. Kaartinen, "Detection of vertical pole-like objects in a road environment using vehicle-based laser scanning data," *Remote Sensing*, vol. 2, no. 3, pp. 641–664, Feb. 2010.

(a) Nominal case (b) Good ground fitting (c) Unfiltered poles (d) Non-annotated poles (e) Unmapped features

Fig. 7: Examples of map-aided annotated images from our dataset in different situations. The first row corresponds to a naive projection of the HD map features with a simple filtering of far away features. The second row adds a lidar-based filtering to remove the occluded features. The last row adds an additional ground segmentation to refine the height estimation.

(a) Ground truth (b) Prediction from BDD100K model (c) Prediction from Compiègne model

Fig. 8: Ground truth and predictions from models trained on BDD100K and Compiègne on a Compiègne validation image. The green crosses are true positives, red ones false negatives and the blue circles are misdetections.

[6] B. Rodrguez-Cuenca, S. Garca-Corts, C. Ordez, and M. Alonso, "Automatic detection and classification of pole-like objects in urban point cloud data using an anomaly detection algorithm," *Remote Sensing*, vol. 7, no. 10, pp. 12 680–12 703, Sept. 2015.

[7] F. Ghallabi, G. El-Haj-Shhade, M.-A. Mittet, and F. Nashashibi, "Lidar-based road signs detection for vehicle localization in an HD map," in *IEEE Intelligent Vehicles Symposium*, Paris, France, June 2019, pp. 1484–1490.

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Computer Vision – ECCV*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

[9] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.

[10] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A dataset for semantic scene understanding of lidar sequences," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, October 2019.

[11] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2016, pp. 3212–3223.

[12] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "BDD100K: A diverse driving dataset for heterogeneous multitask learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2020, pp. 2636–2645.

[13] H. Dong, X. Chen, S. Srkk, and C. Stachniss, "Online pole segmentation on range images for long-term lidar localization in urban environments," *Robotics and Autonomous Systems*, vol. 159, p. 104283, 2023.

[14] C. Sun, J. M. U. Vianney, Y. Li, L. Chen, L. Li, F.-Y. Wang, A. Khajepour, and D. Cao, "Proximity based automatic data annotation for autonomous driving," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 395–404, 2020.

[15] W. H. Lee, K. Jung, C. Kang, and H. S. Chang, "Semi-automatic framework for traffic landmark annotation," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 2, pp. 1–12, 2021.

[16] S. Lee, H. Lim, and H. Myung, "Patchwork++: Fast and robust ground segmentation solving partial under-segmentation using 3d point cloud," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022, pp. 13 276–13 283.

[17] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint:2207.02696*, 2022.